FACTORS INFLUENCING PARTICIPANT SATISFACTION WITH FREE/LIBRE AND OPEN SOURCE SOFTWARE PROJECTS

BY

BRENDA LYNNE CHAWNER

A THESIS

submitted to the Victoria University of Wellington
in fulfilment of the requirements for the degree of
Doctor of Philosophy in

Information Systems

Victoria University of Wellington

2011

Brenda Lynne Chawner: *Factors Influencing Participant Satisfaction with Free/Libre and Open Source Software Projects*, A Thesis, © July 2011, Wellington, New Zealand

Some rights reserved: CC BY–SA

Dedicated to my parents, Grace and the late Bob Chawner, who helped me understand and appreciate the value of education and knowledge.

ABSTRACT

The purpose of this research was to identify factors that affect participants' satisfaction with their experience of a free/libre open source software (FLOSS) project. The research built on existing models of user satisfaction from the information systems literature, and also incorporated two characteristics of FLOSS projects first identified by Ye, Nakakoji, Yamamoto, and Kishida (2005), product openness and process openness. The central research question it answered was, *What factors influence participant satisfaction with a free/libre and open source application software project?*

Richard Stallman's reasons for setting up the GNU project and the Free Software Foundation arose from his frustration at being forced to be a passive user of software used for a Xerox printer. These suggest that being able to be an active participant in a FLOSS project is one factor that should be examined, and therefore the first sub-question this project answers is, *What types of contributions do participants make to free/libre and open source software projects?*

Several studies have shown that the extent of participation in a FLOSS project varies from individual to individual, and this variation leads to the second sub-question, *Do the factors that influence satisfaction vary for different types of participation? If so, in what way?*

A preliminary conceptual model of factors affecting participant satisfaction was developed, reflecting the key concepts identified in the literature. The main theoretical goal of this research was to test the model using empirical data.

The research used a sequential, mixed methods approach. The first, qualitative stage involved reviewing documents from selected projects and interviewing a purposive sample of FLOSS project participants. The second, quantitative stage involved an online survey of FLOSS project participants, and the data gathered were used to test the conceptual model.

The results of the first stage showed that participation in FLOSS projects was a more complex construct than previously reported in the literature. Seven distinct categories of activities were identified:

- use;
- interaction with code;
- supporting the community;
- outreach;
- sponsorship;

- management; and
- governance.

Four attributes that modified these categories were also identified: organisational focus, role formality, remuneration, and time commitment.

Data from 154 responses to the online survey were used to test the model using stepwise multiple regression, which determined the effect of each of the variables on overall participant satisfaction. Moderated regression analysis was used to test the effects of three potential moderating variables. The results showed that that perceived system complexity had the largest effect, decreasing satisfaction if respondents perceived that the software was complex, while project openness and perceived developer communication quality accounted for the most variance in satisfaction.

The main theoretical contribution of this research lies in its extension of satisfaction studies to FLOSS communities, showing that communication and openness are more important than in conventional software projects. Its practical contribution will help people involved in the management and governance of FLOSS projects to identify ways of increasing their participants' satisfaction, which may in turn encourage them to contribute more.

ACKNOWLEDGMENTS

This thesis would not have been possible without the encouragement and support of many people.

First, I would like to thank the people who agreed to be interviewed for the first stage of the research. They all found time to answer my questions, and gave me new insights into their perspectives on being involved with a free/libre open source software project.

The next group of people who need to be thanked are anonymous—the 205 people who completed the online survey, and provided the quantitative data needed to test the model and hypotheses. Without your contributions, there would have been little to write about.

When collecting data and writing this thesis, I used free software whenever possible, and so another group of people who need to be acknowledged are the developers of this software. These include the people who wrote NSurvey and LyX, not to mention LaTeX and TeX. In addition, Nick Mariette and André Miede, who developed the LaTeX and LyX templates used to format the thesis need to be thanked for making it so easy to produce a high-quality typeset document. The bibliography was formatted using BibTeX, and I found the BibDesk citation manager software very easy to use. Even the Kp-Fonts package used for the thesis qualifies as free software, since it was released under the GPL, so my thanks go to its developers as well. Last, but certainly not least, the many people who contributed to OpenOffice.org, particularly those involved with Calc and Draw, made it much easier for me to produce high-quality figures and graphs to include here.

To my supervisors, Sid Huff and Gary Gorman—thank you for having confidence in my ability to complete this project. Sid gave me constructive and detailed feedback on my drafts, and Gary encouraged me to take a closer look at ontology and epistemology. Gillian Oliver also read the final draft of the thesis, and made valuable comments.

My colleagues in the School of Information Management were also supportive, and will no doubt be pleased that I have finally finished this research. In addition, much of this thesis was written while I was based at the School of Library and Information Studies at the University of Alberta in Edmonton, Canada. The School provided me with a spacious office while I was there, and Ann Curry and Heidi Julien gave me ongoing support and encouragement.

My husband, John Rankin, also deserves my thanks, for being there when I needed someone to talk to, and for being such a dedicated proofreader. He was also my LaTeX expert, helping me when I found myself out of my depth with its intricacies. My examiners, Kevin Crowston from

Syracuse University, James Noble, from Victoria University of Wellington, and Arvind Tripathi, from the University of Auckland, provided detailed and constructive feedback, which helped me improve the final version of this thesis.

Last, but certainly not least, I'd like to thank Richard Stallman, without whom there would have been no topic. His comment several years that the results of my research would be "very interesting" made me smile, and helped me stay motivated to finish.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

FLOSS Free/Libre and Open Source Software

IS Information Systems

LIM Library and Information Management

1

INTRODUCTION

The topic of this thesis, factors that influence satisfaction with free/libre and open source software (FLOSS) projects, is motivated by a story about a jammed printer and a sense of frustration. This was documented by Williams in his 2002 biography of Richard Stallman (p.1–4). The following version is based on this, but has been extended with details provided by Stallman in a series of personal communications (October 2008).

1.1 PAPER JAMS AND THEIR CONSEQUENCES

Thirty years ago, Richard Stallman, who was then working as a systems programmer in the Massachusetts Institute of Technology Artificial Intelligence Laboratory (AI Lab), had a problem. Xerox had given the AI Lab one of its latest laser printers, which was based on photocopier technology. This machine had many advantages: it used ordinary paper and was very fast; however, once installed, it turned out to be subject to frequent paper jams. This was annoying to everyone working in the lab; they would go to the printer for their output, only to find that the machine needed attention. Once someone resolved the issue, everyone then had to waste time waiting for it to print their work. The lab's previous printer had also suffered from paper jams which caused delays; Stallman's solution to this involved modifying its source code so that everyone whose print job was affected by a malfunction was sent a message asking them to check the printer. As a result, printer problems were resolved as quickly as possible, rather than waiting for someone to notice them. However, this time he was frustrated—no source code was available for the new Xerox printer, and he was therefore unable to implement a similar solution. When he did track down a computer scientist at Carnegie Mellon University who had a copy of the source code he wanted, Stallman was refused access to it because the scientist had signed a non-disclosure agreement with Xerox.

This ran counter to Stallman's experience at Harvard's Cruft Lab and in the MIT AI Lab. According to the prevalent 'hacker ethic', code was meant to be freely shared, and it was a programmer's duty to share innovations with others on request. Stallman had been familiar with this co-operative behaviour since the early 1970s, when Ed Taft, a systems administrator at the Cruft Lab, told him that the lab had a policy of refusing to install software unless the source code could be displayed for users. Taft also said that this was considered to be appropriate behaviour

for an educational institution (personal communication, R. M. Stallman, 23 October 2008).

Stallman's frustration with an increasing trend to restrict access to source code led him to start the GNU project in 1983; its objective was to develop an operating system that users were free to modify. This was followed two years later by his establishing the Free Software Foundation, whose mission is "to promote computer user freedom and to defend the rights of all free software users" (Free Software Foundation, 2010). Stallman identified four essential computer user freedoms:

- the freedom to run the program, for any purpose (Freedom 0);
- the freedom to study how the program works, and adapt it to local needs (Freedom 1);
- the freedom to redistribute copies so others can benefit from the software (Freedom 2);
- the freedom to improve the program, and release the improved version to the public, so that the community can benefit (Freedom 3) (1999, p.56).

In order to exercise their rights under Freedoms 2 and 4, users need access to a program's source code. Stallman's frustration with the AI Lab's Xerox printer would have been easily resolved if the license for the printer driver had provided this freedom. Any software user who has had an experience similar to Stallman's could potentially use Freedom 1 to resolve their frustration by taking advantage of this provision, if the software complied with the Free Software Definition.

Now, over 25 years later, the impact of Stallman's actions is considerable. Free (as in freedom) software is now used in millions of computers worldwide, with thousands of developers involved in projects to write software released under licenses which guarantee access to source code. The following examples show that this impact covers all aspects of computer use, such as operating systems, database management software, web servers, programming languages, and end-user application software.

In 2008, VDC reported that GNU/Linux was the most popular operating system chosen for embedded systems, growing from 15.5% in 2004 to 18% in 2008 (Linux Devices, 2008). Netcraft's regular surveys of web servers consistently show that Apache is used on over 50% of the world's web servers (Netcraft 2010). Over 3700 individuals, working for over 300 companies, have contributed to the Linux kernel (Kroah-Hartman, Corbet, and McPherson, 2009). Programmers and web developers have a choice of free languages, including Perl, Python, and PHP; MySQL and PostgreSQL provide relational database capabilities for large-scale application development. Prominent companies such as Amazon.com and Google make extensive use of free software in their technical infrastructure (Dahl, Banerjee, and Spalti, 2006, p.25). Computer users

can choose from the OpenOffice productivity suite (providing word processing, spreadsheet, presentation, graphics and database functionality), the Firefox web browser, the Thunderbird email client, the GIMP image editor, and the WordPress blogging engine, to name just a few. All users of these (and all) FLOSS packages have the right to improve them to suit local needs, assuming they have access to the necessary technical skills.

The free software movement, and the related open source development process, which split off from the original movement, taking on on a separate identity in 1998[1], have attracted considerable attention from academic researchers, particularly in the last 10 years. Recent literature reviews by Aksulu and Wade (2010), and Crowston, Wei, Howison and Wiggins (2012, in press) show the breadth and depth of this research. Specific research topics have included:

- demographics of the FLOSS developer community (e.g. Dempsey, Weiss, Jones, and Greenberg 2002; David and Shapiro 2008);
- the open source development process, which is often perceived as lacking formal structure, in contrast to traditional approaches to software development (e.g. Scacchi 2004);
- the motivation and beliefs of individual developers (e.g. Hars and Ou, 2002; Hann, Roberts, and Slaughter 2004); and
- project culture (e.g. von Krogh, Spaeth, and Lakhani 2003).

However, little research has been conducted with respect to Stallman's main motivation for setting up the GNU project and the Free Software Foundation: a lack of satisfaction caused by his inability to modify the source code for the printer in order to make it work more effectively. The goal of this project was to fill this gap by identifying factors that affect participant satisfaction with free/libre and open source software projects. The research extended existing models of user satisfaction from the information systems literature, in particular those of Mahmood, Burn, Gemoets, and Jacquez (2000), McKeen, Guimaraes, and Wetherbe (1994), and Guimaraes, Staples, and McKeen (2003), by incorporating two characteristics of FLOSS projects first identified by Ye, Nakakoji, Yamamoto, and Kishida (2005).

1.2 RESEARCH QUESTIONS

The central research question this study answers is:

> *What factors influence participant satisfaction with a free/libre and open source application software project?*

Stallman's reasons for setting up the GNU project and the Free Software Foundation came from his frustration at being forced to be a passive

1 The distinction between 'free software' and 'open source' is explained more fully in Section 2.2 on page 15.

consumer ("user") of the software used in the Xerox printer This suggests that being able to be an active participant in the project is one factor that should be examined, and therefore the first sub-question this project answers is:

> *What types of contributions do participants make to free/libre and open source software projects?*

Existing models of participation in a FLOSS project show that different people are involved in different ways (see for example Crowston, Wei, and Howison 2006), and this variation leads directly to the second sub-question:

> *Do the factors that influence satisfaction vary for different types of participation? If so, in what way?*

1.3 SIGNIFICANCE OF TOPIC

Studies of customer satisfaction are common in the retail and service industries. Allen suggested that these are popular because satisfaction research has shown that there is a relationship between the level of satisfaction and a desired business outcome (2004, p.2). These desired outcomes include increased sales and long term customer retention. Denove and Power showed that there is a positive relationship between customer satisfaction with an automobile brand and sales of the brand (2006, p.5). In particular, brands with high customer satisfaction ratings increased their sales by 44% between 1998 and 2003, while sales for brands with low satisfaction ratings decreased by 4% (p.6). Allen acknowledged that factors influencing customer retention are complex, but said that empirical research suggests that customers who have positive experiences are most likely to continue to use a product or service, while those who have unresolved problems are less likely to be loyal to the product or service (2006, p.6). He also suggested that it is less expensive to retain an existing customer than to attract new ones (p.115).

Other evidence of the importance of satisfaction and its measurement to modern businesses came from the researcher's recent experiences. In July 2010, a sign on the door of my local branch of a Canadian bank said "We want you to be very satisfied with your visit today". This suggests that the bank's managers see satisfaction as a key measure of their customers' perceptions of the quality of the service they receive. In October 2010, I received an invitation to complete an online survey measuring my satisfaction with a recent stay in a large hotel chain. My responses included a mild complaint about the complexity of setting up an Internet connection, which resulted in a personal message from the hotel manager saying that he would look into the issue. These experiences show that two very different businesses are applying the results of satisfaction research in their interactions with their customers, in-

dicating that satisfaction research has practical as well as theoretical implications.

Both of the previous examples show that satisfaction is being used as an indication of service quality. This makes intuitive sense because satisfaction (and its opposite, dissatisfaction) can have powerful effects. Blackshaw suggested that someone who has an unsatisfying experience is likely to influence many more people than someone who is satisfied (2008, p.1–2). This means that it is important to understand the nature of satisfaction, how to measure it, and what factors influence someone's satisfaction with a project, organisation, or software package, since no one, whether a company director, an IT manager, or a software developer, wants to have unhappy users of their products, services, or software discouraging other people from using it.

The apparent lack of previous research into factors that affect participant satisfaction with a free/libre and open source project means that project developers who wish to take steps to maintain or increase people's satisfaction with their projects currently have little evidence to act on, and must rely on intuition or anecdotes, rather than data. The results of this research are intended to fill that gap.

1.4 VALUE OF RESEARCH RESULTS

The results of this research have both theoretical and practical value. These are discussed in more detail in Chapter 9 on page 209, and are summarised below.

Its theoretical contribution lies in two main areas:

- the results include a multidimensional framework for understanding types of contributions to FLOSS projects, extending the more conventional code-centric, developer-focused models that dominate the FLOSS research literature, by including categories relating to the wider user and stakeholder community, such as project governance; and
- the research extends models of factors that influence satisfaction with software to a FLOSS context, and shows that these follow a different pattern to satisfaction models for conventional software projects.

These findings will benefit people who participate in FLOSS projects, particularly those who are involved in management, community-building, or governance activities. The results showed that project openness, or the extent to which the project is perceived as welcoming new community members and encouraging their contributions, has the strongest relationship with satisfaction, followed by the perceived quality of developer communication. In addition, perceived complexity was found to decrease satisfaction. Understanding the importance of these character-

istics will allow people involved in managing or governing individual projects to review and improve the way they present their projects to others, and to put policies in place to ensure that the quality of communication is high.

Practitioners in the field of library and information management who are interested in becoming involved with a FLOSS project may also benefit from the results of this research, since it gives them a framework to identify different ways in which they could participate in projects.

1.5 THEORETICAL APPROACH

Previous research has identified three groups of characteristics that have a significant effect on satisfaction in conventional software development projects:

- features of the software itself, such as perceived complexity, usefulness, and ease of use;
- aspects of the overall development process, such as the quality of developer communication and perceived user influence; and
- attributes of individual users, such as their education, experience, and training.

This thesis built on existing models of user satisfaction from the information systems literature, in particular those of Guimaraes, Staples, and McKeen (2003), Mahmood, Burn, Gemoets, and Jacquez (2000), and McKeen, Guimaraes, and Wetherbe (1994), incorporating two characteristics of FLOSS projects first identified by Ye, Nakakoji, Yamamoto, and Kishida (2005), product openness and process openness. In addition, this research extended the satisfaction models so that they could apply to any participant in a FLOSS project, rather than limiting them to users, since the boundaries between roles have been found to be less clearly defined than in conventional software development projects (Gacek and Arief 2004).

1.6 RESEARCH APPROACH

The research used a sequential, mixed methods approach, which suited its post-positivist epistemology. Post-positivism treats knowledge as being objective and measurable, but acknowledges that this knowledge may also be imperfect or incomplete. By using a sequential, mixed methods approach, data to answer the research questions were gathered using complementary techniques, in order to provide multiple perspectives and, at least partially, compensate for imperfections or incompleteness.

Once the preliminary research area had been determined, a literature review was conducted, in order to develop a deeper understanding of

the research context. This focused on two main areas: FLOSS project structure and participant activities, and satisfaction, particularly user satisfaction with software projects. Once the literature review was complete, specific research questions, a preliminary conceptual model, and hypotheses were developed, reflecting the key concepts identified in the literature.

This was followed by a qualitative stage which involved reviewing documents from selected projects and interviewing a purposive sample of FLOSS project participants, chosen to represent a range of roles and types of projects. The results of a content analysis of the observations and interviews were used to develop a user-centric model of FLOSS participation, and to review the preliminary conceptual model and hypotheses. The final quantitative stage involved an online survey of FLOSS project participants, and the data gathered were used to test the conceptual model and hypotheses, leading to a final, revised model.

1.7 DELIMITATIONS

The most important delimitation of this research was that it was restricted to participants in FLOSS application projects intended for use in library and information management, rather than in the full spectrum of FLOSS projects. The definition of the term application software in Section 1.8.1 on the following page shows that this type of software is intended for use by end users to carry out their work or business tasks. In addition, this type of software is typically adopted by an organisation, rather than by an individual. This means that the findings cannot necessarily be generalised to all types of FLOSS projects, or even to all FLOSS application projects.

Another delimitation was that the interviews and survey were conducted in English, limiting participation to people who spoke the language. This means that the results may not be generalised to all participants in FLOSS application software projects. In particular, it may not represent the views of people in Europe, or in developing countries such as India and Pakistan, who are heavily involved in using and developing FLOSS.

1.8 TERMINOLOGY

The term 'open source', now widely used, was coined by a group of 'free software' proponents (among them Eric S. Raymond and Tim O'Reilly) in February 1998 as a marketing device to overcome what they saw as an anti-business bias of the free software movement started 13 years earlier by Richard Stallman (DiBona, Ockman and Stone 1999, p.3; Feller and Fitzgerald 2002, p.38). They established the Open Source Initiative to promote the Open Source Definition (OSD) and certify software

licences as being OSD compliant. Stallman has continued to promote the term 'free software', and argues that there are basic ethical and philosophical differences between the two movements (Stallman 2002). These differences primarily relate to terminology. Stallman prefers to emphasise the importance of freedom and the rights of users to access and modify source code for all software they use, as defined in the Free Software Definition. In contrast, the Open Source Definition emphasises licence requirements, with a focus on developers' rights.

It is important to note that the term 'free', as used by Stallman, does not refer to cost, but is instead about 'freedom'—the freedom of software users to modify source code for their own purposes, and to redistribute the changes freely. While the two movements represent different underlying philosophies, they have similar objectives, and in the spirit of compromise, this thesis uses a hybrid term, Free/Libre and Open Source Software (FLOSS). The term 'libre' is French, and has the same root as the English word 'liberty', a synonym for freedom. By including it in the acronym used throughout this thesis, the researcher is emphasising the 'freedom' aspect of FLOSS, to avoid implying that 'free software' must be free of cost.

1.8.1 *Definitions of key terms*

The definitions below are based on the literature review carried out as part of the study.

Application software

Software that is designed for direct use by an end user, and performs 'real world' tasks. Examples of application software include word processors, spreadsheets, and web browsers. Application software also includes industry-specific software such as library management systems or digital library/institutional repository management packages.

Developer

A person involved in a software project who works directly with the source code, by either writing and testing new code or by changing existing code.

Free/libre open source software

Software released under a free/libre and open source licence, which ensures that the source code is available to users (and potential users), and allows them to modify it to suit their needs, as well as redistribute the source code, and their modifications, to others.

Participant

A person who contributes to a FLOSS project, for example by writing source code, fixing bugs, writing documentation, supporting other members of the project's community, taking part in project governance, or providing resources.

Satisfaction

"The satisfying of a need or desire as it affects or motivates behaviour" where 'satisfying' means "meeting or fulfilling the wish or desire or expectation of; to be accepted by (a person, his taste, judgement, etc.) as all that could be reasonably desired" ('satisfaction' and 'satisfy', Oxford English Dictionary Online, 2010).

Source code

Statements in a programming language that comprise a computer program or software package and are readable by humans.

User

A person who uses application software in the course of his or her normal tasks, but does not necessarily contribute to the source code.

1.9 STRUCTURE OF THIS THESIS

The remainder of this thesis is presented as nine chapters:

- Chapter 2 reviews relevant literature and concludes with a preliminary research model and hypotheses.
- Chapter 3 describes the methodology used in this study.
- Chapter 4 provides a demographic overview of the projects, interviewees, and survey respondents.
- Chapter 5 presents the results of the qualitative analysis of interview data relating to the types of contributions participants make to FLOSS projects.
- Chapter 6 presents the qualitative analysis of interview data relating to factors that influence FLOSS participant satisfaction and concludes with a revised research model and hypotheses.
- Chapter 7 analyses the survey data and tests the revised research model and hypotheses.
- Chapter 8 discusses the findings and their implications.
- Chapter 9 summarises the research project and presents suggestions for further research.

2

LITERATURE REVIEW

This thesis involves two main topics: participation in free/libre and open source software projects and participant satisfaction. Section 2.1 on the current page begins with a brief history of the development of FLOSS, followed by a discussion of the emerging FLOSS research literature, with an emphasis on project structure and participation. Section 2.5 on page 28 discusses the nature of satisfaction, focusing on selected information systems research literature dealing with satisfaction. This includes a small number of studies that have looked at satisfaction in the context of FLOSS projects. The chapter concludes with a preliminary research model of factors that are related to participant satisfaction with FLOSS, showing the relationships and hypotheses that were tested as part of this research.

Material for this literature review was collected using a range of techniques. The first step involved searching library catalogues and a range of databases including ProQuest, Library Literature and Information Science Fulltext, and the ACM and IEEE digital libraries, for material on satisfaction, participation, and FLOSS. Citations in recent review articles were used to locate older, classic articles, and Web of Knowledge was used to locate newer material that cited key early papers. In addition, conference web sites, such as the International Conference on Information Systems and HICSS (Hawaii International Conference on System Sciences), were searched for papers on these topics. Finally, FLOSS portals such as the MIT Free/Open Source Research Community portal (http://opensource.mit.edu/), the opensource.ucc.ie: resources for researchers (http://opensource.ucc.ie/), and FLOSS@Syracuse (http://floss.syr.edu/) were checked regularly for new material. Proceedings from FLOSS conferences and workshops, particularly those associated with the annual International Conference on Software Engineering and the IFIP Working Group on Open Source Software, were also checked for relevant papers.

2.1 A BRIEF HISTORY OF FLOSS

Sharing source code for software is not a new concept. In the late 1950s, when mainframe computers were programmed in assembly language, the Project for the Advancement of Coding Techniques (PACT) allowed programmers from Lockheed, Douglas, and North American Aviation to collaborate, even though the companies were competitors (Leonard 2000). Leonard quoted Wesley S. Malahn, a participant in the project, as saying that the main motivation for this sharing was to save coding

and machine time. In a summary of the results of a panel discussion on 'What is Proprietary in Mathematical Programming?', Smith (1961) reported that "[r]epresentatives of several organizations who are not in the business of supplying methods and codes indicated that they usually gave away general-purpose computer routines", and justified doing so because they had benefited as the recipients of software shared by other organisations.

The Association for Computing Machinery (ACM) has published algorithms (known as CALGO, Collected Algorithms of ACM), in a variety of formats and publications since 1960 (Hopkins 2002). Hopkins noted that "[t]he idea was to provide a means for programmers to make available their coded versions of algorithms for both pedagogical and reuse reasons". Graham (2001) suggested that another reason was to validate research results. In software research, the 'result' is often source code, and other researchers need to be able to execute the code in order to validate claims made about a new technique. Access to the source code may also be needed for this, particularly if there are unexpected results. Initially the CALGO algorithms were published with a statement saying that "the reproduction of algorithms is explicitly permitted ... without any charge" (Herbold 1960) making this an early example of FLOSS. However, even though the ACM continues to distribute algorithms (now in electronic form), its current licence limits the use, modification, and redistribution of CALGO to "academic, research, and other similar non commercial uses" (Association for Computing Machinery, 1998), making the current version of CALGO open only to the specified groups, rather than fully compliant with the Free Software Definition.

With the rise of commercial computing in the 1960s and 1970s, most software became closed or proprietary, and sharing source code was largely restricted to computer science academics and students. Some software, such as Donald Knuth's TEX, continued to be released along with source code. Knuth has provided extensive documentation of his experiences in publishing the TEX source code (Knuth 1992a; 1992b), saying "the real test begins as people with many different viewpoints undertake their own experiments" (Knuth 1992a, p.262). This suggests his goal in releasing the source code was to improve its quality—both in design and reliability (or freedom from bugs). Throughout the 1980s, publishing source code for educational purposes continued. Bentley's *Programming Pearls* (1986, revised 1989) and *More Programming Pearls* (1988), compiled from his columns in *Communications of the ACM* are examples of this; the source code for his examples is now available on the Web at http://www.cs.bell-labs.com/cm/cs/pearls/code.html. The conditions of use say "You may use this code for any purpose, as long as you leave the copyright notice and book citation attached." There are many other examples of source code being published for teaching purposes—two library examples are Davis and Lundeen's *Illustrative computer programming for libraries: selected examples for information spe-*

cialists (1981) and Cooper's *Design of library automation systems: file structures, data structures, and tools* (1996).

AT&T's development of Unix was the next major event in the FLOSS time line. In 1956, United States federal government decreed that AT&T was forbidden to enter markets such as computing, and that it was required to license its patents (Leonard 2000). This meant that, at least initially, the source code for Unix, which was developed at AT&T's Bell Labs in 1969, was distributed to universities and research institutes for a nominal fee. The licensing terms allowed the software to be modified and redistributed to organisations who held an AT&T licence. The University of California at Berkeley was able to use its experience with this to develop a freely-redistributable version of Unix (commonly known as BSD Unix, from Berkeley Software Distribution). Licensing issues that occurred as a result of the commercial release of Unix following the break up of AT&T in the 1980s, and a subsequent lawsuit, led to the Berkeley-based project being disbanded in 1995. BSD Unix lives on, though, in several variations, and is the foundation of the current Macintosh operating system, Mac OS X.

In September 1983, Richard Stallman announced the GNU project to develop a wholly free operating system, and in early 1985, he established the Free Software Foundation (FSF). Its main objective is to promote the concept of 'free software', where the term 'free' refers to freedom, not price. The GNU (a recursive acronym for Gnu's Not Unix) project has produced a range of software, including GCC (GNU Compiler Collection,) and the multipurpose Emacs editor. Stallman's background and motivation for establishing the FSF are well documented (see Moody 2002 for one example); to him, free software is an ethical obligation. The most popular FLOSS licence, the GNU General Public License (GPL), was developed for the GNU project; it has been in use since 1999, with several revisions to reflect changes in the software development environment. The current version is GPLv3, released in 2007 (Smith 2010).

The terms of the GPL and other FLOSS licenses are often misunderstood. The GPL does not require users to contribute their local changes back to the community, unless they are distributed to third parties. If this is the case, the changes must be made available to any interested party on the same terms as the original licence.

The next milestone in the FLOSS timeline came in 1991 when Linus Torvalds, then an undergraduate computer science student at the University of Finland, began what some regard as the FLOSS community's most significant achievement, the Linux operating system. Glyn Moody's *Rebel code: Linux and the open source revolution* (2002), provides a detailed description of its development. Torvalds began by writing an operating system kernel; a kernel (also known as a nucleus) is a low-level component of an operating system that manages the computer's resources and their interactions (Hansen, 1970). The Linux kernel is typically used in combination with other software, much of it originating

from the GNU project. Torvalds felt that making the source code for his kernel freely available through FTP[1] was a better option than making it available as shareware, which would require people to pay a fee. Torvalds disliked paying fees, but felt guilty for not paying them, which was one of his reasons for releasing his kernel at no charge (Moody 2002, 44–45); however, he also encouraged others to modify and improve it, saying "I've enjouyed [sic] doing it, and somebody might enjoy looking at it and even modifying it for their own needs." (Torvalds 1991). Linux versions 0.02 through 0.11+VM were released under Torvalds' own licensing terms, which required any redistribution to be free of charge, but with v. 0.12 he adopted the GNU GPL.

During the 1990s, Stallman and other GNU developers continued to work on free software projects. At the same time, Linux became more popular, attracting a growing community of developers. The movement to share source code and allow its free redistribution gained momentum in early 1998 when Netscape Communications decided to release the source code for the Netscape Communicator 4.0 web browser under a FLOSS license, hoping that would increase its popularity in the face of competition from Microsoft's Internet Explorer. This raised the profile of the open source movement, since at the time there was considerable interest in using more flexible web browsers as a replacement for the previously popular menu-based Gopher approach to access Internet resources. This had the effect of introducing the concept of open source software to a new group of people, since the Netscape browser was designed for end users, rather than being a tool for developers. This increased the visibility of FLOSS, and also resulted in many more developers releasing source code under FLOSS licenses. As a result, hundreds of thousands of FLOSS projects are now available. The list includes programming languages like Perl and PHP, development environments such as the LAMP (Linux, Apache, MySQL, and Perl/PHP) stack, and application software, all carrying on the code sharing practices that began in the early days of commercial computing.

FLOSS approaches have also become attractive to people developing software for specific application domains. One example of this is the field of library and information management. FLOSS information management software has been available since the late 1990s. The list of available software includes library management software (Koha and Evergreen), digital library/repository software (Greenstone, EPrints, DSpace, Kete, and Fedora), recordkeeping software (Alfresco) and software to provide access to resources in archives (Archivist's Toolkit and Archon).

This extensive activity to develop and share software under FLOSS licenses suggests that many people expect to gain benefits as a result. Studying the factors that influence participants' satisfaction with their

1 FTP stands for File Transfer Protocol, a commonly used technique for transferring a file from one computer to another via the Internet.

involvement in the projects will provide existing project community members with insights into which aspects of the projects most affect this.

2.2 DISTINCTIONS BETWEEN 'FREE/LIBRE' AND 'OPEN SOURCE' SOFTWARE

A literal interpretation of the phrase 'free/libre and open source software' emphasises giving software users access to the original code for the program. However, the official definitions of 'free software' and 'open source' cover other aspects which are important in understanding the distinguishing features of this type of software, and are described briefly in the following sections.

2.2.1 *The Free Software Definition*

The Free Software Foundation (FSF) maintains a formal definition of free software at http://www.fsf.org/philosophy/free-sw.html. It specifies four aspects of software freedom:

- the freedom to run the program, for any purpose (freedom 0)
- the freedom to study how the program works, and adapt it to local needs (freedom 1)
- the freedom to redistribute copies so others can benefit from the software (freedom 2)
- the freedom to improve the program, and release the improved version to the public, so that the community can benefit (freedom 3)

All four freedoms relate to the user, not the programmer or developer, and the goal of the FSF is to promote computer user freedom (Brown 2010). In order for software to qualify as 'free' under freedoms 1 and 3, a user must have access to the source code, since it is not possible to understand how a program works or change it (to take advantage of freedom 1) without being able to see the source code, and it is not possible to improve the program (to take advantage of freedom 3) without changing its source code. The FSF maintains a list of licences that comply with the Free Software Definition at http:/ /www.fsf.org/licenses/licenses.html; the list is revised from time to time, as new licenses are evaluated and added to the list.

2.2.2 *The Open Source Definition*

The Open Source Initiative maintains the Open Source Definition (OSD), available from http://www.opensource.org/docs/osd. The OSD has 10

clauses, covering redistribution, access to source code, the ability to make derived works, a requirement for unrestricted use by individuals, groups, or fields of endeavour, license distribution, license independence, and the need for the license to be technology neutral. The original open source definition was published in 1998, in response to the perception that the FSF's emphasis on freedom, with the implication that the source code was available free of charge, rather than on other benefits of this approach, was limiting the attraction of 'free software' to businesses. The people behind the Open Source Initiative, who included Todd Anderson, Chris Peterson (of the Foresight Institute), John "maddog" Hall and Larry Augustin (both of Linux International), Sam Ockman (of the Silicon Valley Linux User's Group), Michael Tiemann, and Eric Raymond, felt that in order for free software to become more widely used, it needed support from businesses. They hoped that by emphasising 'openness' they would avoid the confusion associated with the FSF's use of the word 'free' (Open Source Initiative, 2010a).

2.2.3 *'Free' vs 'Open': two different philosophies*

These two definitions represent different underlying philosophies. The OSI emphasises access to source code, with a focus on developers, while the FSF emphasises freedom and users' rights to change and redistribute the source code. However, in practice there is a significant overlap, since both the Free Software Definition and the Open Source Definition cover licensing and distributing software, and say nothing about the methods and processes used to develop and modify it. Both definitions require software developers to let users see the source code, change the software, and redistribute their changes.

In January 2010, the OSI listed 65 licences that comply with the OSD (Open Source Initiative, 2010b), while the FSF listed over 80 'free' licences (Free Software Foundation GNU Project, 2010). A majority of FLOSS licenses, including the GNU GPL and the GNU Lesser General Public License (LGPL), appear on both lists; each organisation evaluates new licenses only on request, which explains the difference in the number of licenses each lists. It also explains why some licenses are found on one list but not the other. The GPL is by far the most popular FLOSS licence, used by over 62% of the roughly 200,000 projects listed on SourceForge (the largest FLOSS project repository) in February 2010. At that time, the next most popular licences for projects listed on SourceForge are the LGPL, used by 10.7% projects, and the BSD (Berkeley Software Distribution) License, used by 7.1%.

The extent of this overlap shows that the free software and open source movements have similar criteria for approving licenses despite the differences in their underlying philosophies. This thesis therefore treats 'free software' and 'open source' as being roughly equivalent,

while acknowledging the philosophical differences between the two groups. This approach is consistent with Crowston et al.'s 2012 literature review, which also used the umbrella term 'FLOSS'. Another reason for considering the two as equivalent is to avoid the implicit bias introduced by the developer focus of 'open source' versus the user focus of 'free software'; by including both concepts, the researcher is indicating that the scope of the project includes both users and developers. Aksulu and Wade noted that "there is no single definitive definition of open source" (2010, p. 577); the research literature shows that some authors use the term 'open source', while others use 'FLOSS'. The definition of FLOSS developed for this thesis is "Software released under a free/libre and open source licence, which ensures that the source code is available to users (and potential users), and allows them to modify it to suit their needs, as well as redistribute it to others." The definition has been worded to incorporate the main clauses in both the Free Software Definition and the Open Source Definition, and does not introduce any other concepts.

2.3 RESEARCH INTO FLOSS PROJECTS AND PRACTICES

Academic researchers have taken considerable interest in the FLOSS phenomenon as its popularity has increased. Much of this interest has been based on perceived differences between FLOSS projects and conventional software development. To give one example, because many FLOSS projects cross organisational boundaries and have an informal organisational structure, there is a large body of research that looks at the development processes used in FLOSS projects, and how work is coordinated. Some examples include German (2003), who conducted a case study of development in the GNOME project, Scacchi (2004), who looked at development practices in the computer game community, Stark (2006), who looked at peer review practices of FLOSS developers, and Crowston, Li, Eseryel, and Howison (2007), who examined how tasks are assigned to individual developers.

Feller and Fitzgerald noted that FLOSS projects do not follow a standard development process (2002, p.23). However, they also say that FLOSS projects share the following characteristics:

- a distributed community of developers;
- parallel work on different components of the software; and
- incremental releases of the software (2002, p.24).

Fogel provided recommendations for 'best' practice in coordinating the work of these distributed developers, including a list of recommended communication channels such as email discussion lists, version control and bug tracking software, and synchronous chat software such Internet Relay Chat (2006).

Other research has focused on the demographic characteristics of FLOSS participants, with a recent example published in David and Shapiro's 2008 article reporting on the results of an extensive survey of FLOSS developers (n=1588). Their key findings were that their respondents were predominately male (98.4%), largely based in North America and Western Europe (80%), spent an average of 10 hours per week working on a FLOSS project, and had been involved with the project for approximately 5 years. They did not ask any questions about what roles respondents filled, instead using the term 'developers' to characterise all respondents.

Another common assumption is that people work on FLOSS projects as unpaid volunteers[2], which has resulted in a number of studies of developer motivations. These include Hars and Ou (2002), Roberts, Hann, and Slaughter (2006), Shah (2006), and David and Shapiro (2008). These typically find that FLOSS developers have multiple motivations for being involved in a project, ranging from a belief that FLOSS is the best way to develop software to wanting to learn new skills to enjoying the challenge of being involved in a large, distributed project. David and Shapiro (2008) found that ideology and belief in the value of FLOSS were more important to their respondents than economic incentives, and their respondents placed more importance on a project's technical challenge than its status or visibility, when they had a choice of projects in which to participate.

One aspect of a FLOSS project of particular relevance to the current research is its structure. The following section discusses the models of project structures that have been identified in the research literature to date.

2.3.1 *FLOSS project roles and activities*

Much of the research interest to date has had the objective of understanding FLOSS development practices, roles, and activities. This research distinguished between activities, roles, and practices. Practices were considered to be groups of individual activities, usually done in a way acceptable to the project's community. For example, a FLOSS project might have an agreed procedure for reporting bugs (a practice), which would involve several related activities (see if the problem is reproducible, check the bug repository to see if it has already been reported, etc.). A role involves carrying out a range of activities, which may relate to a number of different practices. Because of this, most of the existing

2 Though some people work on a FLOSS project on a voluntary (meaning unpaid) basis, most recent research identifying characteristics of developers has shown that many developers are paid for at least some of the time they spend working on the project. David, Waterman, and Arora (2003) found that 43.2% (of 1487 respondents) had earned money for their work on FLOSS projects, and that 40.7% (of 1488 respondents) worked on FLOSS projects during work time.

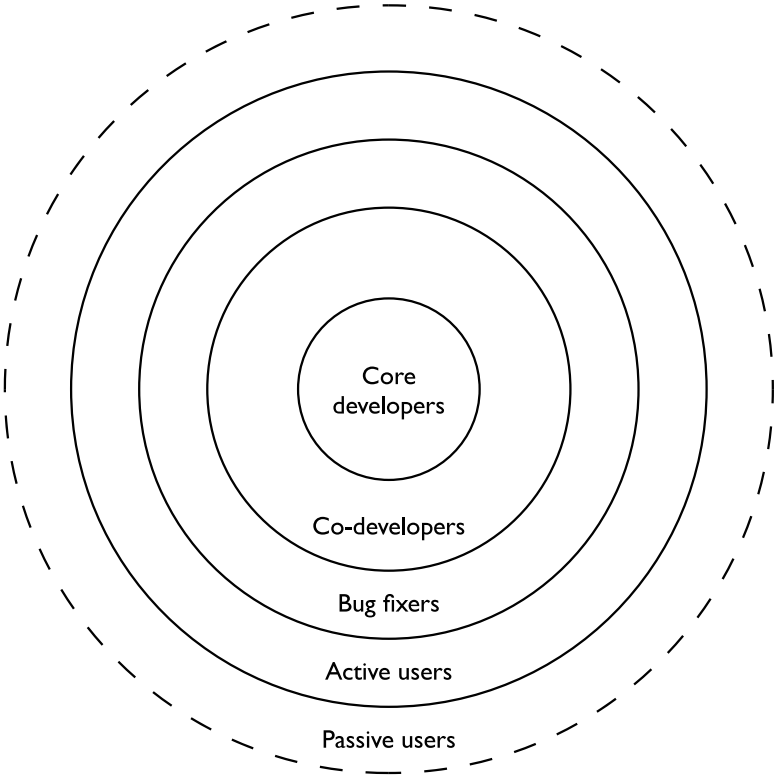
research into FLOSS project roles has focused on activities relating to the project's source code, such as writing new code or fixing bugs.

Gacek and Arief (2004) presented a complex model of user-developer roles and activities, distinguishing between passive users, who contribute nothing to the project, and active users, who report bugs and suggest new features. They also identified different categories of developers, suggesting that there are co-developers, who modify code and implement new features, and core developers, who make decisions about what code to accept. Their model did not include any community-oriented activities, such as writing documentation, or answering questions from users, though it indicated that people could change from being a passive user to an active one, or from being an active user to a developer, over time. One assumption underlying their model is that a FLOSS project participant's role is determined only by the activities they carry out with respect to the source code.

The second approach to representing FLOSS project roles is also developer-centric, placing the core developer at the centre of a hierarchy. Figure 1 on the next page shows a generic version of this style of diagram. The dotted outer line indicates that the boundary is unknown, since it is not usually possible to identify people who have installed FLOSS because there is no requirement for them to register with the project's community.

This approach was first presented by Ye and Kishida in 2003, and subsequently modified by Ye, Nakakoji, Yamamoto, and Kishida (2005) and Crowston, Wei, Li, and Howison (2006). These onion-style models all indicate the extent of participants' interaction with the code by placing different categories along the radius of a circle, with the distance from the centre representing the distance participants are from the code. This code-centric view of FLOSS participation was reinforced by Shah (2006), who considered contributing code and making decisions about what code to accept as the main forms of FLOSS participation available to community members. Such a code-centric approach risks missing or minimising other types of contributions to FLOSS projects, since users may determine the priorities for new developments, for example.

A broader approach was taken by Jensen and Scacchi (2007), who conducted a comparative case study of three large FLOSS projects: Mozilla, Apache, and NetBeans. Jensen and Scacchi identified a wider range of roles than earlier studies, and included community and governance roles in addition to roles that related to software development. Jensen and Scacchi found that each project they studied had unique roles, and also unique processes for moving between roles. However, their generic model of FLOSS project structure is still code-centric, with the community manager role the only one that focuses on the community rather than the code. Scozzi, Crowston, Eseryel, and Li (2008) identified both formal and informal roles in their study of the Lucene project, noting that the formal roles are committers and members of the Project Manage-



(based on Crowston, Wei, Li, and Howison 2004)

Figure 1: FLOSS project hierarchical structure

ment Committee. Ngamkajornwiwat, Zhang, Koru, Zhou, and Nolker (2008) discussed core and secondary or peripheral developers, again emphasising interactions with the code, rather than with the community.

There is little consensus about the terminology used for the roles in these models of FLOSS community roles and structure; Table 1 on the following page shows the roles identified in each of the models described above. The most common role is the 'passive user'; apart from that the terminology used to identify individual roles varies considerably, with different words used for what appear to be similar roles, such as 'Bug reporter' and 'Bug submitter'. The table also shows that the number of distinct roles in the models is increasing over time. This may be because the projects are becoming more structured as they evolve, or it may be because researchers are becoming more aware of subtle distinctions between roles as they increase their familiarity with FLOSS practices.

A small number of studies considered other types of participation in FLOSS projects. Lakhani and von Hippel (2003) investigated user support in the Apache project by studying postings to the comp.infosystems.www.servers.unix newsgroup between 1996 and 1999, supplemented with a survey of people who posted questions between 1 October 1999 and 15 February 2000. They found that people who answer questions also learn from other people's questions and answers, which was their main reason for reading postings to the newsgroup. Their most significant finding was that people usually provided answers they already knew, which took minimal effort (only 2% of their time on site). The analysis of newsgroup postings showed that 100 information providers answered around 50% of the questions, and a small number of people (fewer than 5) were particularly responsive. Because they limited their research to a single project, their findings are not generalisable to other projects, or to the wider FLOSS community. More recently, Zhao and Deek (2004) studied collaboration in FLOSS development, using a web-based survey. They selected 73 of the top 200 projects in SourceForge (based on project activity), and emailed 12 recently active users from each project. Their participation measure identified seven types of participation:

1. find bugs;
2. find usability problems;
3. suggest new features;
4. review and inspect source code;
5. submit source code;
6. offer project administration assistance; and
7. write documentation.

Table 1: FLOSS participant roles

ROLE	IDENTIFIED IN
Passive user	Ye and Kishida (2003); Gacek and Arief (2004); Ye et al. (2004); Crowston et al. (2006); Jensen and Scacchi (2007)
Peripheral developer	Ye and Kishida (2003); Ye et al. (2004); Ngamkajornwiwat et al. (2008)
Active developer	Ye and Kishida (2003); Ye et al. (2004)
Bug fixer	Ye and Kishida (2003); Ye et al. (2004)
Bug reporter	Ye and Kishida (2003); Ye et al. (2004)
Project leader	Ye and Kishida (2003); Ye et al. (2004)
Reader	Ye and Kishida (2003); Ye et al. (2004)
Stakeholder	Ye and Kishida (2003); Ye et al. (2004)
Core developer	Gacek and Arief (2004); Ngamkajornwiwat et al. (2008)
Committee member	Scozzi et al. (2008)
Committer	Scozzi et al. (2008)
Active user	Gacek and Arief (2004)
Co-developer	Gacek and Arief (2004)
Developer	Gacek and Arief (2004)
Non-developer	Gacek and Arief (2004)
Initiator	Crowston et al. (2006)
Release coordinator	Crowston et al. (2006)
Bug submitter	Jensen and Scacchi (2007)
Community manager	Jensen and Scacchi (2007)
Feature requester	Jensen and Scacchi (2007)
Module developer	Jensen and Scacchi (2007)
Module lead	Jensen and Scacchi (2007)
Observer	Jensen and Scacchi (2007)
Project manager	Jensen and Scacchi (2007)
Quality assurance lead	Jensen and Scacchi (2007)
Test case contributor	Jensen and Scacchi (2007)
Veteran tester	Jensen and Scacchi (2007)

Their list did not include responding to other users' questions as a participation option, but instead included this in a section about use of electronic communication media. Overall, their approach lacked a theoretical foundation, and they presented their results as a simple descriptive survey, with no attempt to examine relationships between the variables in their study. Finding bugs was the most common type of participation, followed by suggesting new features and finding usability problems.

In a more recent study, Studer (2007) asked participants in the KDE project to identify how often they carried out each of the following activities: code, coordination, discussion about future developments, art[3], bug management, help, documentation, translation, packaging, web, and bug reports. While this list is slightly more extensive than the one used by Zhao and Deek, adding the support activities 'translation' and 'help', plus the marketing activity 'web' and the content category 'art', it is still focused on the types of activities developers are likely to be involved with. Studer also asked survey respondents to indicate how much prestige they assigned to each type of activity, using a 10-point scale where 1 represented 'No prestige' and 10 was 'Very prestigious'. Code was assigned the highest rating (a mean of 8.46), followed by coordination (7.44). Five activities (bug management, help, documentation, translation, and packaging) were assigned mid-range scores with means between 5.0 and 5.4. Studer described help and documentation as 'non-productive' activities since they did not result in any visible changes to the underlying source code, implying that this result was unexpected. However, by giving these community-oriented activities prestige ratings that were similar to the code-oriented activities of bug management, translation, and packaging, survey respondents showed that they recognised the importance of supporting the community.

Project governance is another aspect of FLOSS development that has received interest in the research literature, though this has not been well-integrated with other research on FLOSS project roles. Based on her qualitative review of empirical research on FLOSS project governance, Markus (2007) suggested that governance roles are complex, involving the establishment of six types of rules: ownership, chartering, community, software development process, conflict, and rules for using information and tools.

The preceding discussion has shown that none of the existing models of FLOSS project roles and lists of typical activities adequately reflects community governance and support activities. Answering the first sub-question for this research project, *What types of contributions do participants make to free/libre and open source software projects?* will extend these models to include other types of activity, in order to give a

3 His term, not defined further. In the GNOME project, the term *art* refers to contributions of graphics and images for use in GNOME software, including icons and desktop themes.

more balanced perspective on the types of activities that occur in FLOSS projects.

2.3.2 *FLOSS community practices*

There is a growing body of literature about FLOSS community practices, particularly the way the distributed teams work effectively together. Gallivan (2001) found that social control was an important mechanism to help members of a distributed FLOSS community work together. Von Krogh, Spaeth, and Lakhani (2003) undertook a detailed case study of the Freenet project using grounded theory techniques to determine how people joined the developer group for the project and how they initially contributed code. The Freenet project began in 2000, originating from a computer science M.Sc. thesis on the theoretical principles of a peer-to-peer distributed file system. At the time of the 2003 study, there were 30 people with commit access to the project's CVS repository, meaning that they could contribute source code without needing a third party to approve it. Von Krogh, Spaeth and Lakhani's results showed that the Freenet community had an implicit 'joining script' (i.e. a practice) that newcomers were expected to follow before being granted commit access to the code repository. The most important activities included in the joining script were participating in technical discussions and offering new code, usually to fix a bug in the current version of the software. This suggests that existing developers with commit access acted as gatekeepers (an impicit role), and that enforcing this type of joining script is an implicit type of quality control mechanism. It also suggests that some contributions will be rejected, and that FLOSS projects may vary in the extent to which they welcome new contributors.

This suggestion is confirmed by research by Ye et al., who identified two dimensions of openness that vary between projects: *product openness* and *process openness* (2005, p.76–77), based on their study of four projects within a single company: GNU Wingnut, the Linux Support project, SRA-PostgreSQL project, and the Jun project. GNU Wingnut was a project to help people port GNU software, such as GCC (GNU Compiler Collection) and GNU Emacs (a text editor) onto supercomputers. The Linux Support project provided support for customers using the Linux operating system, while the SRA-PostgreSQL project added support for the Japanese language to the PostgreSQL database software. The Jun project developed a Smalltalk and Java library for 3-dimensional objects and multimedia data. Yet et al.'s definition of the concept 'product openness' related to the release practices of the project's development community; 'open release' meant that only formal releases were available to the user community, while 'open development' meant that interim releases were available as well. One consequence of the open development option is that any interested member of the

community can be involved in testing early releases of the software, while an open release model restricts this access to a closed group of developers.

'Process openness' was defined as the extent to which members of the project's wider community were able to participate in decisions about the software's development path. Ye et al. (2005) identified three possible values for process openness: closed, transparent, and open. With a closed process, only the 'inner circle' of core developers determined the development priorities, while with a transparent one, the discussions were accessible to all community members, but decisions were made by the core group. An open process allowed any community member to participate fully in decisions about future development. Coleman and Hill (2005) described the way the Debian process openness changed as the project's community grew in size. Initially the process was relatively open, but as the number of developers increased (from ~100 to ~1,000), the developer community recognised a need for more formal procedures. This resulted in the Debian New Maintainer process, which was a more formal approach to determining who could be involved in discussions about the Debian project's future directions, and moved the project from 'open' to 'transparent'. This example shows that the concept of process openness applies to different FLOSS projects than the ones originally studied by Ye et al. (2005), and suggests that it may be an important characteristic to include in the current research.

2.4 ISSUES RELATED TO STUDYING FLOSS PROJECTS AND COMMUNITIES

Most FLOSS projects provide a wealth of freely available project information for prospective users, and this is one characteristic that makes them attractive to researchers. These resources usually include the source code itself, which typically includes a credits file that shows which developer contributed specific code and when it was contributed, plus mailing list archives, bug databases, and other project-related documentation. In addition, participants in these projects have a choice of communication channels, which typically include email discussion lists, IRC[4], or online forums. The archives of the email discussion lists and online forums, plus the logs of the IRC conversations, are often made available to people interested in learning about the project's activities.

These data are particularly useful for writing an in-depth case study of an individual project. Von Krogh, Spaeth, and Lakhani's investigation (2003) of the way new developers joined the Freenet project is one example of research that has taken advantage of access to this type of data. Crowston, Wei, Howison, and Wiggins's 2010 literature review found that 42% of the empirical papers they identified used data about a single

4 IRC is an initialism for Internet Relay Chat, a form of instant messaging used for synchronous, text-based communication

project. The most common projects studied were Linux, Apache, Mozilla, and Gnome, all of which are widely used and have large communities of developers. While such studies of a single, high-profile project can provide insights into good practice or show how that particular developer community interacts, they are unlikely to provide findings that are generalisable to other projects.

For FLOSS research to be more generalisable, data need to relate to more than one project. Identifying suitable samples is challenging, since there is no single authoritative source of high-quality, comparative information on FLOSS projects. SourceForge (http://sourceforge.net/) is the largest world-wide repository of FLOSS projects, listing over 200,000 projects in February 2009 (source: SourceForge.net About page http://sourceforge.net/about, accessed 20 December 2009). While this number is impressive, studies of SourceForge project statistics have consistently shown that only a small number of these projects were being downloaded, and a much larger number were relatively static, with low numbers of downloads and/or contributors. In 2002, Hunt and Johnson showed that the distribution pattern of project downloads was heavily skewed, following a power law or Pareto pattern where a small number of projects had a very large number of downloads, and a large number of projects had a small number of downloads. Ohira, Ohsugi, Ohoka, and Matsumoto (2005) found that 66.7% (50,665 of 90,902) of projects harvested from SourceForge had a single registered developer. The situation has been further complicated by the emergence of other source code repositories, such as CodePlex (sponsored by Microsoft) and Google Code, which means that data about FLOSS projects are now spread across multiple sources.

One possible solution to this is the FLOSSMole project, described by Howison, Conklin and Crowston (2006). It was established to provide a clearinghouse for FLOSS data by harvesting data from multiple source code repositories. In August 2010 it included data for over 500,000 projects from eight different repositories (SourceForge, FreshMeat, RubyForge, ObjectWeb, the Free Software Foundation, SourceKibitzer, Savannah, and Github).

While FLOSSMole does provide a useful resource for comparing data about large numbers of FLOSS projects, it has limitations. Some FLOSS projects use external resources to supplement or complement repository hosting, or are available from a site provided by the main developer or other interested party, rather than a shared repository. Stürmer (2005) studied eight FLOSS projects, five of which were popular web content management systems, while two were web application frameworks and one was a WYSIWYG[5] browser editor. While most of these projects had begun by using SourceForge as their code repository, a majority had migrated to other collaboration platforms as they became more

5 WYSIWYG is an acronym for 'what you see is what you get', an approach used in designing editors where the screen display is similar to the final output, such as a printed document.

popular (Stürmer 2005: 72). One project, Magnolia, used its own collaboration platform for all development activities, and used SourceForge as a download site only. In another example, the PmWiki project is listed on SourceForge, but the majority of the project's resources, including mailing list archives, are hosted on the project's main web site, http://www.pmwiki.org/. In December 2009, the version of PmWiki available on SourceForge was 2.1.27 (dated 2006-1-12), while the version available from the project's main site was 2.2.8 (dated 2009-12-07). It appears that PmWiki uses SourceForge as a historical 'back-up' repository, but this is not made clear in its SourceForge description. The FLOSSMole project itself is hosted on Google Code, but uses email discussion lists provided through SourceForge, showing that information about projects is sometimes split between different repositories.

Selecting FLOSS projects for a research project is therefore not a straightforward matter of choosing a random sample from the FLOSSMole repository, or any of the individual repositories. The large proportion of inactive projects means that a random sample of projects is likely to have a large number of inactive projects, while selecting active projects may bias the results towards particular types of communities and projects. In addition, it will omit projects that do not use one of the harvested code repositories as their primary hosting site. This suggests that researchers need to consider other methods to identify suitable samples of FLOSS projects.

There is little agreement in the FLOSS research literature about the best way of doing this. Crowston et al. (2010) found that approximately 18% of their sample studied fewer than 10 projects, and the next largest group (16%) used repository data to study thousands of projects. When researchers have used a sample of projects as the source of their data, they seldom spell out their selection criteria. A typical example is Singh, Twidale, and Nicols (2009), who studied email discussion threads from NVU, Opera, Filezilla, phpMyAdmin, phpBB, Dropline, Mozilla Firefox, and Moodle, but give no reasons to explain this choice of projects. Others use a small number of criteria to choose their sample, such as Krishnamurthy (2002), who restricted his sample to the 100 most active mature projects on SourceForge. This approach is essentially a convenience sample, since only two criteria were used to select projects (maturity and activity), and only one value was used for maturity. These approaches limit the generalisability of the research results, since the relationship of the sample to the population is either unknown, in the case of Singh, Twidale, and Nichols (2009) or skewed, in the case of Krishnamurthy (2002).

Crowston, Li, Eseryel, and Howison (2007) used purposive sampling, identifying project characteristics that were particularly relevant to their research question, which looked at the process of assigning work to specific developers. Their criteria were that projects had more than seven developers, made their email archives publicly available, had an

active community, had a high number of downloads, and were releasing new software versions. Since their goal was theory development rather than theory testing, they used a case study approach based on three projects chosen from different application areas, rather than choosing a representative sample (p.567).

An alternative approach that has been used by other researchers is a form of cluster sampling. Researchers who take this approach identify one or more application areas, and compare projects across the areas. Scacchi (2002) used it in his study of user requirements in FLOSS projects, choosing FLOSS projects from four application areas: networked computer games, Web infrastructure, X-ray astronomy and deep space imaging, and tools to support academic software development. Spaeth, Stuermer, and von Krogh (2007) suggested using the Debian project and its packages as a source of projects that are known to be active. However, since Debian is primarily an operating system, and does not focus on user-oriented application packages, this specific approach was not felt to be suitable for the current research. However, the approach of finding a suitable cluster of projects compensates for the power law distribution found in source code repositories, since it lowers the probability of selecting a large number of relatively inactive projects. It also allows for theories to be tested across multiple projects. It works best when the application area has a diverse population of FLOSS projects.

There are also issues in identifying individuals who use specific FLOSS packages. Most projects do not have a complete list of their users, since registration is always voluntary and participation in the community is optional. This means that the total population cannot be determined, and it is not possible to develop a sampling frame for research into participants in FLOSS projects. Because of this, a common approach to studying FLOSS developers from a range of projects has been to use a web-based survey, with an open invitation to complete it sent to email lists and posted on the Web. This technique was used in several large-scale surveys of developers, including Ghosh et al. (2002), and David et al. (2008). It is most suited when responses from individual participants are needed because the data in the repositories are not appropriate to answer the research question.

2.5 UNDERSTANDING SATISFACTION

Although satisfaction is a concept that is widely used in many different disciplines, satisfaction research does not generally use a single, agreed definition of satisfaction (Oliver 2010, p.6). McNamara and Kirakowski noted that the definition of satisfaction used in satisfaction research has changed over time, moving from a cognitive assessment of quality to an emotional response, and then back to a more cognitive approach

(2011, p.376). They also noted that satisfaction is "a summary evaluation" (p.377), and suggested that it may include both cognitive and/or affective (i.e. emotional) aspects.

The definition of 'satisfaction' used in this thesis is "The satisfying of a need or desire as it affects or motivates behaviour" where 'satisfying' means "meeting or fulfilling the wish or desire or expectation of; to be accepted by (a person, his taste, judgement, etc.) as all that could be reasonably desired" ('satisfaction' and 'satisfy', Oxford English Dictionary Online, 2010). This definition includes the main characteristics that people consider when they assess their level of satisfaction with a product or service: that is, they judge the extent to which the product or service meets their expectations. This means that satisfaction involves individual judgement, and is not something that can be observed by third parties or measured directly. It also implies that judgements of satisfaction are based on previous experience, and that in order to assess their satisfaction with a product or service, people need to have used the product or experienced the service; satisfaction cannot be judged in a vacuum.

In addition, satisfaction is a generic concept that can be applied to many things, and it is therefore important to understand the context in which it is being measured in order to choose an appropriate measurement tool. Oliver notes that while it is easy for people to assess their level of satisfaction with a product or service, it is much harder for them to define what this actually means (2010, p.7).

Oliver identifies other key aspects of satisfaction that researchers need to acknowledge (2010, p.7). In particular, satisfaction can be measured on many levels, such as satisfaction with events associated with the consumption of a product or experiencing a service, satisfaction with the final outcome (considered in isolation), or satisfaction with the overall experience of consuming the product or experiencing the service. In addition, in some cases, satisfaction is a result of a series of events that occur over time, and what is measured is the level of satisfaction with these blended experiences, rather than the satisfaction with each individual experience. This implies that there will be various ways of measuring satisfaction, depending on the type of product or service of interest. This thesis is concerned with people's overall satisfaction with their involvement with a FLOSS project, rather than their satisfaction with a particular aspect of the project, or their satisfaction with a single interaction with the project's community.

2.6 MEASURING SATISFACTION

Having established that satisfaction is a complex concept, how is it typically measured? Since satisfaction cannot be seen, or inferred from behaviour, it is usually measured by asking people to complete a survey

containing questions about the product or service being evaluated. This section considers the most common approaches to measuring satisfaction.

The standard approach to measuring satisfaction involves using a quantitative survey that identifies a list of dimensions or components that are important in the context of the type of product or service, and then asks people to rate their satisfaction with each of them. This generally uses a Likert-style scale with five or seven choices, ranging from 'not at all satisfied' to 'very satisfied'; Oliver (2010, p.49) notes that this approach to wording the scale is used in order to avoid a positivity bias in the results, because some people find it difficult to choose negative responses such as 'very dissatisfied'.

In addition to indicating their satisfaction with aspects of a product or service, early satisfaction research also asked respondents to indicate the importance of each dimension, again using a Likert scale, in order to calculate a weighted satisfaction index, as $\sum_{d=1}^{n} i_d s_d$, where d is the number of dimensions, i is the dimension's importance, and s the level of satisfaction with the dimension. While this approach seems to have an intuitive face validity since it places the most emphasis on the dimensions that are most important to respondents, experience in using it has been mixed (Oliver 2010, p.50). This appeared to be because survey respondents interpreted the meaning of 'importance' in different ways. They may not all have reported their perceptions of the relative importance of each dimension in contributing to their satisfaction, but in some cases may have considered the dimension's importance in their decision to purchase a product or use a service. This illustrates the distinction between what Oliver terms 'choice criteria' and 'satisfaction drivers' (2010, p.37). Choice criteria are aspects of a product or a service that influence someone's initial decision to purchase the product or use the service, while satisfaction drivers are dimensions of the product or service that people experience after they have made the initial decision. Some features influence both choice and satisfaction, and Oliver terms these 'dual influence features'. Cost is one example of a pure choice criterion, while the politeness of a customer service representative is a pure satisfaction driver for a specific transaction, since it cannot be determined in advance. One example of a dual influence feature is airline seating. The plane's seat layout might influence someone's decision to book a specific flight, while their specific seat and its proximity to the seat in front and the aisle is likely to affect their satisfaction with the flight itself.

In addition, some studies have found that there may be an inverse relationship between the rating people give importance and their level of satisfaction with the feature; in particular, respondents often give features with which they are not at all satisfied more importance than features with which they are completely satisfied. If their experience

subsequently changes to make them more satisfied with the feature, they lower its importance if asked to reassess their satisfaction (Oliver 2010, p.51). For these reasons, most recent research into satisfaction does not use this approach (Oliver 2010, p.52), but instead calculates overall satisfaction as the sum of the individual satisfaction ratings for each dimension.

An alternative way of measuring satisfaction uses a disconfirmation approach (Applegate 1997, p.209). The term 'disconfirmation' refers to the gap between the expected level of performance, and the actual/perceived level; a disconfirmation-based survey typically asks respondents to indicate their expected level of performance, and the level they received. The difference between the two measures is calculated, and used to indicate the gaps between expected and actual (or perceived) performance. The assumption is that the smaller the gap, the greater the 'satisfaction'. Oliver presented a detailed discussion of the advantages and disadvantages of this approach (2010, p.96–127); one of his key points is that in some cases, there can be no gap between expected and perceived performance because both are low, but respondents can still be dissatisfied with an aspect of a service or product. He also suggested that this approach is best used when a researcher is trying to understand the relationship between people's expectations and their assessment of their satisfaction, as opposed to understanding the dimensions that determine someone's overall satisfaction.

Hayes provided guidance on determining the underlying dimensions needed to measure satisfaction with a particular type of product or service, (2008, p.12–23). He noted that while a small number of generic dimensions can apply to different products and services, such as availability and convenience, it is not realistic or meaningful to develop standard dimensions that can be used across a broad range of products and services. This means that it is necessary to identify the dimensions that are most important for the specific type of product or service being evaluated.

Hayes recommended two techniques for identifying the dimensions that are important for measuring satisfaction:

1. using industry-specific literature (2008, p.12); and
2. asking individual customers, consumers, or users to describe specific positive or negative experiences, and using content analysis to reduce this list to identifiable dimensions (2008, p.20–23).

User satisfaction with software and information systems has been studied extensively, and there is already a body of literature that identifies dimensions that have been found to be important in determining satisfaction. This literature is discussed in the following section.

2.7 SATISFACTION WITH INFORMATION SYSTEMS AND SOFTWARE

User satisfaction has been one of the most popular and enduring topics in the information systems literature, in part because it has been used as a surrogate measure of information system (IS) success for many years. In a review of 59 articles about user satisfaction with information systems, Zviran and Erlich (2003) said that satisfaction is "the most prevalent measure of IS success due to its applicability and ease of use" (p.83).

2.7.1 *Classifying satisfaction with information systems and software*

Seddon, Staples, Patnayakuni and Bowtell (1999) presented a two-dimensional framework for classifying IS studies of satisfaction, based on the nature of the application/system being assessed, and the type of respondent whose perspective is sought. The six application/system categories were: an aspect of IT use; a single IT application; a type of IT application; all IT applications used by an organisation; an aspect of system development methodology; and an IT function in an organisation. The five types of respondents were: independent observers; individuals; groups; managers; or countries. This results in a total of 30 unique combinations, and Seddon et al. suggested that each of these might require a different instrument to measure success/satisfaction. Their analysis of 186 studies showed that individual (70) and manager (50) perspectives for a single type of IT application were the most frequent combinations studied. Their research also showed that researchers used a range of measures to assess satisfaction, many of which built on earlier measures. Seddon et al. argued that this is a strength of IS evaluation research, since it allowed researchers to focus on what is most relevant in a study's context, rather than using inappropriate general measures. The current study focuses on individuals' satisfaction with their experience of a FLOSS project; the best fit with Seddon et al.'s categories is an aspect of system development methodology, since FLOSS projects can involve any type of IT application.

Examining research into user satisfaction with information systems/software showed there were some inconsistencies with terminology. Some studies, such as Guimaraes, Staples, and McKeen (2003), used the term satisfaction in their measures, but used the term 'system quality' or 'system success' for the constructs in their models. Other researchers asked questions about effectiveness (or quality), but then named the measure satisfaction. Literature covering both approaches has been included in this review.

2.7.2 *Measuring satisfaction with information systems and software*

In 2003, Zirvan and Erlich reviewed the literature on user satisfaction, identifying six major user satisfaction scales and 60 individual studies that measured user satisfaction. The sections below discuss these six satisfaction scales, summarising their initial development, validation, and subsequent use by other researchers. One point to keep in mind is that these scales were all developed for a conventional software/information systems context, and consider only the user perspective on satisfaction.

Bailey and Pearson (1983)

Bailey and Pearson's widely cited 1983 paper is generally regarded as the first significant achievement in developing a generic instrument for measuring user satisfaction with information systems, particularly because it considered more than just the output of the computer system. It was not the first study to measure how satisfied users were with an information system—earlier research included Gallagher (1974), whose study focused on the value of the reports produced by a management information system, and Jenkins and Ricketts (1979, cited in Zviran and Erlich, 2003), who studied user satisfaction with the output of decision support systems. Bailey and Pearson used a sequential mixed methods approach to develop their questionnaire, which consisted of 39 statements about characteristics of computer-based information systems or services, or staff of the information systems department, with six semantic differential scales[6] for each statement. Four of these were pairs of adjectives reflecting different feelings or judgements about the individual characteristic; in addition, each included a satisfactory/unsatisfactory scale and an important/unimportant one. Bailey and Pearson used data from 29 respondents to test the psychometric characteristics of the instrument, and found that it had high reliability and acceptable content, predictive, and construct validity.

The instrument Bailey and Pearson developed is very much a product of its time. First, it calculated overall satisfaction using the weighted importance*satisfaction approach that is typical of early research into consumer satisfaction. The difficulties with this approach are evident in the results of their first, qualitative stage, which involved interviews with 32 middle management customers of their organisation's information systems department. They note that "causes of satisfaction vary from user to user" (p.533), and that all but nine of the 38 initial dimensions were ranked in the top five by at least one interviewee. This reinforces the idea that satisfaction is a complex construct, and that individual

6 A semantic differential scale anchors the ends of the scale with two words having opposite meanings, such as 'bad' and 'good', in contrast to a conventional Likert-style scale, which typically asks people to indicate their level of agreement with a statement using choices that include 'Strongly disagree', 'Neutral', and 'Strongly agree'.

circumstances play a large part in determining someone's satisfaction with a product or service.

Bailey and Pearson's use of the phrase 'computer user satisfaction' is somewhat misleading for the reader, because their instrument covered not just computer use, but also interactions with the organisation's information systems department. They made the assumption that all of the computer use was mediated in some way by this unit in the respondent's organisation. This may have been true in the 1980s, when Bailey and Pearson carried out their initial research, but it is not necessarily the case in the 21st century, which has been termed the age of 'ubiquitous' computing (Elliott and Kraemer 2007, p.525). Some of the questions in Bailey and Pearson's instrument, such as those about production schedules, volume of output, and error recovery (defined as "the methods and policies governing correction and rerun of system outputs that are incorrect", p.542), reflect the predominant batch processing model used in the early 1980s, and would not be appropriate today.

Bailey and Pearson's instrument has been widely used by other researchers, of whom only a small number have used the full 39 items unchanged. In most cases, they have chosen a subset of items that are particularly relevant to their context and modified them so that they are more meaningful to the sample group. Factor analyses of the underlying dimensions of the Bailey and Pearson instrument, including variations, have had varying results. Raymond (1985, 1987) found four dimensions from his 20-item version; these were output quality, user–system relationship, support, and attitude to EDP staff. In contrast, Tan and Lo (1990) identified eight factors from the full 39-item scale: information quality, user attitude, user knowledge and involvement, security, system integration, user interface, system utility, and management support. All of this research occurred in the late 1980s and early to mid-1990s, and use of the Bailey and Pearson instrument has declined since then, though it is still cited frequently as an early model of an instrument to measure user satisfaction.

Ives, Olson, and Baroudi (1983)

Ives, Olson, and Baroudi reviewed four previously proposed measures of what they termed 'user information satisfaction', noting that they could be grouped into two broad categories. The first had a narrow focus on the products or outputs of the information system, with questions about the accuracy and relevance of the information and its presentation, while the second took a broader perspective, asking about organisational support and interactions with information technology department staff. They placed Bailey and Pearson's 1983 survey in the second category. Ives, Olson, and Baroudi then conducted a two-stage, large-scale empirical study to validate the 39 items used in Bailey and Pearson's scale. The first stage used the original 39-item questionnaire, and the second a shorter

questionnaire with four questions about the quality of service provided by the data processing group. Ives, Olson, and Baroudi received 280 responses for the first survey, and 200 for the second one. The results from the second survey were treated as an independent measure, and used to assess the predictive validity of the longer Bailey and Pearson instrument. This showed a 0.55 correlation between the two surveys (significant at the 0.001 level).

Ives, Olson, and Baroudi then conducted a detailed analysis of each individual item's reliability and correlations with the short survey in order to improve the quality of the Bailey and Pearson instrument and reduce the amount of time required to complete it. Their resulting 'short-form' instrument retained 13 of the original 39 items, with two semantic differential scales per item, and covered:

- users' relationship with IT staff,
- IT staff attitudes,
- IT responsiveness to change requests,
- time required to develop new systems,
- user training,
- user understanding of system,
- user participation,
- accuracy of output,
- precision of output,
- relevance of output,
- completeness of output,
- reliability of output, and
- communication with IT staff.

Ives, Olson, and Baroudi found that this shorter instrument had a 0.90 correlation (significant at the 0.001 level) with Bailey and Pearson's longer questionnaire. It was the most widely used instrument in the literature Zirvan and Erlich (2003) reviewed. Baroudi and Orlikowski (1988) conducted a survey to test the psychometric properties of this instrument, and found that it had acceptable construct validity. A factor analysis of their data, which were drawn from 358 employees of 26 different companies, identified three underlying dimensions:

- EDP staff and services,
- information product, and
- respondent knowledge and involvement.

In a subsequent confirmatory factor analysis using data from 224 respondents, Doll, Raghunathan, Lim, and Gupta (1995) found that four dimensions, which they termed EDP staff, EDP services, information product, and knowledge and involvement, provided the best fit with their data. This was cross-correlated with two further data sets, which also confirmed the four-dimension structure.

Following on from Baroudi and Orlikowski (1988), other researchers have tested the psychometric properties of the scale in different contexts. Galetta and Lederer (1989) is the most widely cited of these. They tested the reliability of the scale by having their respondents complete the instrument twice, the second time after they had been presented with information about reasons for general system failures and successes. They found that some responses changed, even though their subjects' experience with the system being evaluated remained the same. As a result Galetta and Lederer concluded that there were issues with the instrument's reliability, but another interpretation is that the presence of additional information, even though it was about other systems, affected the way in which their subjects judged their satisfaction. This could be expected given the nature of satisfaction, discussed in Section 2.5 on page 28.

Other researchers have treated the scale as a standard way of measuring user satisfaction, using the scale in its original form, or with minor variations. Some examples are Barki and Huff (1985, 1990), Nelson and Cheney (1987), Tait and Vessey (1988), Igbaria and Nachman (1990), Joshi (1990), Hawk and Dos Santos (1991), Kettinger and Lee (1994), McKeen and Guimaraes (1997), and Sengupta and Zviran (1997). McKeen, Guimaraes, and Wetherbe (1994) used 10 questions from the Ives, Olson and Baroudi instrument, omitting three questions that overlapped with other constructs they were testing (relationship with IS staff, participation, and communication with IS staff). Yoon, Guimaraes, and O'Neal (1995) used only the nine items associated with output quality and user–system relationship in their research exploring factors associated with expert systems success. This version was subsequently used by Guimaraes, Yoon, and Clevenson (2001) and Guimaraes, Staples, and McKeen (2003). Guimaraes and Igbaria (1997) chose ten items that focused on output quality. Palvia (2000–2001) used the full 13 items with a single semantic differential scale to measure user satisfaction with synchronous and asynchronous training. Whitten (2004) showed that similar reliability and factors could be achieved using a single scale per item, which reduced the size of the instrument to 13 responses instead of 26.

The flexibility of Ives, Olson, and Baroudi's approach to measuring user satisfaction is shown by its ongoing adoption by other researchers. However, it is based on an assumption that users are supported by information systems staff, which may not apply in a FLOSS context.

Miller and Doyle (1987)

The next significant approach to measuring satisfaction was developed by Miller and Doyle (1987). They were interested in evaluating the effectiveness of the information systems function in financial services sector organisations, and based their instrument on two previous studies: Bailey and Pearson (1983) and Alloway and Quillard (1981). Their 38-item questionnaire used 24 items from Bailey and Pearson, 12 from Alloway and Quillard, plus two new ones. Though they did not provide a detailed rationale for their choice of items, they said that they wanted to assess general information systems effectiveness, using satisfaction as a surrogate. In designing their survey, they followed the then current approach of asking respondents to judge the importance and the performance of each item to their organisation, where performance was assumed to be synonymous with satisfaction. Their results, based on responses from 276 managers in companies in the financial services sector, found that the importance responses showed few patterns, and did not map into clearly defined factors. Unlike Bailey and Pearson (1983), Miller and Doyle did not use the importance ratings to calculate a weighted overall satisfaction score; instead they calculated total satisfaction as the arithmetic mean of the responses.

In contrast, the performance (i.e., satisfaction) responses had good internal reliability (a Cronbach's alpha of 0.94), with a factor analysis identifying seven dimensions:

- system characteristics (including characteristics of output such as completeness, accuracy, relevance, and currency),
- strategic management,
- user involvement,
- IS staff responsiveness,
- end user control,
- IS staff quality, and
- service reliability.

These suggest that overall satisfaction is influenced by a combination of characteristics of the system, the organisational context, and the individual user. This is consistent with the idea drawn from the more generic customer satisfaction literature that satisfaction is more complex than it appears on the surface, and that in order to fully understand the factors that influence it, researchers need to measure a range of user, contextual, and system/software attributes.

Guimaraes and Gupta (1988)

The goals of the three previous studies were to develop reliable instruments for measuring user satisfaction with information systems in

organisations. Guimaraes and Gupta (1988) had a more specific aim: their study was intended to develop a reliable instrument to measure senior management satisfaction with the information systems department in their organisation. They modelled their approach on Bailey and Pearson (1983), but developed a new list of items chosen to reflect the specific concerns of an organisation's senior managers when assessing the performance of a unit in their organisation. They did not use any items from existing measurement scales.

The Guimaraes and Gupta instrument had 19 items grouped into five factors: relationships with other departments, relationships with management, technical image, service quality, and cost/benefit to the organisation. The data they gathered from 109 respondents representing 40 companies showed that most questions had adequate internal reliability (all but two constructs had Cronbach's alpha values greater than 0.80) and content validity.

The general nature of this study means their approach, at least at the factor level, appears less dated than the previous three, and their questions could be used with only minor changes today. Their findings reinforce the idea that satisfaction is context-dependent, and a comparison of their main dimensions with those of Miller and Doyle (1987) shows little overlap. The concepts which are closest in meaning are their dimensions of service quality, which is similar to what Miller and Doyle (1987) termed 'service reliability', and 'relationship with other departments', which is similar to Miller and Doyle's 'IS staff responsiveness'. However, the wording of the questions that relate to these four concepts is very different in the two survey instruments, which suggests that the similarity is coincidental, and that a different underlying concept is being measured in each case.

The significance of Guimaraes and Gupta's work is that it shows the importance of considering the nature of the population being surveyed when developing an instrument to measure their satisfaction with an aspect of an information system. The instruments developed by Bailey and Pearson (1983), Ives, Olson and Baroudi (1983), and Miller and Doyle (1987) were all intended to gather information from people who were the intended *users* of the system, while Guimaraes and Gupta were interested in senior management perceptions of the effectiveness of the organisation's information systems department. It seems obvious that different measurement techniques will be needed for senior managers, because aspects that are relevant to users may not be equally relevant to senior managers. In addition, senior managers are likely to give greater weight to aspects that are not usually considered by users. For example, Guimaraes and Gupta asked about company expectations of the MIS department and company control over MIS activities, neither of which is likely to be considered by users of information systems who are not involved in company-wide decision-making.

Doll and Torkzadeh (1988)

The previous examples of instruments to measure user satisfaction were developed for three distinct situations: measuring people's satisfaction with the services and output they receive from their IT department (Bailey and Pearson 1983; Ives, Olson, and Baroudi 1983), measuring people's satisfaction with services and output they receive from their IT department in a specific industry (Miller and Doyle 1987), and measuring senior management's satisfaction with the performance of their IT department (Guimaraes and Gupta 1988). None of these instruments is easily generalisable to end users who use computers and software relatively independently. Doll and Torkzadeh (1988) developed their End-User Computing Satisfaction (EUCS) instrument specifically for users who have a degree of control over their use of computers and software, and do not rely on an IT department to mediate it. Their final instrument had 12 questions, covering five dimensions: content (four items), accuracy (two items), format (two items), ease of use (two items), and timeliness (two items). Doll and Torkzadeh validated their instrument with 618 responses representing 250 different applications. This showed that the latent variables had Cronbach's alpha values of .78 or higher for each dimension, and a factor analysis showed appropriate loadings on the five dimensions. A subsequent confirmatory factor analysis by Doll, Xia, and Torkzadeh (1994), using survey data gathered from 409 users of 139 different applications, supported the five dimensions, as did a survey of 204 student users of CASE[7] tools by Kim and McHaney (2000).

The EUCS instrument has been used in many subsequent studies, often with slight modifications. Simon, Grover, Teng, and Whitcomb (1996) used the 12-item EUCS with a 5-step Likert-scale ranging from 'disagree' to 'agree' in a study of the impact of training methods and cognitive ability on satisfaction, comprehension, and skill transfer. They found that the scale had a Cronbach's alpha value of 0.98, and did not do any further factor analysis on their results. Though such a high value raises questions about whether there was duplication between individual items on the scale, Simon et al. (1996) did not do any further analysis of individual items. Aladwani (2003) also used the 12-item scale in a study of undergraduate student satisfaction with email technology, testing two versions of the wording: one was in the first person "the e-mail technology provides me with the information I need in time" and one in the third "the e-mail technology provides one with the information (s)he needs in time". Aladwani found no significant difference in responses for the two versions. Rainer and Harrison (1996) used a slightly modified version of the Doll and Torkzadeh instrument to measure overall user satisfaction with the computer-based systems they used, and included

7 CASE is an initialism for Computer-Aided Software Engineering, which typically involves the use of software to assist with the development of a computer system, following a structured methodology.

a wider range of end-users in their sample than the original Doll and Torkzadeh study, which restricted its sample to major users. They found that the five dimensions Doll and Torkzadeh identified were supported by a confirmatory factor analysis, and suggest that this generic approach is suitable for measuring overall user satisfaction with computer systems, as well as with a specific application. McHaney and Cronan (1998) used the EUCS to test developers' and users' satisfaction with computer simulation software, with only minor changes to the wording to reflect the context. They concluded that the instrument was valid as a measurement of user satisfaction with computer simulation software, based on 411 responses. Aladwani (2002) used the EUCS as part of a wider study that examined the impact of management advocacy and internal computing support on user attitudes to computers and their satisfaction, calculating overall satisfaction as the mean of the 12 items. Somers, Nelson, and Karimi (2003) used the full 12-items to measure user satisfaction with enterprise resource planning systems, and found that all five dimensions were supported by their data from 407 end-users in 214 organisations.

Some research has raised questions about the suitability of the EUCS in different contexts. Chen, Soliman, Mao, and Frolick (2000) used concepts from the EUCS to study data warehousing users' satisfaction, but modified the wording to fit a data warehouse context. They also added six items relating to the support provided by the information centre to end users. Their results, based on responses from 42 users, showed that the items relating to information currency, availability, response time, ease of use, and accessibility were not well correlated with overall satisfaction, suggesting that these EUCS dimensions did not translate well to a data warehousing context. After eliminating these questions, and performing a factor analysis on the remaining questions, they found three dimensions: end-user support; accuracy, format, and preciseness of information; and fulfilment of user needs. Townsend, Demarie, and Hendrickson (2001) adapted the four items relating to satisfaction with information content in their study of desktop videoconferencing, finding that the scale had a Cronbach's alpha of 0.93, but do not provide a description of the specific changes they made. They also used three other items from the EUCS (accuracy, format, and ease of use), but used them as a 'system utility' construct, rather than as part of overall user satisfaction.

Four of the dimensions (content, accuracy, format, and timeliness) make an implicit assumption that access to 'information' generated by someone else is the primary purpose of the software or system(s); the ease-of-use factor is the only one that relates to the use of the software/system(s) to create information, or communicate it to others. This means that it may be difficult to apply the EUCS to software whose primary purpose is to create information. Studies that have used the EUCS to evaluate web sites or web-based information systems have had

differing results. Xiao and Dasgupta (2002) used a slightly modified EUCS instrument to measure user satisfaction with web-based portals. Their results, based on 340 responses from university students, had good psychometric values for all questions except 'Does the system provide sufficient information?'. Abdinnour-Helm, Chaparro, and Farmer (2005) used the EUCS for evaluating people's experience of using a web site, after making minor changes to the wording (such as changing 'system' to 'site'). Their results, based on 176 responses from university students, show that all five dimensions were identifiable, but that the 'timeliness factor had poor loadings. They suggest that this factor needs to be redefined to make it more relevant to this context, perhaps by changing it to measure the number of steps required to find the required information. Zviran, Glezer, and Avni (2006) used the 12 items to measure user satisfaction with commercial web sites. In contrast to Xiao and Dasgupta (2002) and Abdinnour-Helm, Chaparro, and Farmer (2005), they found that all five dimensions could be clearly identified, but that the first factor, content, accounted for 61.6% of the variance in user satisfaction, suggesting that the other four dimensions are much less closely linked to satisfaction in this context.

Etezadi-Amoli and Farhoomand (1996)

Etezadi-Amoli and Farhoomand (1996) developed an independent instrument to measure end-user computing satisfaction, arguing that previous instruments had been limited by their focus on the information products of the system being assessed, and that other aspects needed to be considered, in particular its individual and organisational impacts. They identified six dimensions: functionality, ease of use, quality of output, support, security, and documentation, based on 341 responses from people in a range of positions in 22 different organisations. Etezadi-Amoli and Farhoomand's approach is more general than that taken in the previous studies, and their inclusion of documentation and support as individual dimensions implicitly acknowledges the changing nature of the end-user computing environment, moving from being relatively passive 'consumers' of information produced by an organisation's IT department to more active users of software which is an integral part of their daily tasks. Despite this broader approach, the Etezadi-Amoli and Farhoomand instrument has not been used extensively in subsequent research.

Other instruments

The six measurement scales discussed above are not the only approaches to measuring user satisfaction with software or information systems. Zirvan and Erlich (2003) identify six studies that used a single question to measure overall user satisfaction, and eleven that used an independent approach, rather than using or modifying an established scale. Other

research has focused on developing instruments for specific settings or types of software, often using one or more of the six 'standard' scales as a starting point. Selected examples of these other approaches include:

- Palvia (1996) and Palvia and Palvia (1999), who developed a list of dimensions that contribute to small business owners' satisfaction with their IT systems; their measurement instrument included vendor support and training and education as dimensions that contribute to an overall satisfaction measure;
- Cho and Park (2001), who developed the electronic commerce user-consumer satisfaction index (ECUSI), with ten dimensions (product information, consumer service, purchase result and delivery, site design, purchasing process, product merchandising, delivery time and charge, payment methods, ease of use, and additional information services), and calculated an overall satisfaction score as $\sum_{d=1}^{n} s_d$;
- Ong and Lai (2007), who developed a scale to measure user satisfaction with knowledge management systems, using existing user satisfaction instruments and incorporating additional statements that are relevant to knowledge management. The final version, tested with 147 respondents using in-house knowledge management systems in four companies, showed that four dimensions (content, ease of use, personalisation, and community) explained 79.7% of the variance in overall satisfaction;
- Wang and Liao (2007), who developed a scale to measure mobile-commerce user satisfaction, which had 36 items, grouped into four dimensions (content quality, appearance, service quality, and ease of use);
- Ramasubbu, Mithas, and Krishnan (2008), who considered customer satisfaction with enterprise system support services, measuring overall satisfaction with a single question, and asking respondents to rate their satisfaction with four dimensions: responsiveness, technical skills, behavioural skills, and the extent of support, using a 10-point scale ranging from 'not satisfied' (1) to 'highly satisfied' (10). They used a single question per dimension, because their goal was to have a high response rate; to achieve that they believed they needed a short survey. However, because of this, they were unable to quantify the reliability or validity of their data. Ramasubbu, Mithas, and Krishnan found that behavioural skills were more important than technical skills in determining overall customer satisfaction, based on responses from 192 customers (a 77% response rate);
- Tojib, Sugianto, and Sendaya (2008), who developed a scale to measure user satisfaction with business-to-employee portals, finding

five dimensions: usefulness, confidentiality, ease of use, convenience, and portal design; and

- Bargas-Avila, Lötscher, and Sébastian (2009), who developed a new scale to measure user satisfaction with an intranet, finding two main dimensions (content quality and intranet usability) that explained 56.5% of the total variance in satisfaction.

2.7.3 *Dimensions of satisfaction*

The previous section showed how the various instruments to measure user satisfaction with information systems and software have evolved as the context for using technology changes. The underlying factors they measure relate to different aspects of this context. This section lists the individual factors identified in the instruments that were examined, grouped in broad categories to show whether they apply to aspects of the the system, the information it provides, the community, tools (which are resources needed to use the software), the user, or are specific to a particular type of application.

The list of system-related dimensions of satisfaction in Table 2 on the following page shows that 22 different underlying concepts have been measured as components of user satisfaction with a system/software. The only one that is common to a large number of studies is 'Ease of use'. Some concepts appear to be similar, such as Appearance, Layout, and Site design. Reliability and Service reliability are also likely to be measuring the same underlying concept, as are Ease of use and Usability. However, overall the table shows that most of these have been used in a single study, with no apparent overlap between their underlying meanings.

There is considerably more agreement within the information dimensions of satisfaction than with the system-related ones. Table 3 on page 45 lists the actual names used in the studies, but it is clear that there is considerable overlap among them. Some of the terms used appear to be synonyms for the same concept, such as 'Output quality' and 'Quality of output'; this concept might also be the same as 'Information quality', 'Content quality', and 'Information product'. The terms used to name the top four dimensions are the ones used by Doll and Torkzadeh's EUCS, and reflect its popularity as a standard instrument to measure user satisfaction.

Table 4 on page 46 shows that community-oriented dimensions have been considered by a number of researchers. From the names some appear to be similar or to overlap, such as EDP services, EDP staff, and EDP staff and services; and End-user support, Support, and Support factor. However, overall there appears to be little consensus on what dimensions should be measured to assess satisfaction with the community component of the software/system.

Table 2: System dimensions of satisfaction

DIMENSION	NO. OF STUDIES	REFERENCES
Ease of use	11	Doll and Torkzadeh (1988); Doll, Xia, and Torkzadeh (1994); Kim and McHaney (2000); Rainer and Harrison (1996); Somers, Nelson, and Karimi (2003); Abdinnour-Helm, Chaparro, and Farmer (2005); Zviran, Glezer, and Avni (2006); Etezadi-Amoli and Farhoomand (1996); Cho and Park (2001); Ong and Lai (2007); Wang and Liao (2007)
Security	2	Tan and Lo (1990); Etezadi-Amoli and Farhoomand (1996)
Appearance	1	Wang and Liao (2007)
Capability	1	Kekre, Krishnan, and Srinivasan (1995)
Confidentiality	1	Tojib, Sugianto, and Sendaya (2008)
Convenience	1	Tojib, Sugianto, and Sendaya (2008)
Functionality	1	Etezadi-Amoli and Farhoomand (1996)
Installability	1	Kekre, Krishnan, and Srinivasan (1995)
Intranet usability	1	Bargas-Avila, Lotscher, and Sebastian (2009)
Layout	1	Muyelle, Moenaert, and Despontin (2004)
Maintainability	1	Kekre, Krishnan, and Srinivasan (1995)
Performance	1	Kekre, Krishnan, and Srinivasan (1995)
Portal design	1	Tojib, Sugianto, and Sendaya (2008)
Reliability	1	Kekre, Krishnan, and Srinivasan (1995)
Service reliability	1	Miller and Doyle (1987)
Site design	1	Cho and Park (2001)
System characteristics	1	Miller and Doyle (1987)
System integration	1	Tan and Lo (1990)
System quality	1	McKinney, Yoon, and Sahedi (2002)
System utility	1	Tan and Lo (1990)
Usability	1	Kekre, Krishnan, and Srinivasan (1995)
Usefulness	1	Tojib, Sugianto, and Sendaya (2008)
User interface	1	Tan and Lo (1990)
Web system satisfaction	1	Cheung and Lee (2008)

Table 3: Information dimensions of satisfaction

DIMENSION	NO. OF STUDIES	REFERENCES
Accuracy	7	Doll and Torkzadeh (1988); Doll, Xia, and Torkzadeh (1994); Kim and McHaney (2000); Rainer and Harrison (1996); Somers, Nelson, and Karimi (2003); Abdinnour-Helm, Chaparro, and Farmer (2005); Zviran, Glezer, and Avni (2006)
Format	7	Doll and Torkzadeh (1988); Doll, Xia, and Torkzadeh (1994); Kim and McHaney (2000); Rainer and Harrison (1996); Somers, Nelson, and Karimi (2003); Abdinnour-Helm, Chaparro, and Farmer (2005); Zviran, Glezer, and Avni (2006)
Timeliness	7	Doll and Torkzadeh (1988); Doll, Xia, and Torkzadeh (1994); Kim and McHaney (2000); Rainer and Harrison (1996); Somers, Nelson, and Karimi (2003); Abdinnour-Helm, Chaparro, and Farmer (2005); Zviran, Glezer, and Avni (2006)
Content quality	2	Wang and Liao (2007); Bargas-Avila, Lotscher, and Sebastian (2009)
Information product	2	Baroudi and Orlikowski (1988); Doll, Raghunathan, Lim and Gupta (1995)
Information quality	2	Tan and Lo (1990); McKinney, Yoon, and Sahedi (2002)
Accuracy, format, and preciseness of information	1	Chen, Soliman, Mao, and Frolick (2000)
Fulfilment of user needs	1	Chen, Soliman, Mao, and Frolick (2000)
Information	1	Muyelle, Moenaert, and Despontin (2004)
Output quality	1	Raymond (1987)
Quality of output	1	Etezadi-Amoli and Farhoomand (1996)
Web information satisfaction	2	Cheung and Lee (2008)

Table 4: Community dimensions of satisfaction

DIMENSION	NO. OF STUDIES	REFERENCES
Training and education	2	Palvia (1996); Palvia and Palvia (1999)
Vendor support	2	Palvia (1996); Palvia and Palvia (1999)
Community	1	Ong and Lai (2007)
Consumer service	1	Cho and Park (2001)
EDP services	1	Doll, Raghunathan, Lim and Gupta (1995)
EDP staff	1	Doll, Raghunathan, Lim and Gupta (1995)
EDP staff and services	1	Baroudi and Orlikowski (1988)
End-user support	1	Chen, Soliman, Mao, and Frolick (2000)
IS staff quality	1	Miller and Doyle (1987)
IS staff responsiveness	1	Miller and Doyle (1987)
Management support	1	Tan and Lo (1990)
Relationships with management	1	Guimaraes and Gupta (1988)
Relationships with other departments	1	Guimaraes and Gupta (1988)
Strategic management	1	Miller and Doyle (1987)
Support	1	Etezadi-Amoli and Farhoomand (1996)
Support factor	1	Raymond (1987)
Technical image	1	Guimaraes and Gupta (1988)

Table 5: Tool-related dimensions of satisfaction

DIMENSION	NO. OF STUDIES	REFERENCES
Additional information services	1	Cho and Park (2001)
Product information	1	Cho and Park (2001)

Table 6: User-related dimensions of satisfaction

DIMENSION	NO. OF STUDIES	REFERENCES
Attitude to EDP staff	1	Raymond (1987)
End user control	1	Miller and Doyle (1987)
User attitude	1	Tan and Lo (1990)
User involvement	1	Miller and Doyle (1987)
User-system relationship	1	Raymond (1987)

Tools are information resources that support users, and make it possible for them to use the system/software independently. Table 5 on the preceding page lists two tool-related dimensions of satisfaction identified in the literature, which is lower than the number of dimensions found in the other categories. This may reflect researchers' lack of awareness of the importance of tools to software users, or it may mean that tools are only important in certain contexts.

Table 6 on this page shows that user-related dimensions of satisfaction are no longer measured as part of satisfaction, with the only studies that included these concepts being published 15 or more years ago. As Section 2.8 on the current page shows, user characteristics are now considered to be a factor that influences the way a person assesses their satisfaction with software, rather than something that contributes to their satisfaction.

The 8 dimensions of satisfaction in Table 7 on the following page have each been used by a single study, as might be expected. Each of these dimensions represents a concept that is important in the specific application context being studied, and is unlikely to be of interest outside that context.

2.8 OTHER CHARACTERISTICS RELATED TO USER SATISFACTION

The previous section discussed the aspects of an information system/-software that have been used to measure user satisfaction with a system/software. However, research has shown that other characteristics, such as the individual's background, or the wider organisational context, also have an effect on a respondent's reported level of satisfaction. A large number of constructs have been studied as predictor variables with respect to satisfaction, such as use, information quality, system quality, and support quality, which form part of Delone and McLean's success model (1992, 2003).

Table 7: Context-specific dimensions of satisfaction

DIMENSION	NO. OF STUDIES	REFERENCES
Cost/benefit to the organisation	1	Guimaraes and Gupta (1988)
Delivery time and charge	1	Cho and Park (2001)
Fulfilment of user needs	1	Chen, Soliman, Mao, and Frolick (2000)
Payment methods	1	Cho and Park (2001)
Personalisation	1	Ong and Lai (2007)
Product merchandising	1	Cho and Park (2001)
Purchase result and delivery	1	Cho and Park (2001)
Purchasing process	1	Cho and Park (2001)

2.8.1 *The FLOSS context*

Two studies that examined individual and organisational factors were used to identify predictor variables relevant to the FLOSS context of this research. McKeen, Guimaraes, and Wetherbe (1994) found that user participation had a 0.414 correlation with user satisfaction, significant at $p \leqslant .001$. They also found that both user influence and user–developer communication had positive correlations with satisfaction, significant at $p \leqslant .05$, while task and system complexity acted as moderators of the relationship between participation and satisfaction. In a subsequent study, Guimaraes, Staples, and McKeen (2003), examined the relationship between satisfaction and user participation, user experience, user–developer communication, user training, and user conflict. Their findings showed a similar correlation between participation and satisfaction, but found no significant relationship between satisfaction and user influence or user–developer communication. The variables they studied are all relevant in a FLOSS context, since participation in a FLOSS project varies from person to person, and FLOSS projects rely on a range of asynchronous communication tools to allow participants to work effectively together. Similarly, participants in a FLOSS project are likely to have a range of experience and training, suggesting that these would be useful variables to examine.

One other study was used to identify possible factors to include in this research. Mahmood, Burn, Gemoets, and Jacquez (2000) conducted a meta-analysis of 45 user satisfaction studies, and distinguished between

experience and skills/knowledge. Experience related to the length of time the user has been using an information system/software, while skills and knowledge were more general. Since many FLOSS projects are relatively new, and people's experience with them may be limited, this research considered both concepts as predictor variables, to see which, if either, of them is significant.

The sections below discuss each of these concepts in more detail, grouped as individual characteristics and organisational characteristics, following the approach used by Sabherwal, Jeyaraj, and Chowa (2006). The individual characteristics are skills and knowledge, experience, training, and participation, and the organisational characteristics are user influence, user–developer communication, task complexity, and system complexity.

2.8.2 *Individual characteristics*

Individual characteristics vary from person to person, and include attributes that relate to knowledge and skills, and level of participation. Since a person's judgement of how satisfied they are with a product or service involves an assessment of how well it meets their expectations, it makes intuitive sense that there will be variations between individual users' satisfaction because of differences in their backgrounds.

Skills and knowledge

People with higher levels of skills in using computers may be more satisfied with the software that they use, since they are likely to be able to work more efficiently, and may also be more able to solve their own problems. In addition, people with higher skills may have more realistic expectations about what can and cannot be done than people with low skill levels, and therefore have smaller gaps between their expected and actual levels of performance. In a FLOSS context, individual skills and knowledge may be particularly important, because in many FLOSS projects, support is provided by other participants, rather than by a commercial service provider, and the level of this support may vary from project to project.

Research interest in individual skills and knowledge began in the late 1990s, with Munro, Huff, Marcolin, and Compeau (1997) publishing one of the first articles to examine what they termed 'user competence'. Blili, Raymond, and Rivard (1998) found that end-user skills had a low but statistically significant effect on user satisfaction, accounting for 16% of the variance ($p<.001$). Similarly, Jang (2009) found a path coefficient of 0.12 between computer competence and satisfaction, significant at $p<0.01$, in a study of users of an electronic document exchange system. In contrast, in their study of small business owners' satisfaction with

their computer systems, Palvia and Palvia (1999) found no significant correlation between skills and satisfaction.

Different authors have taken different approaches to measuring users' skills. Blili, Raymond, and Rivard (1998) used a concept they called 'end user computing competence', which is related to user skills and knowledge, asking their respondents to choose from four categories of computer user, ranging from non-programming end-user through to functional support personnel. This classification was based on the way users accessed information in databases, assigning them to a higher category if they used command-level and/or programming languages rather than pre-defined menus. This distinction is much less clear today because menu-driven interfaces are now more common than command-line ones.

Palvia and Palvia (1999) used a single item that asked respondents to assess their computer skills as poor, average, and good. This approach is straightforward, but it relies on users' ability to judge their skills realistically, and the use of only three values limits the scale's ability to discriminate between different levels of user skills.

More recently Torkzadeh and Lee (2003) developed a 26 item scale to measure end-user computing skills covering technical skills, business knowledge, educational background, and computing skills. The instrument was tested by asking practitioners to identify any gaps in coverage, and whether the questions could be easily understood. Data from 282 respondents showed that 12 of the questions had good correlation with respondent's overall assessment of their skills, and high internal reliability. This is a generic scale, suitable for use with any type of application software, unlike the 25-item scale to assess user knowledge of using spreadsheets developed by McGill and Dixon (2001), which included questions that related only to spreadsheet functionality.

Training

Effective training can be expected to result in more confident users, which would increase their satisfaction. Considerable research has investigated this relationship, starting with Nelson and Cheney (1987). Although training may be related to knowledge and skills, it is generally measured separately. Training in using software and information systems can come from a variety of sources, including self-study that makes use of documentation or online resources, or through formal courses offered by software vendors or other service providers.

However, empirical research has shown varying correlations between training and satisfaction. Guimaraes, Staples, and McKeen (2003) found that there was a 0.54 correlation between training and satisfaction, significant at $p \leqslant .01$; in contrast, Al-Gahtani and King (1999) found no significant relationship between training and satisfaction. In a meta-analysis of 27 studies that examined the relationship between training

and what they termed 'implementation success' (usually measured as user satisfaction), Sharma and Yetton (2007) found that the mean correlation was 0.28, with a range of −0.05 to 0.51. This range of correlations suggests that the impact of training is sensitive to the context in which the software or information system is being used.

The Nelson and Cheney scale for measuring the effectiveness of training asks people to indicate the extent of training they received from different sources, such as vendor training, in-house training, or self-study.

Experience

Section 2.5 on page 28 explained that a person's assessment of their satisfaction with a product or service is a cumulative judgement, based on their experiences. This suggests that extent of experience may affect people's satisfaction with a FLOSS project. Experience is related to knowledge and skills, but differs in that it includes the concept of time or duration. As people gain experience in using software or information systems, they are likely to develop more realistic expectations, which will then affect their satisfaction ratings. In a meta-analysis of 45 studies of user satisfaction, Mahmood, Burn, Gemoets, and Jacquez (2000) found that user experience had had a large combined effect size (0.565) on respondents' level of satisfaction. In an analysis of 54 articles dealing with user acceptance of information technology, Sun and Zhang (2006) found that experience had a moderating effect on the relationship between perceived ease of use and behavioural intention. However, other research has found conflicting results for the relationship between experience and satisfaction. Guimaraes, Staples, and McKeen (2003) found that experience had a 0.33 correlation with satisfaction, significant at $p \leqslant .01$, though it had a very small effect, increasing R^2 by only 0.03. Lawrence and Low (1993) studied users of two systems, and found that user experience had a significant influence on satisfaction at the 0.01 level for 59 users of one system, but was not significant for 96 users of a different system. The differences between these results suggest that the effect of experience varies with the context of the information system/software being assessed. No studies have examined the relationship between experience and satisfaction in the context of a FLOSS project.

Experience is typically measured using a scale first used by Igbaria, Guimaraes, and Davis (1995), which asks respondents to rate the extent of their experience relative to other members of the project team for five aspects of technology use: using systems of this type, using this particular system, using computers in general, as a member of a development team, and as a member of this development team.

Participation in system development

Participation in system development, and its effects, have been the subject of extensive research in information systems. The terms used for participation vary, with some researchers using the term 'user involvement' rather than 'user participation'. This section reviews research that has used both terms, but follows Barki and Hartwick's (1989) recommendation to distinguish between participation, which they defined as "the set of operations that individuals have or have not performed" (p,53), and involvement, which they argued was a subjective state representing the user's judgement of the system's importance and relevance. This thesis is concerned with actual participation, rather than the psychological state of involvement.

In a meta-analysis of 45 studies that considered user satisfaction as an outcome variable, Mahmood et al. (2000) found that user participation in system development had the largest combined effect size on satisfaction of all the variables considered. Their meta-analysis of 121 studies that examined relationships between a range of individual and organisational factors and satisfaction, found participation had a path coefficient of .24 with satisfaction, significant at $p<0.0001$. More recently, He and King (2008) conducted a meta-analysis of 82 empirical studies of user participation in system development, which showed that participation had been studied as a predictor variable for the outcome variables satisfaction, use intention, use (meaning actual use), individual impact, team performance, project quality, and project success. Their results showed that a majority of studies (47) had examined the relationship between participation and satisfaction, and that participation and satisfaction had the highest integrated effect size of 0.325, significant at $p<.000$ level. The individual effect sizes for participation ranged from 0.10 to 0.68, which suggests that context has an impact on the size of the correlations between participation and satisfaction.

Most studies have considered the impact of user participation on satisfaction only from the users' point of view. Subramanyan, Weisstein, and Krishnan (2010) examined the relationship between user participation and satisfaction from the developers' perspective as well. Their data were gathered from a conventional systems development context, with clearly defined roles for users and developers. They found differences between user and developer satisfaction that varied with the extent of user participation in projects at two stages of their life-cycle: new developments and maintenance projects. Their findings generally showed that users who had extensive participation in new developments were less satisfied than those who participated less, while the opposite was the case for maintenance projects. Developer satisfaction followed a different pattern, increasing along with user participation for new developments, but decreasing for maintenance projects.

Participation in system development has been measured in different ways. Markus and Mao (2004) noted that some researchers have focused on the system development process, asking questions about participation in requirements generation, software development, and system testing, while other researchers have focused on implementation and project management. Igbaria and Guimaraes (1994) asked survey respondents to indicate how much they had been involved in nine specific activities; this approach was also used by Yoon, Guimaraes, and O'Neal (1995). Doll and Torkzadeh (1989b) developed a general-purpose instrument to measure participation, with eight items relating to three aspects of a conventional system development project: systems analysis, implementation, and administration. However, because all of these measures assume that the project being assessed used a conventional life-cycle approach to system development, none is suitable for the context of a FLOSS project. This is because FLOSS development processes typically lack these explicit stages.

2.8.3 *Organisational characteristics*

In a FLOSS context, organisational characteristics refer to characteristics of the overall project, or characteristics of the respondent's interactions with the project and/or software.

Perceived influence

When people participate in a software development project, they expect this participation to have an effect. Markus (1983) described the main goal of participation as the improvement of the system's functionality and interface so that it is a closer match to users' mental models (p.440). For this to happen, the users must be able to influence the development process and priorities. Edström (1977) was one of the first to study the impact of users' perceived influence on their perceptions of the success of a software development project, finding that there was a .51 correlation between the two (significant at the 5% level). McKeen, Guimaraes, and Wetherbe noted that if there is no or limited participation, then there can be little or no influence (1994, p.434). Doll and Torkzadeh (1989a) suggest that perceived influence is important when people participate in systems development because it makes them feel that their time has been well spent, and their efforts are appreciated. McKeen, Guimaraes, and Wetherbe (1994) found that perceived user influence had a 0.208 correlation with satisfaction, significant at $p \leqslant .05$. More recently, Larsen (2009) found that influence had a 0.29 path coefficient with Doll and Torkzadeh's (1988) End-User Computing Satisfaction (EUCS) instrument's accuracy, significant at $p \leqslant .01$, and a .18 path coefficient with EUCS ease of use, significant at $p \leqslant .05$. These values were

similar to those for communication (.25 with EUCS accuracy and .23 with EUCS ease of use, both significant at $p \leqslant .05$).

Several approaches to measuring perceived influence have been developed. Edström (1977) used a single question that asked respondents how much influence they had had in each of six stages in the software development process. The most common instrument for measuring perceived influence was developed by Franz and Robey (1986); it asks respondents to indicate the extent to which they, rather than IS staff, made key decisions at the design and implementation stages of a software development project. An implicit assumption in this measure is that users and IS staff have different responsibilities, which may not necessarily be the case in a FLOSS context, since the distinction between users and developers can be less clear-cut.

Perceived developer communication quality

Communication is generally assumed to be an important aspect of developing information systems/software, since developers need to understand user requirements in order to implement them effectively. Studies of the role of good communication in information system development can be traced back to Edström (1977), with many other researchers stressing its importance. Some examples include Robey and Farrow (1989), and Gefen and Keil (1998). Robey and Farrow (1989) found that good communication reduced conflict during the development of an insurance processing system, while Gefen and Keil (1998) found that perceived developer responsiveness, a construct which is related to developer communication, had a .36 path coefficient with perceived usefulness and a .39 path coefficient with perceived ease of use (both significant at $p<.01$). McKeen, Guimaraes and Wetherbe (1994) found that there was a low, but statistically significant correlation between perceived quality of developer communication and satisfaction (.195, $p \leqslant 0.05$). While these examples suggest that the relationship between developer communication and satisfaction is consistently strong, Guimaraes, Staples, and McKeen (2003) found no significant correlation between the two constructs in their results, suggesting that context may be important to this relationship. Perceived communication quality has not yet been tested in a FLOSS context.

Perceived complexity

Rogers' widely used Diffusion of Innovation theory (DOI) included the concept of complexity as one of five factors that influences people's willingness to adopt a new technology (2003). Rogers defined complexity as "the degree to which an innovation is perceived as relatively difficult to understand and use" (2003, p.15). Complexity has been studied in the context of information system adoption more often than in studies of user satisfaction. Though some studies, such as Agarwal and Prasad

(1997), replaced the concept of complexity with ease of use, Rogers' definition makes it clear that the DOI concept of complexity is broader, involving the cognitive element of understanding as well as ease (or difficulty) of use. Sharma and Yetton (2007) made a clear distinction between the two concepts, arguing that complexity is a broader concept, including technology and support requirements as well as ease of use. Sun and Zhang (2006) recognised this difference, suggesting that perceived complexity affects the relationship between perceived ease of use and behavioural intention. They argued that perceived ease of use is more important when the underlying technologies are perceived to be difficult to understand than when they are considered simple. In the FLOSS context for this thesis, the broader concept of complexity was judged to be more appropriate than the narrower concept of 'ease of use', particularly because the line dividing users and developers is less clear than in conventional systems development projects.

McKeen, Guimaraes, and Wetherbe (1994) identified two types of complexity: *system complexity* and *task complexity*. System complexity is associated with the overall environment, while task complexity relates to uncertainty participants have about aspects of their activities. Where people need to choose between alternative ways of accomplishing a specific goal, task complexity will be high. System complexity, on the other hand, relates to the way the system as a whole is perceived. McKeen, Guimaraes, and Wetherbe (1994) found that both task and system complexity moderated the relationship between participation and satisfaction. In particular, they found that the correlation between participation and satisfaction was higher when perceived task and system complexity were both high, and lower when task and system complexity were both low. They argued that this showed that it is particularly important to involve users in information system/software development when the requirements were complex, since that would help ensure the system's developers developed a better understanding of what functions the system needed to provide. In contrast, Blili, Raymond, and Rivard (1998) found that task complexity had a direct relationship with satisfaction, acting as an independent predictor. Their results showed that task complexity had a $-.05$ path coefficient (significant at $p<.05$) with satisfaction, suggesting that as user tasks became less certain (i.e., more complex), their perceived satisfaction with software decreases slightly.

As the range of tasks people can carry out with software increases, the importance of both task and system complexity as a moderator of satisfaction may increase, and so this concept was judged to be relevant to the context of this research. Task complexity is usually measured using Rizzo, House, and Lirtzman's 1993 scale assessing role conflict and ambiguity, and system complexity using Tait and Vessey's 1988 scale that asked people to assess complexity relative to other systems they had used.

2.8.4 *Satisfaction and FLOSS projects*

Only a few studies have examined satisfaction in a FLOSS context.

Crowston, Howison, and Annabi (2006) posted a message on SlashDot, a popular developer news site, asking for readers' views on what measure(s) would represent FLOSS project success. Two of the resulting suggestions were user and developer satisfaction, suggesting that research into participant satisfaction with FLOSS projects would be very relevant to the developer community. However, only four studies were identified that considered satisfaction in this context.

The first, Chin and Cooke (2004), used a model derived from research on factors that influence job satisfaction to measure the relationship between Linux developers' intrinsic and extrinsic motivations for being involved with the project and their intrinsic and extrinsic satisfaction with their participation in a FLOSS project. Their results showed that intrinsic satisfaction, related to individual achievement and challenges, was correlated with opportunities to develop technical skills, intrinsic and extrinsic motivation, and group trust. Extrinsic satisfaction was less important, and did not correlate as strongly with other variables, suggesting that extrinsic satisfaction was not as important to their respondents. Their results were limited by a small sample size (42), and low correlations between measurement items for their constructs which meant that most of their measures involved a single item. In addition, because some FLOSS participants work on the projects as volunteers, rather than as part of their paid employment, using concepts related to job satisfaction may not have been appropriate in this context. This may also explain why the intrinsic satisfaction component of the model was more important than the extrinsic one.

The second, Lee, Kim, and Gupta (2009), measured user satisfaction with the Linux operating system as one component of a research model based on DeLone and McLean's 2003 IS success model. They measured satisfaction only from the users' perspective, and used McKinney, Yoon, and Zahedi's 2002 4-point scale to measure overall satisfaction; their results showed that there was a significant positive relationship between both perceived software quality and perceived community service quality and user satisfaction. However, perceived community service quality had no significant relationship with use, though perceived software quality did. Their measure of perceived community service quality was based on concepts in the SERVQUAL[8] model, but omitted the 'tangible' component. Their rationale for doing so was that the community was virtual, with no tangible components, but this overlooked such tools as documentation and email discussion list archives. This shows again that researchers have given little attention to the importance of the role tools play in a FLOSS context.

8 SERVQUAL is a discrepancy-based approach to measuring customer perceptions of service quality.

Casaló, Cisneros, Flavián, and Guinaliu (2009) looked at the influence of reputation on community members' satisfaction with their participation in the social networks associated with a FLOSS project. They measured overall satisfaction using three questions that asked respondents to say whether they made a correct decision to participate, how satisfied they were with their experience of the online network, and how satisfied they were with their relationship with the online network and its members. Casaló, Cisneros, Flavián, and Guinaliu did not distinguish between different types of participation, and considered all respondents to fit in the single category of 'developer'.

Braccini, Silvestri, and Za (2010) conducted a small-scale survey of students who used the Moodle[9] electronic learning management system. They based their questions on the ISO 9126 and ISO 2500 software quality model, restricting them to the dimensions they judged to be relevant to end users: functionality, reliability, usability, and quality in use. Their results were limited by the small size of their sample (59 responses), and the use of a 4-item Likert scale for responses. This meant the response distribution had a low variation, shown by the standard deviations which ranged from .20 to .66. Their results showed that respondents had a high level of general satisfaction (a mean of 3.41 out of 4), and that reliability had the lowest score (2.92).

2.8.5 *Implications for this research*

While the examples in the previous sections demonstrate the popularity of satisfaction as a research topic in information systems, they also show that there have been many different approaches to defining and measuring satisfaction in this context, with no consensus about which approach is best. This extensive body of literature has not resulted in a single, widely used scale that is suitable to measure user satisfaction with software/information systems. Although most authors refer to 'user satisfaction' as if all research has been measuring the same underlying concept, the preceding discussion shows that there is considerable variation in what is being asked and how the various researchers have named the factors that contribute to satisfaction.

The recent research literature also appears to show that the number of approaches to measuring satisfaction is increasing, rather than consolidating around a single approach. Though Chen, Rogers, and He (2008) suggested that this means there are fundamental problems with the way satisfaction has been defined and measured, most of the earlier studies have demonstrated acceptable internal reliability, with Cronbach's alpha values of 0.70 or higher. In addition they have shown that they have adequate construct validity, generally indicated by the results of a factor analysis showing that individual questions map onto distinct

9 Moodle is a FLOSS learning management system originally released in 2002 and installed in over 49,000 sites by mid-2010. (http://moodle.org/stats/)

dimensions. There have been several studies of the test–retest reliability of the Doll and Torkzadeh (1998) EUCS instrument. Though Galetta and Lederer (1989) questioned its reliability, subsequent research by Hawk and Raju (1991), Torkzadeh and Doll (1991), Hendrickson, Glorfield, and Cronan (1994), and McHaney, Hightower, and White (1999) have confirmed that the instrument demonstrates adequate test–retest reliability.

Since the evidence presented in these articles shows that all these instruments gathered reliable and valid data in the specific research context where they were used, one possible conclusion is that there are underlying difficulties with the way the construct of 'user satisfaction' has been conceptualised. However, another interpretation comes from acknowledging satisfaction as a complex construct, composed of a number of interrelated aspects. Oliver identified six different approaches to assessing satisfaction, relating to the different types of comparisons people may make when they indicate their satisfaction with a product or service. These are need fulfilment, expectations, quality, value, equity, and regret (2010, p.19). It may be unrealistic to expect a single instrument to capture all of these aspects. The current diversity reinforces Seddon et al.'s 1999 comment that this is a strength of IS research into satisfaction. The range of approaches means that researchers have a choice of instruments and approaches which can be adapted to the specific context for the software/system being assessed.

The purpose of this research, to examine factors that influence participant satisfaction with FLOSS projects, is a new context for information systems satisfaction research. None of the existing instruments has been developed specifically for this context, and this research therefore needed to determine which dimensions are most relevant to participants in a FLOSS project. This was done in part by identifying key characteristics of FLOSS projects from previous research, complemented by interviews with a purposive sample of people with differing levels of involvement in selected FLOSS projects.

2.9 THEORETICAL MODEL FOR THIS RESEARCH

The theoretical model developed for this research draws primarily on factors that affect user satisfaction with information systems/software identified in Mahmood et al. (2000), plus McKeen, Guimaraes, and Wetherbe (1994) and Guimaraes, Staples, and McKeen (2003). These factors were supplemented with two additional factors from the FLOSS literature — perceived process and product openness.

Figure 2 on the facing page illustrates the preliminary research model developed for this research. Each of the factors is discussed in the following sections.

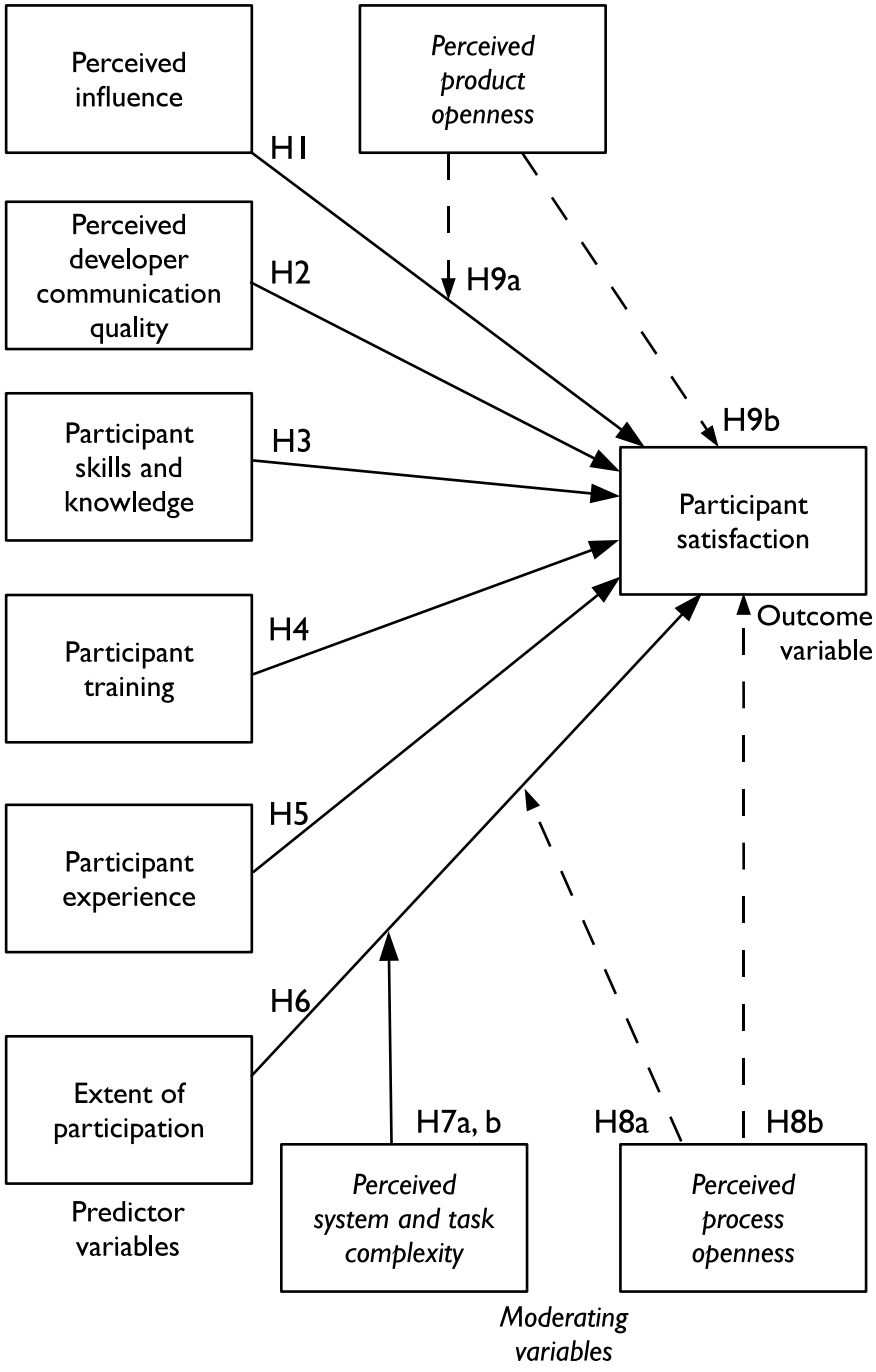


Figure 2: Research model

2.9.1 *Perceived influence*

Perceived influence was discussed in Section 2.8.3 on page 53. In a FLOSS context, perceived influence relates to the extent to which a participant's ideas and suggestions are implemented in the software, or affect other decisions relevant to the project. Previous research that considered the relationship between perceived influence and satisfaction has had mixed results, with McKeen, Guimaraes, and Wetherbe (1994) finding that there was a statistically significant, positive correlation between the two variables. In contrast, Guimaraes, Staples, and McKeen (2003) found no statistically significant relationship. This suggests that context plays a role in determining how strong the relationship between perceived influence and satisfaction is, which justifies its inclusion in the current research.

2.9.2 *Perceived developer communication quality*

Perceived developer communication quality was discussed in Section 2.8.3 on page 54. Since most FLOSS projects cross organisational boundaries, and may involve participants from around the world, this construct was considered to be particularly relevant to the current research. While it makes intuitive sense that there would be a strong correlation between perceived developer communication quality and satisfaction, the results of two studies that considered this factor differ. McKeen, Guimaraes, and Wetherbe (1994) found a low but statistically significant correlation between the two variables, but Guimaraes, Staples, and McKeen (2003) found no evidence of a statistically significant relationship. This suggests that it is similar to perceived influence, and that context may play a role in determining how strong the relationship between perceived developer communication quality and satisfaction is, which justifies its inclusion in the current research.

2.9.3 *Participant skills and knowledge*

Participant skills and knowledge were discussed in Section 2.8.2 on page 49. In a FLOSS project, participants may come from a wider variety of backgrounds than in a conventional software development project, which typically involves employees selected because they have the required skills and knowledge. Since FLOSS projects have flexible boundaries, and may involve people who have never met face-to-face, the same selection criteria are unlikely to apply. As with perceived influence and perceived developer communication quality, previous research into the relationship between participant skills and knowledge and satisfaction has had varying results, with Blili, Raymond, and Rivard (1998) and Jang (2009) finding evidence to support a statistically significant, pos-

itive relationship, while Palvia and Palvia (1999) found no significant correlation. Again, context may play a role in determining when this relationship is significant, which justifies including participant skills and knowledge in the current research.

2.9.4 *Participant training*

Participant training was discussed on page 50 in Section 2.8.2. Training has not previously been considered in the context of a FLOSS project, but as with skills and knowledge, there is likely to be considerable variation in the type of training that participants have had. As with the constructs discussed above, previous research into the relationship between training and satisfaction has had varying results, with Sharma and Yetton (2009) finding that correlations ranged from -.05 to .51 in their meta-analysis of 27 studies. It appears that context again plays a role in determining when this relationship is significant, which justifies including participant training in the current research.

2.9.5 *Participant experience*

Participant experience was discussed in Section 2.8.2 on page 51. In a FLOSS project, participants are likely to have differing levels of experience, and as with participant skills and knowledge, standard criteria are unlikely to be used to select participants for a FLOSS project. While some research, such as Guimaraes, Staples, and McKeen (2004), has shown a significant correlation between experience and satisfaction, other research, such as Lawrence and Low (1993), has found the opposite. Context appears to play a role in determining when this relationship is significant, which justifies including participant experience in the current research.

2.9.6 *Extent of participation*

The concept of participation was discussed in Section 2.8.2 on page 52. This factor measures the extent to which individual users, or user representatives, are active participants in the FLOSS project's community. There is an extensive body of research discussing the importance of user participation in system development, and overall these show that there is a strong relationship between user participation and satisfaction. He and King's 2008 meta-analysis showed an overall positive correlation between participation and satisfaction, though its strength varied, and appeared to be context dependent.

FLOSS projects evolve only when members of their communities participate in project-related activities, and the ways in which they contribute code has been studied in the past. This material was reviewed

in Section 2.3.1 on page 18. However, the relationship between extent of participation and satisfaction has not yet been studied in a FLOSS context, which justifies including this concept in the current research.

The standard approach to measuring participation involves generating a list of activities that involve users, and asking respondents to indicate whether they had carried out each activity. As approaches to developing software change, these lists of activities have also needed to be modified. For example, McKeen, Guimaraes and Wetherbe (1994) used Ives and Olson's 1984 44-item list of activities as a starting point, but removed items that no longer applied, modified the wording of others to reflect changes in terminology, and added five items to cover newer forms of participation. Barki and Hartwick (1994) said that a measure of user participation should include both formal and informal activities, and should cover all stages of the system development process. Their measure, which included 25 items, was based on the traditional waterfall approach to developing information systems, with clearly defined stages and built-in assumptions about the roles of users and information systems staff.

Section 2.3.1 on page 18 showed that a definitive list of FLOSS roles and activities has not yet been developed, and none of the existing scales for measuring participation is suitable for this context. This led to the first sub-question for this research, *What types of contributions do participants make to free/libre and open source software projects?* This question needed to be answered in order to develop a suitable measurement scale to assess the extent of participation in a FLOSS project.

2.9.7 *Perceived complexity*

Section 2.8.3 on page 54 discussed the concept of complexity and its relationship to satisfaction, and identified two types of complexity: system complexity and task complexity. FLOSS projects cover a range of application areas, and can be assumed to vary in complexity. McKeen, Guimaraes, and Wetherbe (1994) showed that both system and task complexity modified the relationship between participation and satisfaction. Since the relationship between complexity and satisfaction has not yet been studied in the context of a FLOSS project, it was included as a moderating variable in this project's research model.

2.9.8 *Perceived process openness*

Perceived process openness was discussed in Section 2.3.2 on page 24. This concept relates only to FLOSS projects, and was first developed by Ye et al. (2005). It refers to the extent to which FLOSS project participants are able to participate in project decision-making. There have so far been no empirical studies of the impact of process openness on participant

satisfaction, and no scales exist to measure it. Two relationships between perceived process openness and satisfaction have been hypothesised: first, that it is an independent predictor variable for the outcome variable satisfaction, and second that it moderates the relationship between extent of participation and satisfaction. The development of this scale for this research is discussed in more detail in Chapter 3.

2.9.9 *Perceived product openness*

Perceived product openness was discussed in Section 2.3.2 on page 24. It is similar to perceived process openness in that it was identified by Ye et al. (2005), and it was conceptualised in the context of a FLOSS project. Perceived product openness relates to the way the project releases new code, and the extent to which information about the project's future plans is available to FLOSS project participants. There have so far been no empirical studies of the impact of product openness on participant satisfaction, and no scales exist to measure it. Two relationships between perceived product openness and satisfaction have been hypothesised: first, that it is an independent predictor variable for the outcome variable satisfaction, and second that it moderates the relationship between perceived influence and satisfaction. The development of a measurement scale for this research is discussed in more detail in Chapter 3.

2.9.10 *Hypotheses*

Figure 2 on page 59 shows the hypotheses that were tested in this research. Hypotheses supported by previous research are shown as solid lines in Figure 2 on page 59, while those that are new are shown as broken ones.

H1 The higher the perceived participant influence, the higher the participant satisfaction.

Previous studies of this relationship have had differing results. McKeen, Guimaraes, and Wetherbe (1994) found that (perceived) user influence was an independent predictor of user satisfaction ($p \leqslant 0.05$), while Guimaraes, Staples, and McKeen (2003) found that it had no significant relationship with satisfaction. This discrepancy suggests that the relationship needs further testing. In addition, both studies surveyed users involved in in-house systems development projects, and cannot necessarily be generalised to the type of cross-organisational teams typically found in a FLOSS project.

H2 The higher the quality of perceived community member communication, the higher the participant satisfaction.

As above, previous studies of this relationship have had different results. McKeen, Guimaraes, and Wetherbe (1994) found that (perceived) user/developer communication was an independent predictor of user satisfaction ($p \leqslant 0.05$), while Guimaraes, Staples, and McKeen (2003) found that there was no significant relationship between the variables. This discrepancy suggests that the relationship between these needed further testing. As with participant influence, the relationship needs to be studied in a FLOSS context. Since much of the literature on FLOSS projects emphasises the importance of communication between participants, we would expect to find a positive relationship between communication and satisfaction.

H3 There is a positive relationship between participant skills and knowledge and participant satisfaction.

Previous studies of this relationship have had varying results. Lee, Kim, Lee (1995) and Palvia (1996) found that there was no significant relationship between perceived skills and satisfaction, while Yoon and Guimaraes (1995) found a statistically significant relations between user characteristics (which included user skills) and user satisfaction (.442, significant at $p < .01$).

H4 There is a positive relationship between participant training and participant satisfaction.

Training can increase people's confidence in their ability to use software effectively. However, previous studies of the relationship between training and satisfaction have had inconclusive results. The role of training in a FLOSS context has not been previously tested with empirical data.

H5 There is a positive relationship between participant experience and participant satisfaction.

Mahmood et al.'s (2000) meta-analysis identified 12 studies that considered the impact of user experience on satisfaction, all of which had a statistically significant, positive relationship between experience and satisfaction.

H6 There is a positive relationship between the extent of participation and participant satisfaction.

Previous research has consistently shown that there is a statistically significant relationship between participation and satisfaction. Examples include McKeen, Guimaraes, and Wetherbe (1994) and Lawrence and Low (1993), both of which found positive correlations between the two factors significant at $p < .001$.

H7A The greater the perceived system complexity, the greater the relationship between extent of participation and participant satisfaction.

H7B The greater the perceived task complexity, the greater the relationship between extent of participation and participant satisfaction.

McKeen, Guimaraes, and Wetherbe (1994) found that both system and task complexity were a moderators of the relationship between participation and satisfaction ($p \leqslant 0.05$); the more complex the system, the stronger the relationship between participation and satisfaction. FLOSS projects range from extremely simple to very complex, and this hypothesis will examine the effect of perceived complexity in a FLOSS context.

H8A The higher the perceived process openness, the greater the relationship between extent of participation and participant satisfaction.

H8B There is a positive relationship between process openness and participant satisfaction.

Ye et al. (2004) identified the FLOSS project characteristic of product openness, but did not examine its impact on participants' satisfaction or on levels of participation. Hypotheses 8a and 8b represent two possibilities. Under Hypothesis 8a, process openness influences the relationship between perceived participant influence and participant satisfaction, while Hypothesis 8b says that it has a direct influence on participant satisfaction. No previous research has been done to test this relationship, and therefore the proposed research will examine both hypotheses.

H9A The higher the perceived product openness, the greater the relationship between perceived participant influence and participant satisfaction.

H9B There is a positive relationship between product openness and participant satisfaction.

Ye et al. (2004) identified the FLOSS project characteristic of product openness, but did not examine its impact on participants' satisfaction. Hypotheses 9a and 9b represent two possibilities. Under Hypothesis 9a, product openness influences the relationship between perceived participant influence and participant satisfaction, while Hypothesis 9b says that it has a direct influence on participant satisfaction. No previous research has been done to test this relationship, and therefore the proposed research will examine both hypotheses.

2.10 SUMMARY

This chapter reviewed selected literature on FLOSS project structures, community practices, the nature of satisfaction, and how it has been measured for information systems/software. This showed that researchers

have taken a range of approaches to measuring satisfaction, and that satisfaction measures have evolved as information systems/software moved from mediated to ubiquitous computing, and as web-based information systems became more popular.

The chapter also discussed factors that influence user satisfaction judged as being relevant to the context of a FLOSS project, and presented a preliminary research model and hypotheses. The next chapter discusses the research methodology used to answer the research questions, including the way the concepts were measured.

3

METHODOLOGY

This section begins with a discussion of the philosophical paradigms that shaped the research design, followed by a description of the research methodology.

3.1 PHILOSOPHICAL PARADIGM

All research is based on fundamental assumptions about the nature of the world and sources of knowledge about it. These assumptions are often referred to as the research philosophy (Pickard 2007, p.3), worldview, or paradigm (Creswell and Plano Clark 2007, p.21). The choice of research design is influenced by these assumptions, as well as the nature of the research questions. Bryman identified two main aspects of the philosophical worldview that need to be considered: ontology and epistemology (2008, p.13–21). The specific ontology and epistemology that underpin a research project also determine the most suitable methodology to answer the research questions. The three main types of methodology are qualitative, quantitative, or a combination of the two generally referred to as 'mixed methods research' (Creswell and Plano Clark 2007, p.5).

3.1.1 *Ontology*

The term 'ontology' refers to the researcher's position about the nature of reality, especially social concepts (Bryman 2008, p.18; Creswell and Plano Clark 2007, p.21; Henn, Weinstein, and Foard 2006, p.17). Two of the main ontologies used in social research are *objectivism* and *constructionism*.

Objectivism takes the perspective that social phenomena have an independent existence and can be treated as objects. Researchers who use an objectivist ontology assume that there is a single reality that can be described or measured, and that the researcher is an independent observer of the phenomena. In addition, objectivism focuses on relationships between concepts (Bishop 2007, p.49).

In contrast, constructionism says that social phenomena are produced by people's interactions, and that there are multiple views of reality, based on individual perspectives (Creswell and Plano Clark 2007, p.22). This approach says that the meaning of social concepts changes slowly as culture evolves (Bishop 2007, p.65). Constructionists often study

social phenomena to understand how this meaning changes over time (Bryman and Bell 2007, p.24).

A researcher's choice of ontology is reflected in the selection of a research design. Objectivist research focuses on measurement, and often uses quantitative methods such as surveys and statistical analysis in order to produce generalisable results, while constructionist research focuses on developing a rich description of attitudes and experiences, focusing on differences between individuals gathered from qualitative data. Section 2.5 on page 28 and Section 2.6 on page 29 showed that research into satisfaction generally follows an objectivist ontology, focusing on measuring satisfaction and understanding factors that influence it. Therefore, this research also takes an objectivist approach, which is demonstrated in the way it treats a FLOSS project as an object that can be studied in order to classify types of activities. The objectivist ontology is also apparent in its assumption that differences in satisfaction can be measured and related to other constructs.

3.1.2 *Epistemology*

Epistemology is concerned with the nature of the sources of knowledge associated with the phenomenon of interest to the researcher (Bryman 2008, p.13). Researchers who study social issues must consider whether the approaches traditionally used in the sciences can be adapted to studies of social phenomena, or whether the subjective nature of social phenomena means that they require a different approach.

Scientists who study the so-called 'real' world are concerned with repeatable physical phenomena that can be observed, documented, and measured. In addition, the researchers making the observations are considered to be independent of the object being observed. This view, which says that knowledge comes from observations based on concrete evidence, is described by the term *positivism* (Orlikowski and Baroudi 1991). Positivism is usually associated with a search for generalisable models that can be tested and verified or disproven (Henn, Weinstein, and Foard 2006, 11–13). In addition, positivist research is usually quantitative, involving measurements and statistics (Bryman 2008, p.22).

An alternative view is *interpretivism*, which takes the view that social phenomena are fundamentally different to ones that occur in nature, because they involve individual perceptions. Interpretivism says that these perceptions, and the meanings people attach to them, are subjective, and will vary between people depending on their social and cultural background, as well as the meanings they place on objects, events, and relationships (Henn, Weinstein, and Foard 2006, p15). Social researchers who use an interpretivist paradigm acknowledge the social context of their research, and generally seek to develop a deep understanding of the ways their research participants view the world, rather than testing

specific hypotheses. Gorman and Clayton (1997, p.23) said that interpretivism is generally inductive, and is used for theory building rather than theory testing. Interpretive research is usually qualitative, involving words and descriptions, rather than measurements (Bryman 2008, p.22). In addition, it typically uses a flexible and unstructured research design that may evolve as the research progresses (Henn, Weinstein, and Foard 2006, p.15).

The positivist focus on outcomes means that it does not result in a deep understanding of individual perspectives—instead positivism seeks objective answers, while interpretivism is inherently subjective. More recently, a third epistemology has been described, known as *postpositivism*. Its origins are in the traditional positivist paradigm; Creswell (2009) said that postpositivist research is similar to positivism, since it seeks to "describe causal relationships of interest" (p.7). However, a postpositivist approach acknowledges that the evidence used for this may be imperfect or incomplete (Pickard 2007, p.10), which means that postpositivists say that it is therefore not possible to 'prove' a hypothesis (Creswell 2009, p.7). A postpositivist researcher therefore does not claim that a hypothesis is proven, but instead says that it is consistent with the evidence.

Creswell (2009, p.7) identifies the key steps in postpositivist research as:

1. Theory development;
2. Hypothesis generation;
3. Data gathering;
4. Hypothesis testing; and
5. Theory revision.

These steps represent the deductive approach that is typical of positivism, and therefore also of postpositivism. In contrast, interpretivism does not generate theory until after the data are collected, and it does not involve hypothesis testing.

These differences between positivism, interpretivism, and postpositivism suggest that the researcher's beliefs about knowledge shape the nature of the research question, and this then determines the choice of the epistemological paradigm used in a research project. This researcher's educational background includes an undergraduate degree in chemistry and mathematics, which are both positivist disciplines. However, she also recognises that all observations are incomplete, and that context may be important in conducting social research. This means that the thesis takes a postpositivist perspective, measuring the effect predictor variables have on the outcome variable 'participant satisfaction'. The central question of this thesis, *What factors influence participant satisfaction with a free/open source application software project?*, assumes

that specific factors can be identified objectively, and that their influence on satisfaction can be measured.

3.1.3 *Research methodologies*

Creswell (2009, p.4) identifies three common research methodologies:

1. qualitative;
2. quantitative; and
3. mixed methods.

Qualitative research is typically used to explore and understand problems from the research participant's point of view, and is most commonly used when the researcher has a constructivist ontology and an interpretive epistemology. It does not usually start with a theory or hypotheses, but instead looks deeply at the meaning of the data collected in order to identify themes or patterns. Qualitative research follows an inductive approach, in which theory is developed from the data collected (Bryman 2008, p.11).

In contrast, quantitative research collects measurements and examines relationships between variables, reflecting an objectivist ontology and a postpositivist epistemology. It usually follows a deductive approach, by collecting data that is intended to test a theory (Pickard 2007, p.18).

Mixed methods research generally combines both qualitative and quantitative data. In the 1970s and 1980s, this was considered to be questionable because of the differences in the main philosophical paradigms the underpin the two approaches (Creswell and Plano Clark 2007, p.15). More recently, mixed methods research has become more acceptable because researchers have recognised that using "both types of data will result in a better understanding of the research problem than one data type alone could produce" (Creswell and Plano Clark 2007, p.168). Pickard noted that mixed methods research is a good match for a postpositivist epistemology, since the qualitative data can be used to support the quantitative findings (2007, p.7). In addition, gathering both qualitative and quantitative data provides a form of triangulation, because it provides multiple sources of evidence to test the hypotheses (Leedy and Ormrod 2001, p.105).

3.2 SPECIFIC TECHNIQUES

This research used a two-stage sequential mixed methods approach to answer the research questions identified in Section 1.2 on page 3. Specifically, it followed what Creswell and Plano Clark call an exploratory sequential design (2011, p.71), since it began with a qualitative stage intended to gain familiarity with the research context, followed by a

quantitative stage to test a theoretical model. In Creswell and Plano Clark's 2007 notation, it was a qual→QUAN study, indicating that the qualitative stage was intended to assist with development of the instrument used in the subsequent quantitative stage (p.76).

Stage 1 was a two-part qualitative investigation of a purposive sample of FLOSS projects and their participants, designed to gain a deeper understanding of the research setting, in particular the types of contributions participants make to FLOSS projects. The data gathered in this stage were also used to review the preliminary research model and associated hypotheses, and to inform the development of the survey instrument.

Stage 2 involved a quantitative survey of participants involved in a range of FLOSS projects to test the research model and hypotheses. Both stages of the project gathered data from human subjects, which meant that the researcher needed human ethics approval in order to proceed. Section 3.3 on page 84 discusses the ethical issues involved in this research, as well as the consent process.

The specific techniques used in each stage of the research are discussed in the following sections.

3.2.1 *Stage 1: Qualitative investigation*

The main objectives of this stage of the research were to gain a deeper understanding of the ways in which participants in FLOSS projects interact, in order to review the preliminary research model, and to gather data to answer the first research sub-question:

> *What types of contributions do participants make to free/libre and open source software projects?*

Population

This section discusses the approach used to select an appropriate subpopulation of FLOSS projects for the research.

Section 2.3 on page 17 showed that FLOSS communities are virtual, with participants using a range of channels to communicate with each other. Williams and Liong found that recruiting subjects from online communities was problematic (2009, p.135), recommending that researchers studying a virtual community find ways of establishing a connection with their subjects through shared interests and practices. One technique used to establish this type of connection is through a shared background; this researcher has worked as a librarian, and currently teaches in a postgraduate library and information studies programme, suggesting that library-related FLOSS projects may be suitable for her.

A wide range of FLOSS projects intended for library and information managers has become available in the last 10 years. The OSS4LIB (open source software for libraries) web site (http://www.oss4lib.org/) listed

over 80 projects in September 2004, growing to over 400 projects by late 2010. These vary in size and complexity from simple Perl scripts to generate statistics from transaction logs (Ovid Statistics Log Report Generator) to library management systems (Evergreen, Koha, OpenBiblio, and PhpMyLibrary) to software for creating digital libraries/repositories (Greenstone, DSpace, EPrints, and Fedora). Some of this software is intended to be used primarily by systems librarians, while other projects have both librarian and end-user interfaces. Several of these projects have hundreds of users; there is one plug-in to a commercial library management system that has over 250 users. These characteristics suggested that this sub-population provided a suitably diverse range of projects for this research, and also had the type of connection with the researcher's background recommended by Williams and Liong (2009). Much of the literature about FLOSS projects for LIM applications is in the form of descriptive case studies, intended to inform practitioners about the software's existence and functionality. Recent examples include Darby (2006), Sessoms and Sessoms (2008), Lascarides (2009), and Helling (2010).

In addition, restricting research to particular applications or subject domains is established practice in FLOSS research. In an early example, Hertel, Niedner, and Hermann (2003) restricted their survey to mailing lists used by members of the Linux kernel community. In another, Maass (2004) studied members of the Apache Cocoon project. Crowston et al.'s 2010 review of 193 papers published between 1999 and 2006 showed that the highest proportion of empirical studies of FLOSS projects were case studies of a single project, and that studies involving high numbers of projects were based on data harvested from FLOSS repositories. They also found that Linux and Apache were the most common projects studied, followed by Mozilla and Gnome. Porter, Williams, and Weitzer (2004) say that survey fatigue, defined as "overexposure to the survey process", is a common reason for low response rates to surveys. Because of the emphasis placed on Linux, Apache, Mozilla, and Gnome in previous research, participants in these communities may be suffering from survey fatigue. By choosing to study a group of FLOSS projects that have not been studied extensively in previous research, the researcher hoped to minimise the problem of survey fatigue. However, this also may limit the generalisability and comparability of the results, which is discussed further in Section 9.4 on page 214.

Choice of sampling technique

The choice of sampling technique was determined by the purpose of this stage of the research. Four approaches to sampling were considered:

- convenience;
- snowball;

- random; and
- purposive.

In a convenience sample, respondents are selected because it will be easy to interview them (Bryman 2008, p.183). Convenience samples are unlikely to be representative of a population, and some types of respondents may be over- or under-represented (Alreck and Settle 2004, p.43). This means that the results from a convenience sample cannot be generalised to the wider population. Leedy and Ormrod suggested that convenience sampling is best suited for exploratory research (2001, p.218), while Bryman and Bell noted that convenience sampling may be used when a researcher is presented with an unexpected opportunity to gather data (2007, p.198). When the interviews for this stage of the research were being planned in mid-2006, only two New Zealand libraries had implemented FLOSS systems, and only two library-related FLOSS projects were known to have New Zealand-based developers. Although the exploratory nature of this stage of the research suggested that convenience sampling might be a suitable approach, the limited number of potential interviewees conveniently available in the researcher's home country means that convenience sampling was rejected.

'Snowball' sampling is an iterative process of developing a sample from a small group of initial key informants (Pickard 2007, p.65). These initial interviewees are asked to suggest other people for the researcher to approach for subsequent interviews. Bryman (2008, p.184) noted that this is useful when it is difficult to identify people with specific characteristics, or when it is important for interviewees to have specific relationships with each other. One example of this is when researchers want to be able to compare individual perspectives on a specific event experienced by several people, or when they want to survey family members. However, Bryman also recommended that this approach be used with caution, because it is likely to result in a sample that is not representative of the population. Bryman and Bell suggest that a snowball sample is most appropriate when it is impossible to identify a sample using a sampling frame (2007, p.200). Since this research was not intended to gather comparative data, snowball sampling was not considered suitable for this project.

In a random sample, every member of the population has an equal chance of being selected as a research participant (Alreck and Settle 2004, p.43). This requires access to a list of all individuals in the population, so that a probability sampling technique such a drawing names from a hat, or using entries from a table of random numbers to identify sample members, can be used (Pickard 2007, p.61-62). While data gathered using a random sample have the advantage that the results are more likely be generalisable to the wider population than with a non-random sample, using one for this research was not feasible because of the reasons discussed in Section 2.4 on page 25. Specifically, there

are no authoritative lists of FLOSS projects and their participants from which to draw a random sample.

Purposive sampling is a form of non-random sampling commonly used in qualitative research (Pickard 2007, p.59). Bryman (2008) described a purposive sample as "strategic" (p. 458), meaning that potential respondents are identified based on their ability to provide data relevant to the research questions. In addition, a purposive sample is intended to produce a sample that reflects different characteristics of the population (Bryman and Bell 2007, p.201). Gorman and Clayton (1997, p.127) emphasised that a purposive sample needs to include respondents with a range of characteristics that are relevant to the research project. Pickard reinforced this by suggesting that a researcher preparing a purposive sample use a simple framework of relevant characteristics before the sample is drawn (2007, p.64). The main limitation of a purposive sample is that it is not representative of the total population, and is therefore not suited for quantitative studies. However, Stages 1 and 1b of this research were qualitative, and by using a purposive sample, the researcher was able have a mix of interviewees in different roles, and from different projects. In their study of the processes FLOSS projects used to assign work to specific developers, Crowston, Li, Eseryel, and Howison (2007) used a framework of five characteristics of FLOSS projects they felt were most relevant to their research. This shows that purposive sampling has previously been used in a similar context to this research, and therefore a purposive sample was chosen as the best fit for this research project. Details of the characteristics used to select the sample are provided in Section 3.2.2 on the current page.

3.2.2 *Stage 1: Qualitative investigation*

This stage used qualitative methods to develop the participation and satisfaction measures, and to validate the preliminary model of factors that influence user satisfaction, shown in Figure 2 on page 59. A qualitative approach was selected as the most appropriate for this phase because it is best suited to the collection of in-depth data, and for the development of theories and concepts (Bryman 2001, p.284–285).

Stage 1a: Document review

This phase involved the ongoing document review of a purposive sample of 10 LIM FLOSS projects. Projects were chosen to include a range of community sizes (measured as number of sites, as far as this could be determined), project ages, application types, number of developers, software complexity, and activity levels, in order to develop a preliminary list of activities carried out by participants in the projects. These characteristics were used as the selection criteria for two reasons. First, they could be observed for all potential projects, meaning that a project

would not be excluded from consideration because of missing information. Secondly, they represented aspects of the project and its community that were known to vary, and could be selected to include as many combinations of characteristics as feasible, in order to achieve maximum variation, as recommended by Pickard (2007, p.14).

Table 8 on the next page shows the range of characteristics in the sample, and Section 4.1 on page 97 includes a description of each project and its characteristics.

For each project, documentation, email discussion archives, source code, and web sites were examined to identify typical activities carried out by project community members.

Stage 1b: Interviews

The next phase of the qualitative stage involved individual interviews, in order to validate and improve the preliminary research model shown in Figure 2 on page 59. The population from which the sample for this stage of the research was selected included the authors of articles and conference papers about LIM FLOSS projects, as well as people who participated in project-related email discussion lists and contributed to the OSS4LIB portal. This range of sources of information about project participants was used in order to develop a population for the sampling frame that was as broad as possible, given the limitations associated with FLOSS project data discussed in Section 2.4 on page 25.

A preliminary list of 57 names was drawn up as the sampling frame for this stage. Prospective interviewees were chosen from the list using a purposive sampling approach that considered gender, project role (to the extent that this could be determined), background, and observed level and type of activity.

As with Stage 1a, a purposive sample was chosen because it ensured that interviewees represented different viewpoints and experience, and were involved in a range of roles in different projects. In addition, the sample was chosen so that both genders were represented equally. The criteria used to choose the sample also ensured that it included people who were involved in different roles; in particular, the sample included people who were developers, users, and system administrators. Section 4.2 on page 102 describes the characteristics of the interviewees.

In total, 22 interviews were conducted, involving 24 people (one interviewee chose to involve two co-workers in the interview). The interviewees represented six different LIM FLOSS projects—at this point, analysis of the interview data showed that theoretical saturation had been reached, and that no significant new insights were likely to be gained by further interviews (Bryman 2001, p.303).

The interviews used open-ended questions to gather qualitative data focusing on types of participation, satisfaction with FLOSS software, and the constructs included in the preliminary research model. The main

Table 8: Summary of project sample characteristics

CHARACTERISTIC	VALUE	NUMBER OF PROJECTS
Number of sites	Low: fewer than 100	3
	Medium: between 101 and 249	3
	High: more than 250	4
Project age	Less than 5 years	1
	5-10 years	6
	More than 10 years	3
Application type	Integrated library system	3
	Institutional repository	2
	Digital library	1
	Record manipulation utility	1
	Electronic resource management	1
	Web portal	1
	Journal publishing	1
Number of developers	Low: 1-5	4
	Medium: 6-20	5
	High: More than 20	1
Complexity	Low: 1-5 data types	1
	Moderate: 6-10 data types	5
	High: more than 10 data types	4
Activity level	Low: less than one message per week/one release per year	3
	Moderate: 2-5 messages per week/two releases per year	5
	High: more than 5 messages per week/more than two releases per year	2

advantage of using semi-structured interviews for this stage was that all interviewees were asked the same basic questions, but the interviewer could probe for further elaboration, if the original answer was brief (Bryman and Bell 2007, p.474–475). The face-to-face interview guide is included as Appendix E on page 235. The five questions were based on the key components of the preliminary research model, covering:

1. Interviewees' background, including their education and skills;
2. The FLOSS project they were using or contributing to;
3. Interviewees' relationship with the project, including their own and other participants' contributions;
4. Interviewees' satisfaction with various aspects of the project; and
5. An opportunity to comment on anything else they wanted to mention.

Since the LIM FLOSS community is global, with projects and participants based in North America, Europe, and Australasia, email was used as the main interview communication medium, with face-to-face and telephone interviews being done when possible. The advantage of using an asynchronous email technique was that respondents did not need to be in a compatible time zone, unlike synchronous techniques such as face-to-face, phone, or chat (instant messaging). This approach also allowed respondents to reply at their convenience (Mann and Stewart 2003: 82). However, asynchronous email interviews also have disadvantages, such as the inability to see an interviewee's body language, and increased difficulty in building rapport between the interviewer and the interviewee (Bryman 2008, p.643). Data obtained from email interviews may be more formal than that from face-to-face ones, and responses may be more considered, since interviewees have had more time to consider the questions. However, since most participants in FLOSS projects communicate with each other using asynchronous channels such as email, using email interviews was considered a suitable technique for this research.

For the email interviews, the interview questions were broken into three groups of questions, with each group sent in a separate message over a period of 2–3 weeks. Appendix H on page 243 includes a copy of this version of the interview guide. The questions were included as part of the body of the message and interviewees answered them using the 'reply' command. This approach was selected to increase people's willingness to participate, as it involved a smaller amount of time for each reply than if all of the questions were sent at once in a single message. By ensuring that the interaction took place over a longer time period than a single interview, the email exchanges took the form of an extended conversation, and had a degree of informality. However, one interviewee, known to be a frequent user of instant messaging,

asked to be interviewed using that medium rather than email, and this interview was therefore conducted at a mutually convenient time. The email interviews took place between 24 November 2006 and 9 March 2007.

The face-to-face interviews were held between 29 November 2006 and 28 February 2007. They ranged in duration from 25 to 65 minutes, with a median of 45 minutes. There was a general correlation between the length of time taken for the interview and the length of time the interviewee had been involved with the project; overall, the greater the interviewee's experience with the project, the more they had to say. The same pattern was apparent in the email interviews: people with more experience tended to give longer answers, and were generally aware of more aspects of the project than people who were relatively new to the project.

All interviewees were initially invited to participate in the research project by email, and were asked to reply if they were willing to be interviewed. A copy of the introductory email messages are included in Appendices B on page 229 and F on page 237.

Face-to-face and telephone interviews were recorded on a digital recorder, and subsequently transcribed. The researcher did the transcription personally, which meant that she was able to reflect on each interview as she was transcribing it, and became more conscious of recurring themes and patterns. A copy of the interview transcript was sent to each interviewee as soon as possible after the interview, so that they could check it for accuracy as recommended in Creswell and Plano Clark (2011, p.211).

Two interviewees made minor corrections to the transcripts of their interviews, primarily to names of projects and people. One interviewee sent a lengthy response, expanding the answers to some questions, and providing more background detail to explain the initial responses. This interviewee provided brief answers during the interview, and had did not responded to further probing about during the interview. She may have been more comfortable providing the additional detail in an asynchronous mode, rather than in the face-to-face context of an interview. These additional data were incorporated into the transcript before it was analysed, following the practice outlined in Pickard (2007, p.179). The email and single instant messaging interviews had the advantage of being 'self-transcribing', and therefore email interviewees were just sent a final 'thank you' message.

3.2.3 *Stage 2: Quantitative survey*

The second stage of the research involved developing and administering a quantitative, web-based survey to test the model and hypotheses.

Quantitative methods are best suited for collecting measurements for statistical testing (Bryman 2001, p.20).

This type of data was needed to test the model and hypotheses, and provide an answer to the primary research question, *What factors influence participant satisfaction with a free/libre and open source application software project?* These data were also used to answer the second subquestion *Do the factors that influence satisfaction vary for different types of participation? If so, in what way?*

A self-completion web-based survey was chosen as the most appropriate method of reaching the target population for this stage of the research, since potential respondents could be located anywhere in the world. Other advantages of web-based surveys are that they are low cost, since they do not involve printing or postage costs, and they save time (Yun and Trumbo 2000). In addition, they reduce coding errors, since respondents are in effect coding the data themselves when they choose responses from a pre-defined list (Umbach 2004). Finally, it is possible to reach a wider population of potential respondents with a web-based survey, since invitations to complete it can be sent to email discussion lists as well as to individual email addresses. This approach is particularly suitable when it is difficult to develop a complete sampling frame for the target population. This was the case for this research, since FLOSS users are not required to register in order to download and install the software, meaning that it was not possible to develop a list of all participants in the selected projects. The drawbacks of web-based surveys include coverage error, measurement error, and non-response error (Umbach 2004, p.25).

Coverage error occurs when the sample is not representative of the target population; this generally occurs when email addresses are missing or out-of-date, causing some members of the population to be omitted from the sample frame. To minimise coverage error, the invitation to complete the web-based survey for this research was sent to email discussion lists, rather than individuals, since most people keep their email discussion list subscription addresses up-to-date.

Measurement error occurs when respondents choose an inaccurate response that does not reflect their opinion. It can happen when response options are inappropriate or confusing, or when the appearance of the survey changes due to the use of a different operating system or web browser (Umbach 2004, p.26). Development of the survey, discussed in more detail in Section 3.2.3 on the following page, followed Umbach's recommendations to minimise measurement error; in particular, it used radio buttons that allowed only a single response when appropriate, and the researcher tested the survey instrument on multiple operating systems and Web browsers.

Non-response errors occur when the people who respond to a survey are not representative of the total population, introducing bias in the survey results. Umbach says that non-response bias is a particular prob-

lem when potential respondents have different levels of technical ability or varying levels of access to the Web (2004, p.27). Since all FLOSS projects are web-based and their participants use web-based tools to communicate with each other, this concern does not apply to the target population.

Because the potential respondents could be based in different countries, using a structured interview, where the researcher administers the survey instrument and records the responses, was considered impractical. This approach was also likely to limit the number of respondents, since it would require them to be in a compatible time zone due to the synchronous nature of this technique.

Other research methods that provide data suitable for hypothesis testing, such as conducting a laboratory or field experiment, were not appropriate for the proposed research. In discussing experimental research, Bryman says "manipulation, then, entails intervening in a situation to determine which of two or more things happens to people" (2008, p.36); it would be impractical, and also unethical, to artificially manipulate characteristics of free/open source software projects to measure changes in participants' satisfaction.

Stage 2a: Developing the survey instrument

The results of Stage 1 were used to review the conceptual model and develop a web-based survey to test it. Where possible, existing scales were used or modified for the model's constructs, since these were known to have appropriate reliability and validity. New scales were developed for the product and process openness constructs, since these had not been previously used in empirical research. Extent of participation was measured in two ways, first by asking respondents what their current role in the project was, and second by asking them to identify specific activities they had carried out as part of their participation.

The survey was created using NSurvey software (version 1.8.0.0) hosted on a School of Information Management server. This software was released under the GNU General Public License version 2.0, which meant that it qualified as FLOSS, a characteristic that was expected to be very important to some potential respondents, and therefore influence their willingness to complete the survey. By choosing survey software that met the FLOSS definition, the researcher showed that she respected the beliefs of potential respondents who shared Stallman's view that software licensing had an ethical component. Failing to do this could have meant that these people would not complete the survey on principle, introducing non-response bias.

The survey design followed the best practices for a web-based survey outlined by Umbach (2004). It was designed as a single, scrollable page, to ensure that respondents could read through the complete survey before beginning it, to minimise non-response bias. The survey did not

capture information identifying respondent's IP addresses, use cookies to record visits, or include a counter that measured visits, since Umbach said that all of these can cause people to refuse to complete a survey, and therefore increase non-response bias. To minimise measurement error, it used pre-coded radio buttons allowing only a single response, where possible. Umbach recommended avoiding drop down boxes, which increase the length of time required to complete a survey. The survey included only one drop down box, for country of residence. All questions were optional. Though this had the potential to reduce the overall quality of the data, since respondents could omit a significant number of questions, forcing them to answer questions would also have had consequences. First, it may have made people reluctant to complete the survey, thereby increasing the number of non-responses. Secondly, it could have caused them to choose inappropriate responses, resulting in measurement error (Umbach 2004).

The researcher tested the survey on computers running Mac OS X and Windows XP, using the Safari (Mac OS), Internet Explorer (Windows XP), and Firefox (both platforms) web browsers, to ensure that there were no significant differences in the way the survey appeared. Finally, the survey was kept short, so that most respondents could expect to complete it in 15 to 20 minutes, which was slightly longer than Porter's recommendation of 13 minutes as the optimal maximum length for a survey (2004).

A copy of the final version of the survey is included as Appendix K on page 255. As noted above, existing measures were used for the various constructs, with minor changes to wording to make them suitable for the research context.

The first section of the survey asked for information about the respondent's background, including demographic information about age (Q1), gender (Q2), highest educational qualification (Q3), and country of residence (Q4). This section also included questions about the length of time the respondent had been using a computer, measured in years (Q5), and their self-assessed level of knowledge and skills in a number of areas (Q6), using a rating scale ranging from 'minimal' to 'extensive'.

The next section of the survey asked about participant's attitude toward and use of free/open source software, at work and/or at home. By including home use of FLOSS, the survey design acknowledged that their employer's policy might restrict their ability to use FLOSS at work, but it allowed respondents to express a preference in a situation where they would have more control over the decision.

The third section asked respondents to choose one library or information management free/open source software project they used or were involved with in some way. If they were involved with more than one project, they were asked to choose the one that they had used or contributed to most recently. The questions in this section asked them to:

- name the project (Q13),
- say how long they had been using or contributing to the project (Q14),
- name their current role (Q15),
- identify other roles they had held in the project (Q16),
- indicate how much time they had spent per week working with or contributing to the project in the last six months, on average (Q17),
- indicate how much of this time had been part of their paid employment (Q18),
- indicate which activities they had carried out with the software (Q19 and 20),
- identify training they had received that related to their use of the software (Q21 and 22),
- indicate their satisfaction with various aspects of the project (Q23),
- compare the extent of their experience of using the software relative to their perception of other people involved with the project (Q24),
- indicate their views on the developers' communication styles (Q25),
- indicate their perceptions of the project's culture (Q26)
- indicate the amount of influence they have had on the software, either locally or on the shared version (Q27 and 28)
- assess the complexity of the software and the work they do with it (Q29).

The final question was open ended, and gave respondents an opportunity to make further comments on the project, or reasons for their satisfaction or dissatisfaction with it.

Stage 2b: Pilot study

The survey instrument was tested with a pilot study involving six respondents selected from non-LIM FLOSS projects. Hair, Babin, Money, and Samouel, (2003) recommended that a pilot study could involve as few as four people if most questions had been previously validated, which was the case for this research, since many of the scales were adapted from previous research. The main purpose of the pilot study was to improve the survey's format, questions, and scales, and to determine how long it was likely to take respondents to complete it. Pilot study respondents

were asked to complete the survey and comment on its length, wording, and structure. Only question 17 generated feedback from more than one person. It was reworded and rechecked with the people who commented on it until they reported that its meaning was clear.

Since feedback from the pilot study related to just one question, and most of the questions were based on previously used questions, it was decided not to undertake a formal pretest. In their discussion of the use of pretests, Presser et al. (2004) said "we have much better tools for diagnosing questionnaire problems than fixing them" and report that after using three different methods to pretest 83 questions, and subsequently revising 12 of them, "the revised questions generally did not appear to outperform the original version" (p. 124). In other words, while pretesting is good at identifying questions that may pose problems for respondents, it does not lead to any improvement in the questions themselves. This supports the decision to rely solely on the responses from the pilot study, which provided consistent feedback.

Stage 2c: Web-based survey

The last data collection stage consisted of a self-administered web-based survey to gather quantitative data to test the model and hypotheses. A copy of the survey is included as Appendix K on page 255.

POPULATION AND SAMPLE As discussed in Section 3.2.1 on page 71, the research selected participants in LIM FLOSS projects as its population. The technique of harvesting individual email addresses from email discussion list messages in order to issue individual invitations to participate, which has been used in previous FLOSS survey research (see Sagers 2004 for an example), was considered, but rejected because it would result in a bias toward active project participants. Neilsen (2006) claimed that roughly 90% of participants in most virtual communities are lurkers, spending their time observing rather than interacting with other community members. Therefore the approach of issuing an open invitation to complete the survey was chosen, because it was better suited to the proposed research, which sought responses representing different types of participation. This approach assumes that people who have voluntarily joined an email discussion list do so because they have an interest in the project, but their presence on the list does not imply that they all have similar roles or involvement with the projects. In most cases this group is likely to include the project's initiator/owner, its core developers, and regular users of the software, as well as other interested parties.

In order to attract respondents who were involved with different projects, an invitation to complete the survey was sent to the email discussion lists for three active FLOSS library projects, the Open Source for Libraries email list, and the Code4Lib email list. The text of the

invitation email message is included in Appendix J. Once the initial invitation was 'out in the wild', it was more widely disseminated by third parties who had an interest in the research topic. After receiving the initial invitations, several people offered to repost it to other relevant email lists, including another library FLOSS project-related list, and a list for Indian librarians with an interest in technology. In addition, a number of library-related bloggers posted the invitation on their blogs, including Michael Stephens' Tame the Web, Nicole Engard's What I Learned Today, and Meredith Farkas's Information Wants to be Free. This publicised the survey to a range of communities, and potential respondents representing a variety of backgrounds, projects, and nationalities. The survey was available from 16 October 2008 through to 14 November 2008.

The main advantages of this approach were that the survey was simple to administer, and respondents could complete it at their convenience. The disadvantages were that it was not possible to control who answered it, and the respondents were likely to be people who are interested in the topic of the research, which could bias the results.

The target number of responses was determined by the number of constructs in the model. Green's rules of thumb (1991) were used to estimate this number. To use multiple regression with a model having N predictors, the minimum sample size is $50 + (8 * N)$; based on the research model, this meant a minimum of 122 usable responses. For testing the relationships of the individual predictor variables to the outcome variable, the minimum sample size is $104 + N$, which is 113. Since the research is intended to test both of these, the larger value (122) applied.

3.3 ETHICAL CONSIDERATIONS

Both stages of the research involved human subjects, and ethical permission was required from the School of Information Management Human Ethics Committee. Separate applications were submitted for Stages 1b and 2c, and are included as Appendices A on page 221 and I on page 245.

In Stage 1b, potential interviewees were initially invited to participate in the research by phone or email, depending on their location. If they indicated that they were interested in being interviewed, they were sent a copy of the project information sheet. Face-to-face interviewees were also asked to sign a consent form, while the phone, email, and IM interviewees were asked to indicate that they were aware of the conditions for the research, which were posted on a web page. The URL for this page was included in all of the email messages sent to interviewees, to ensure that they remained aware of the conditions during the interviews. Copies of these documents are included as Appendices B on page 229 through G on page 239. Interviewees were told that their identity would

be kept confidential, and that the thesis and any publications resulting from the research would not include any information that would identify them.

Respondents to the web-based survey in Stage 2c viewed an electronic version of the information sheet at the beginning of the survey. The survey was designed to be anonymous, and no data that would identify individuals were collected.

A consequence of these guarantees for privacy is that this thesis does not associate individuals with specific projects, and the projects with which the interviewees were involved are not named, to prevent people familiar with the projects from speculating about who the interviewees were. Since the purpose of the research is not to measure respondents' satisfaction with specific projects, but rather to develop and test theory about general factors affecting this, naming individual FLOSS projects is unnecessary.

3.4 DATA ANALYSIS

Both phases of Stage 1 gathered qualitative data, and content analysis was used to identify themes and patterns. For the preliminary analysis, manual coding was used to identify key phrases in the interview transcripts for each of the main constructs, such as educational background, activities, roles, and satisfaction. A photograph of some of these working documents is included as Appendix L on page 265.

This stage of the research had a relatively narrow focus, first to identify types of participation, in order to answer the first research subquestion, *What types of contributions do participants make to free/libre and open source software projects?*; and second to validate the preliminary conceptual model presented in Section 2.9 on page 58. Using qualitative analysis software such as NVIVO was considered for the analysis, but rejected in favour of traditional manual coding and categorisation. Pickard noted that this type of software is best suited for large-scale, complex qualitative projects, because of the time required to learn to use the software effectively (2007, p.281). In addition, she said that the researcher still needs to do the coding, analysis, and categorisation, whether or not analysis software is used. Since the number of interviews was relatively small, and the purpose of this stage of the research was clearly defined, manual coding was considered to be the better approach.

3.4.1 *Stage 1a: Development of participation construct*

Specific types of contributions to FLOSS projects were identified by reading individual postings to selected email discussion lists, and from project web sites. These were listed in an OpenOffice Calc spreadsheet, and then assigned to categories based on the general purpose/object of

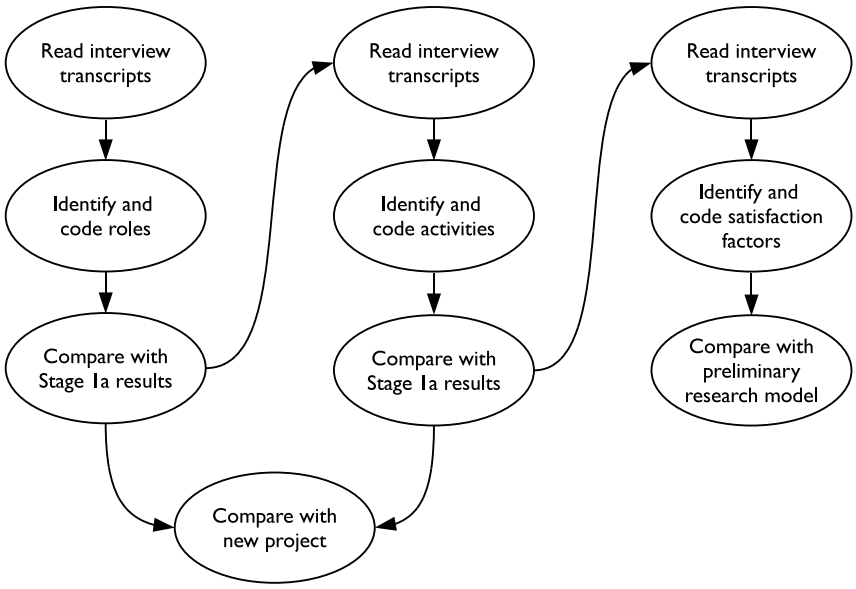


Figure 3: Stage 1b analysis flow diagram

the activity (such as code, community, resource). In addition, a list of project roles was developed from the small number of projects that had a more formal organisation structure. Sample pages from the spreadsheets are included as Appendices L on page 265 through N on page 269.

3.4.2 *Stage 1b: Validation of preliminary model and the participation construct*

The data from the interviews were analysed using the approach outlined in Bryman (2008). Transcripts from the synchronous interviews and the text of the email and IM interviews were first read through several times, in order to gain an overall impression of the main points made by the interviewee. They were then analysed question by question. Each response was first coded to identify roles and activities; these were then compared to the lists of roles and activities developed from the Stage 1a data analysis. Once all the transcripts had been coded, a master list of project activities was prepared in an OpenOffice Calc spreadsheet, and the activities were then grouped by broad theme. Figure 3 on this page illustrates the process.

The main goal of this stage of the analysis was to identify themes that were discrete with minimum overlap, so that each activity could be clearly placed in a single main theme. A similar approach was taken to identify key project roles. To confirm that the themes were as complete as possible, once the main themes were identified, a new project, outside the original sample, was observed independently, and its activities classified by theme. No gaps were identified, though the distribution of activities in this project followed a different pattern.

The next stage of the qualitative analysis focused on satisfaction with the FLOSS project, and again the transcripts were read through several times before this aspect was coded. This part of the analysis also involved looking for patterns in responses, and comparing responses from different interviewees, in order to test the preliminary model and identify any gaps.

Once this analysis was complete, the research model was reviewed to take into account the new insights gained from this stage of the research, and the results used to design the quantitative survey used in Stage 2c. The results of this review are discussed in Section 7.1 on page 137.

3.4.3 *Stage 2: Quantitative survey*

Data were exported from NSurvey in XML format. Using a combination of XSLT and a simple PHP program, the XML file was converted to a comma-delimited format; this was imported into OpenOffice Calc and the resulting file was then transferred to PASW (formerly SPSS) version 18 for further analysis.

The initial approach to analysing the survey data used descriptive statistics (frequency and per cent) for the ordinal and nominal data, and content analysis for the free text questions. Appendix Q on page 275 lists each question in the survey, its description, the measurement scale used, and the initial analysis technique. The following section summarises the way each construct was measured.

Construct measurement

When they were available, existing scales from previous research were used to measure each construct, since their reliability and validity had already been determined. Minor changes were made to the wording to suit the context of this study.

Each construct in the research model was measured using multiple indicators. Bryman (2008, p.148) identified three reasons for this. First, using more than one measure increases the reliability of the data, since a single indicator might involve considerable measurement error. Second, using only one question may not adequately represent the construct being measured, or may be too general. Finally, using more than one measure allows finer distinctions to be made between respondents, since more combinations of responses are possible. The response scales in the survey followed Hinkin's 1998 recommendation to offer five Likert-type options, since scales with more items have not been shown to result in increased reliability.

Strictly speaking, the data from Likert-type scales are ordinal, which restricts the types of statistical tests that can be used with them. However, Bryman and Cramer said that it is now common practice to treat multiple-item Likert-scales as interval, since the resulting values are not

restricted to a small number of categories (2005, p. 70). This research followed their recommendation and treated the summary data that was derived from the multiple item scales as interval.

The following section lists each construct in the order it appeared in the survey instrument, and identifies the main changes that were made to existing measurement scales.

PARTICIPANT KNOWLEDGE AND SKILLS (QUESTION 6) This scale was based on Torkzadeh and Lee's 2003 scale, adapted for a FLOSS context. It consisted of eight of the original 12 items, omitting the items that related to designing screens, output formats, and using mainframes. Torkzadeh and Lee found that the scale had a Cronbach's alpha of .95. Hulin, Netemeyer, and Cudeck (2001) suggested that such high values of alpha may occur when the number of items in the scale is very high, or when there are items that are too similar in meaning. Though they did not specify an absolute cut-off value, they recommended using commonsense in reviewing scale items if they result in a high value of alpha, which is what was planned for this scale.

EXTENT OF PARTICIPATION (QUESTIONS 19 AND 20) This was developed for this research, since existing scales to measure participation, such as the one in Guimaraes, Staples, and McKeen (2003), assumed that system development followed a structured process. This is not necessarily the case for a FLOSS project, where the development process has been characterised as a "bazaar" model (Raymond, 2001). This scale was based on the most common activities identified by the participants in the Stage 1 interviews. It consisted of 20 items, covering both user and developer activities, with a dichotomous (Y/N) response scale. An open-ended question was included to give survey respondents an opportunity to add any other types of contributions they had made to the project.

PARTICIPANT TRAINING (QUESTIONS 21 AND 22) This was based on Nelson and Cheney's 1987 proposal, which was used by Guimaraes, Staples, and McKeen (2003). They found that the scale had a Cronbach's alpha of .83. This research used four of Guimaraes, Staples, and McKeen's five items, omitting the item relating to in-house company courses, which was not considered relevant to the present study. Since training has not been studied extensively in the context of FLOSS project participants, an open-ended question was also included to give survey respondents an opportunity to indicate any other types of training they felt were important.

PARTICIPANT SATISFACTION (QUESTION 23) This question asked respondents to assess their satisfaction with 12 characteristics of a FLOSS project that the Stage 1b interviewees identified as important, us-

ing a measurement scale ranging from 'not at all satisfied' to 'completely satisfied'. The wording of this measurement scale was chosen to minimise positivity response bias, since early satisfaction research showed that a more normal distribution of responses occurred when all options included the term 'satisfied', using qualifiers like 'not' or 'somewhat' (University of Minnesota, Department of Psychology 1967). In contrast, using a scale from 'Very dissatisfied' to 'Very satisfied' resulted in a strong bias to the upper end of the scale, with most responses between 'Satisfied' and 'Very satisfied'. Since one of the assumptions of multiple regression is that the predictor variables are unconstrained, this type of positivity bias would have been a problem (Field 2009, p.220).

PARTICIPANT EXPERIENCE (QUESTION 24) This scale consisted of four items that asked respondents to rate their experience relative to their peers, adapted from Guimaraes, Staples, and McKeen (2003). They found that the scale had a Cronbach's alpha of .84. The measurement scale ranged from 'Considerably less than most' to 'Significantly more than most'.

DEVELOPER COMMUNICATION QUALITY (QUESTION 25) This scale was based on the Communicator Competence scale originally developed by Monge, Buckman, Dillard, and Eisenberg (1983). It used nine of the original items, omitting the ones involving listening and speaking, which were not considered to be relevant to the virtual context of a FLOSS project. Guimaraes, Staples, and McKeen (2003) also used this scale, finding that it had a Cronbach's alpha of .98. Hulin, Netemeyer, and Cudek's (2001) cautions were also planned for this scale, if its alpha was found to be this high. The measurement scale ranged from 'Strongly disagree' to 'Strongly agree'. Participants could also choose an 'n/a' option.

PROCESS AND PRODUCT OPENNESS (QUESTION 26) These scales were developed for the current study. Process openness was measured using five items, while Product openness involved four. The measurement scale ranged from 'Strongly disagree' to 'Strongly agree'. One item was reverse coded, meaning that a low score on the scale indicated a more open process and vice versa.

PERCEIVED INFLUENCE (QUESTIONS 27 AND 28) Previous studies of the relationship between influence and satisfaction have been done in the context of in-house software development, and so the scale used was not appropriate for a FLOSS context. This construct was measured with two questions, asking participants about the extent of their influence locally, and on the overall project.

SOFTWARE AND TASK COMPLEXITY (QUESTION 29) This research adapted Tait and Vessey's 1988 scale for system complexity for a FLOSS context. It consisted of two items asking about the complexity of the software requirements and design. Tait and Vessey found that the scale had a Cronbach's alpha of .70, and McKeen, Staples, and Wetherbe (1994), who also used the scale, found that it had a Cronbach's alpha of .65 . Task complexity was measured with three items from Rizzo, House, and Lirtzman's 1993 scale, which was first published in 1970. In their 1994 study, McKeen, Staples, and Wetherbe found that this scale had a Cronbach's alpha of .75.

Specific techniques

OPERATING SYSTEM USE Data entered as free text in questions 9 and 11 were initially analysed using content analysis, in order to identify suitable categories for further analysis. In particular, names of specific versions of operating systems were translated to six new dichotomous variables to show whether the respondent used Windows, Mac OS, and GNU/Linux (for work use and for personal use); similarly, two new variables were set up to identify the total number of operating systems used by the respondent, coded as one (1), two (2), or more than two (3). Finally, two more new variables were created to categorise the respondent's work and personal operating system use as 'proprietary only' (1), 'FLOSS only' (2), or 'mixed' (3).

PARTICIPANT ROLE Questions 15 and 16 related to respondent's current and previous roles in the project. Because there were many possible variations in roles between different projects, these were both free-text fields to allow respondents to enter a term that was meaningful to them, rather than using predefined roles that would limit their responses to the researcher's perspective. The values entered by respondents were listed and compared, and two new variables were set up to categorise respondents as 'Developers/Non-developers' and 'Users/Non-users'. Respondents were assigned to the 'Developer' category if their current or previous role included working with source code, and to the 'User' category if they identified a role that indicated they were a user of the software. Examples of roles assigned to 'Developer' included developer, maintainer, and interface designer. Examples of roles assigned to the 'User' category included user, project manager, and trainer.

CONSTRUCT RELIABILITY The guidelines provided by de Vaus (2002), Bryman (2008), and Field (2009) were used to assess the reliability of each measurement scale. Table 9 on the next page shows the initial approach to measuring the constructs in the research model.

The internal reliability for all constructs except Extent of participation and Influence was tested using Cronbach's alpha, a widely-used

Table 9: Construct composition

CONSTRUCT	QUESTION(S)	NO. OF ITEMS	RANGE	CALCULATED AS
Skills and Knowledge	6	7	1–5	Mean
Extent of participation	19	20	0–20	Count of 'Yes'
Training	21	4	1–5	Mean
Satisfaction	23	12	1–5	Mean
Experience	24	4	1–5	Mean
Developer communication	25	9	1–5	Mean
Project openness	26 a–e	5	1–5	Mean
Product openness	26 f–g	2	1–5	Mean
Influence	27/28	2	1–5	Maximum
Software complexity	29a–b	2	1–5	Mean
Task complexity	29c–e	3	1–5	Mean

statistical technique for assessing the degree of correlation between responses to a set of questions. This technique could not be used for Extent of participation and Influence, because of the way these constructs were measured. Extent of participation was measured as the total of the 'yes' responses for 12 different activities, and Influence was measured as the maximum of respondents' answers to questions 27 and 28.

The measurement of a construct was considered to have adequate reliability if the value of alpha was 0.70 or higher (Bryman 2008, p.151). The first step involved using the PASW Scale Reliability function, which calculates Cronbach's alpha and item–total correlations for a set of items, and also indicates what value alpha would have if each item was removed from the scale. When this indicated that alpha would increase significantly, the item indicated was considered for removal from the scale. In addition, the item–total correlation was checked to ensure that all values were higher than 0.30 (Field 2009, p.678). Correlations less than 0.30 indicate that the item does not correlate well with the other items in the scale, and may measure a different underlying construct. Hulin, Netemeyer, and Cudeck (2001) suggest that very high values of alpha may occur when the number of items in the scale is very high, or when there are items that are too similar in meaning.

Once the reliability of the preliminary constructs had been tested, factor analysis was used to confirm that the items making up each construct mapped to the expected single factors. Items which did not load as

expected were dropped from the construct, and alpha was recalculated for the revised construct.

Once the components of each construct were confirmed, the arithmetic mean of the questions relating to each construct was calculated for each respondent and used in subsequent calculations. The mean was used rather than the total so that missing data did not skew the results. While there are other approaches to dealing with missing data, such as deleting cases or imputing values based on sample or group means (de Vaus 2002, p.67), using the mean is a standard approach when constructs are measured with more than one item, since it automatically adjusts the scores based on the number of valid answers (de Vaus 2002, p.129).

MODEL TESTING The theoretical model being tested in this research has a single outcome variable and multiple predictor variables, which means that hierarchical multiple regression was the most appropriate statistical technique for testing the model (Hair, Black, Babin, Anderson, and Tatham 2006, p.176). This approach tests the extent to which the predictor variables contribute to the variance (R^2) in the outcome variable.

Correlations were first calculated between each of the predictor variables and the outcome variable. Pearson's r was used to measure the correlation between pairs of variables.

Stepwise hierarchical regression was used to allow the order of the predictor variables to be specified; the variables were selected based on the amount of variance they explained, starting with the variable that resulted in the highest value of R^2. A regression coefficient indicates how much effect one variable has on another (de Vaus 2002, p.279). It can be expressed using the formula $Y = a + bX$, where X is the predictor variable, Y is the outcome variable, a a constant, and b the slope of the regression line. Where there is more than one predictor variable, the equation takes the form $Y = a + b_1X_1 + b_2X_2 + b_3X_3$ etc. The value of b_i indicates direction and degree of influence the predictor variable X_i has on the outcome variable Y, and it reflects the underlying units used to measure the predictor variables. The beta (β) value, which represents a standardised coefficient calculated in standard deviation units, is more useful for comparing the amount of influence different predictor variables have on the outcome variable (de Vaus 2002, p.380). The main advantage of the stepwise approach to multiple regression is that it results in a parsimonious model, including the smallest number of predictor variables needed to explain the maximum variance in the outcome variable (Hair et al. 2006, p.249).

Moderated regression analysis was used to test the effects of the hypothesised moderator variables; the calculations involved are discussed in Section 7.5.3 on page 179.

STATISTICAL SIGNIFICANCE The results of the regression analysis were considered in terms of their statistical significance. Statistical significance is the likelihood that the results represent a real effect in the sample, rather than occurring due to chance (Field 2009, p.49). By setting a predefined level at which the results are considered to be significant, researchers seek to minimise false positives, or Type I errors. Type I errors occur when the effect is *not* present in the population, but the results suggest that it is. The standard technique for minimising the occurrence of false positives is to set a high confidence level (usually 95%) before saying that the results are statistically significant (Field 2009, p.51). This means that there is a .05 probability that the results will occur by chance. This research set its confidence level at 95% (p=.05) when discussing the results of the statistical tests. Only results that had a .05 probability, or lower, were reported as significant. However, this threshold does not eliminate the possibility of false negative, or Type II results, since some effects might be present in the sample, but at a level too low to be detected.

In order to minimise the possibility of missing small effects, a power analysis was conducted. This is a statistical technique that assesses whether the sample size is large enough to avoid false negatives (de Vaus 2002, p.181). The magnitude of the effect size, which is the amount of variance explained by the model, is important in a power analysis, because larger samples are needed to detect small effects. In contrast, a large effect may be detected using a small sample (Field 2009, p. 56). Statistical power ranges from 0 to 1; Field recommends a minimum power level of .80 in order to be confident that a result minimises the probability of false negatives as well as false positives (2009, p.58). The most common approach to determining the statistical power of a sample is to use a standard table showing the sample sizes needed for different effect sizes at a range of confidence and power levels. Ellis provided a table showing that at a power level of .80, the minimum sample size is 3,137 to detect an effect size of .05, but only 29 for an effect size of .50. He also noted that the minimum effect size that could be detected with a sample size of 10 was .70, but this value dropped to .14 with a sample size of 300 (2010, p.64). If the likely effect size is known in advance, then it is possible to determine the minimum sample size needed to be confident it will be detected. However, the conceptual model for this research is new, and has not been previously tested with empirical data. This means that it was not possible to estimate the effect size in advance, in order to determine the sample size required for a power level of .80. Therefore a retrospective power analysis was conducted once the value R^2 had been determined, based on the tables in Ellis (2010, p.62, 140), since in multiple regression, the effect size is the value of R^2.

3.5 RELIABILITY

Reliability is defined as the extent to which research results are consistent and free from measurement error or noise (Ruane 2005, p.67-71). Different techniques are used to determine reliability in qualitative research and quantitative research. In qualitative research, internal reliability relates to the extent to which members of a research project agree about the meaning of the data, while in quantitative research, internal reliability relates to the extent to which the measurement items are consistent with each other (Bryman and Bell 2007, p.163, p.410). Because of this, the two stages of this project used different techniques to ensure that their results are reliable.

In the first, qualitative stage, the main technique used to increase the reliability of the results included reviewing documents and interviewing participants from a range of projects, and then comparing the results of the analysis to ensure that they applied to more than one project. In addition, the researcher made an effort to get to know members of the local free/open source community, and attended local Software Freedom Day activities. Gorman and Clayton (1997) referred to this as "immersion in the context" (1997, p.59) and said that it offers researchers opportunities to observe phenomena at different times and in different places. This helped establish reliability by providing an alternative perspective on the data, as the researcher gained familiarity with the wider context of the research.

In Stage 2, the internal reliability of the measures was calculated using Cronbach's alpha, as discussed in 3.4.3 on page 87. Scales that were significantly below the recommended threshold value of 0.70 were dropped from the analysis.

3.6 VALIDITY

Validity is defined as the extent to which the research measures what it claims to measure (Bryman and Bell 2007, p.164). In qualitative research, Lincoln and Guba recommended credibility as one way of determining the validity of qualitative research (1985, p.296). Credibility is achieved by using multiple accounts of the phenomenon being studied, complemented by respondent validation and triangulation (Bryman and Bell 2007, p.411). Respondent validation involves providing people who have provided qualitative data with the preliminary research findings, to see if there is a match between the findings and the participants' perspectives. Triangulation involves gathering different types of data about the phenomenon being studied, and comparing the results of the analysis. In quantitative research, validity includes face validity, and construct validity (Bryman and Bell 2007, p.165). Face validity measures the extent to which the meaning of the construct is reflected in

the measures, and is essentially a subjective judgement (Ruane 2005, p.63). Construct validity is measured by determining the extent to which the construct has the expected relationships with other variables. One technique used to determine this is factor analysis.

Factor analysis is a statistical technique that analyses correlations between items to identify communalities between items, which de Vaus defined as the variance they have in common (2002, p.137). Hair, Black, Babin, Anderson, and Tatham said that factor analysis helps define the "underlying structure among the variables" (2006, p.104), and they noted that it helps identify which items are can be combined for use in subsequent multivariate analyses. Specifically, factor analysis tests the individual items for convergent validity, and the constructs for discriminant validity. Hair et al. define convergent validity as "the degree to which to measures of the same concept are correlated" (2005, p.137). If the items for a construct exhibit convergent validity, they will be grouped together in the factor analysis. Discriminant validity tests the amount of correlation between different factors, which should be low (Hair et al. 2005, p.138). High correlations between different factors suggest that the factors overlap, and may not be clearly defined.

As with reliability, the two stages of this project used different techniques to establish the validity of the results.

Stage 1 used a combination of document review for selected projects and semi-structured interviews. This provided a form of triangulation, which allowed the results of the document review to be compared with the results of the interviews. Bryman notes that this combination can increase researchers' confidence in their findings (2008, p.379). In addition, having the face-to-face interviewees review the transcripts of their interviews provided a form of respondent validation, which Bryman says can ensure that they were recorded accurately (2008, p.377), and increase the creditability of the research. In addition, the preliminary findings of this stage were discussed with members of the local FLOSS community, in order to confirm that the results matched their perceptions and experience.

In the second, quantitative stage, the first technique used to establish face validity was the use of established scales, where these were available. In addition, factor analysis was used to check that individual items mapped to their expected scales. The results of the factor analysis were used to confirm which items could be grouped as variables in order to test the research model in the subsequent stepwise hierarchical regression analysis. Items that did not exhibit sufficient convergent validity were dropped from the analysis, and the extent of correlation between factors was checked to ensure that all values were low to moderate.

3.7 SUMMARY

This chapter described the research design, beginning with its philosophical paradigm. The research took a postpositivist, objectivist paradigm, treating knowledge as being objective and measurable. The research design involved a sequential, mixed methods approach, using an initial qualitative stage to develop the participation construct and validate the proposed research model. Victoria University of Wellington's policy for ethical approval for research involving human subjects was followed, and participants in the first stage were assured that their identity would be kept confidential. The second, quantitative stage did not collect any information that could be used to identify respondents. The online survey was designed following 'best practice' guidelines, and used existing measurement scales whenever possible, modifying them to fit the FLOSS context of the research where necessary. The reliability and validity of the data were tested in multiple ways, to ensure that the results of the research are as sound as possible.

4

PROJECT, INTERVIEWEE, AND RESPONDENT DEMOGRAPHICS

This chapter summarises the characteristics of the projects selected for document review in Stage 1a, the interviewees in Stage 1b, and the people who completed the Stage 2c web-based survey. Overall the results show that LIM FLOSS projects cover the full range of project characteristics, that most projects are international, and that people from a range of backgrounds participate in them.

4.1 STAGE 1A: DOCUMENT REVIEW FOR SELECTED PROJECTS

This section describes the 10 projects selected for ongoing document review. It begins with a description of each project, including its name, purpose, and history.

Section 3.2.1 on page 72 discussed the purposive sampling technique used in this stage of the research. The projects were chosen to be representative of the range of FLOSS applications available in the LIM field, using as selection criteria:

- the number of installed sites;
- project age;
- the application type;
- the number of developers;
- the project's complexity, and
- its activity level.

The *number of installed sites* was determined using several techniques, depending on the project. Some project web sites included a page listing known users of the software. In other cases, a harvesting site such as OpenDOAR[1] (http://www.opendoar.org/) provided statistics about the number of known users of different institutional repository software packages. In a small number of cases, little information was about the user base was available, and the locations of people posting to the project's email discussion list and/or discussion forum were used as a surrogate for the number of users.

The *project age* was determined by examining the source code archives for the earliest release date for the software.

1 OpenDOAR is an acronym for the Directory of Open Access Repositories, a directory of academic open access repositories.

The *application type* was determined from the description of the project's functionality, available on its website.

The number of developers was established by examining the source code archives for contributions by different people.

The *complexity* was estimated by identifying the number of data and transaction types supported by the software.

Finally, the *activity level* was determined by reviewing archives of the project's email discussion list and/or discussion forum to determine the average number of messages sent each week, as well as tracking the number of releases of the software in the last five years.

Overall the typical FLOSS LIM project was found to have a low number of installations, a medium number of developers, but moderate to high complexity. Because of the purposive approach to choosing the sample, it represented a cross-section of combinations, giving a broad perspective on the way the projects developed and the way participants interacted. Table 10 on the next page summarises the characteristics of each project.

4.1.1 *Greenstone*

The Greenstone project is one of the earliest FLOSS projects available for the LIM field. It began as a research project in the Department of Computer Science at the University of Waikato, and was first released under the GPL in the late 1990s. Greenstone is usually described as digital library software, and its main purpose is to provide web-based or CD-ROM access to a collection of digital documents. UNESCO and the Human Info NGO, a humanitarian organisation that provides information, data processing and dissemination services, cooperate with the Greenstone Digital Library Project in its development and distribution. A detailed description of Greenstone's purpose, history, and development is available in Witten, Boddie, Bainbridge, and MacNab (2000) and Witten and Bainbridge (2007). The project's web site is http://www.greenstone.org/.

4.1.2 *EPrints*

The first version of EPrints was released in late 2000. The project was initiated by Stevan Harnad, an early proponent of the open access movement to make scholarly and academic research more widely available. EPrints functionality is designed to support institutional archiving of research papers, and make them available on the Web. The software also exposes the metadata for the items in the repository to search engines such as Google. The School of Electronics and Computer Science, University of Southampton, develops EPrints. Millington and Nixon

Table 10: Stage 1a Project Characteristics

	NO. OF SITES[a]	PROJECT AGE	TYPE	NO. OF DEVEL-OPERS[b]	COMPLEXITY[c]	ACTIVITY LEVEL[d]
1	High	More than 10 years	Digital library	Medium	Moderate	High
2	Medium	5-10 years	Institutional repository	Medium	Moderate	Moderate
3	High	5-10 years	Integrated library system	High	High	High
4	Low	Less than 5 years	Integrated library system	Medium	High	Moderate
5	Low	5-10 years	Record manipulation utility	Low	Low	Low
6	Low	More than 10 years	Web portal	Low	Moderate	Low
7	Medium	5-10 years	Integrated library system	Low	High	Low
8	Medium	More than 10 years	Electronic resource management	Low	Moderate	Moderate
9	Medium	5-10 years	Institutional repository	Medium	Moderate	Moderate
10	High	5-10 years	Journal publishing	Medium	High	Moderate

[a] No. of sites: Based on the number of installed sites that could be identified as using the software. Low=fewer than 100; Medium=between 101 and 249; High=over 250

[b] Number of developers: Determined from the available source code. Low=1-5; Medium=6-20; High=More than 20

[c] Complexity: A categorisation based on principles derived from function point analysis (Albrect and Gaffney 1983), which involved the number of data types handled by the software, and the associated number of different transaction types available to its users. As a rule, each different type of data defined for the software requires some special processing, so systems that handle many different types of data tend to have more transaction types as well. Low=1-5 data types and associated transactions; Medium=6-10 data types; High=more than 10 data types.

[d] Activity level: A categorisation based on the release frequency and the number of postings to the project's main email discussion list. Low=less than one message per week, and no more than one release per year; Moderate=1-5 messages per week, and no more than two releases per year; High=more then 5 messages per week, and more than two releases per year.

(2007) provide a summary of EPrints features. The project's website is http://www.eprints.org/.

4.1.3 *Koha*

The Koha software was first released in late 2000. The project was initiated by the Horowhenua Library Trust in Levin, New Zealand, and the software was developed by Katipo Communications in Wellington. Koha is an ILS (integrated library system), providing functionality to support a range library processes, such as acquisitions, cataloguing, and circulation, as well as having an online catalogue for library patrons to use. It is written in Perl and MySQL. Ransom, Cormack, and Blake (2009) describe the history and development of the project in detail, including its adoption by other libraries around the world. The project's website is http://koha-community.org/.

4.1.4 *Evergreen*

The Evergreen software, first released in 2006, was initiated by the Georgia PINES (Public Information Network for Electronic Services). Like Koha, it provides ILS functionality, supporting cataloguing and circulation, as well as an online catalogue for library patrons. Evergreen is written in Perl and PostgreSQL, plus JavaScript and XUL. Unlike Koha, which was designed for use by a single library (or library system), Evergreen's original focus was on meeting the needs of a library consortium. A majority of Evergreen's current users are in North America. Molyneux (2009) provides an overview of the project's history, development philosophy, and future plans. The project's website is http://www.open-ils.org/.

4.1.5 *MARC-Record*

MARC-Record is a set of Perl utilities to manipulate bibliographic records in the MARC (MAchine-Readable Cataloging) format. It was based on an earlier project, marc.pm, developed by a group of library programmers in the late 1990s, and the first utility, MARC::Record, was released in 2001. Because the structure of MARC records is static, this software is relatively stable, and it had the least activity in the group selected for this stage of the research. The project's website is http://marcpm.sourceforge.net/.

4.1.6 *MyLibrary*

The MyLibrary software was designed to allow a library to provide its users with the ability to personalise a portal view of the library's col-

lection. Library staff create descriptions of resources that are available to library users; these descriptions include detailed subject metadata. Library users can create a personalised profile that indicates the subject(s) they are interested in, while the software uses this information to display relevant information resources automatically. It was developed by Eric Lease Morgan, and first released by North Carolina State University Library in 1999. It is now developed and maintained by Notre Dame University Library. Morgan (2008) describes the most recent version of the software, with a case study of its use at the University of Notre Dame's Hesbrough Library. The project's website is http://mylibrary.library.nd.edu/.

4.1.7 *PhpMyBibli*

PhpMyBibli is also an ILS, developed in France. It was first released in 2003 by François Lemarchand, and is now maintained by the French company PMB Services. It provides similar functionality to Koha and Evergreen, specifically cataloguing, circulation, and an online catalogue. Unlike Koha, which is primarily written in Perl, PhpMyBibli is written in the Php programming language. A majority of PhpMyBibli's users are located in Europe. The project's website is http://www.pmbservices.fr/nouveau_site/pmbservices.html.

4.1.8 *reSearcher*

reSearcher is a suite of software packages that provides support for library electronic resource management. Its components are:

CUFTS software that allows library staff to manage information about electronic journals, aggregations of journals, and subscriptions;

GODOT software that matches citation information with journals, in order to locate full-text versions in a single step; and

DBWIZ a tool for searching across multiple databases.

These tools are used with Open KB, a database that records which journals are available in which databases, and works with CUFTS and GODOT. The software was originally developed by Simon Fraser University Library in British Columbia, with the support of the Council of Prairie and Pacific University Libraries and the British Columbia Electronic Library Network. Components of the reSearcher software suite have been available since the late 1990s; Stranack (2007) provides an overview of the history and main components of reSearcher. The project's website is http://researcher.sfu.ca/.

4.1.9 *DSpace*

The DSpace project began as a result of increased interest in providing access to the research outputs of academic staff within a single institution. The software allows people to upload copies of articles, videos, images, or other types of files to a central repository. Like EPrints, the metadata associated with these documents can be exposed to search engines such as Google, in order to make the resources more widely available. The software was originally developed by MIT and Hewlett-Packard; additional funding was provided by the Mellon Foundation once the software was released under a FLOSS license in November 2002. DSpace is similar in purpose to EPrints, but written in a different language (Java rather than Perl). Walker (2007) discusses the history and development of DSpace. The project's website is http://www.dspace.org/.

4.1.10 *Open Journal Systems*

Open Journal System (OJS) is part of the Public Knowledge Project started by John Willinsky at the University of British Columbia. OJS supports publishing an online journal, including submitting articles, assigning them to referees, adding referees' comments, notifying authors about the comments, and publishing the final version of the paper. The first version of the OJS software was released in 2001. Willinsky (2006) discusses the OJS software in more detail. The project's website is http://pkp.sfu.ca/?q=ojs.

4.2 STAGE 1B: SEMI-STRUCTURED INTERVIEWS

This section presents and discusses the characteristics of the 24 people who were interviewed for this research.

A total of 29 people was invited to be interviewed for this stage of the research; six of the people who received invitations to participate in email interviews and one person invited to participate in a face-to-face interview did not respond to the invitation. In each case, a replacement interviewee with similar characteristics, though not necessarily from the same project, was selected, and subsequently interviewed. One person accepted the invitation to be interviewed, and indicated a preference for using instant messaging (IM) for a synchronous interviews, rather using email. However, this person did not respond to several messages asking to suggest a suitable time for the interview, and another interviewee with a similar background was selected. One other person also asked to use IM for the interview, rather than email. One face-to-face interviewee suggested inviting two co-workers to participate in the interview, in order to give a more complete picture of the use of the FLOSS application within their organisation, and this suggestion was accepted.

Table 11 on this page lists the characteristics of each interviewee, and the type of interview.

Table 11: Stage 1b Interviewee Characteristics

PARTICIPANT	TYPE	GENDER	PROJECT
1	Synchronous	Female	A
2	Synchronous	Male	A
3	Synchronous	Male	A
4	Synchronous	Female	A
5	Synchronous	Male	A
6	Synchronous	Male	A
7	Synchronous	Female	A
8	Synchronous	Female	A
9	Synchronous	Female	A
10	Synchronous	Female (3)	B
11	Asynchronous	Male	B
12	Synchronous	Male	C, G
13	Asynchronous	Male	A
14	Asynchronous	Female	A
15	Asynchronous	Male	A
16	Asynchronous	Male	D
17	Asynchronous	Female	D
18	Asynchronous	Male	D
19	Asynchronous	Male	D
20	Asynchronous	Male	E
21	Synchronous	Female	F
22	Asynchronous	Female	A

The ethical permissions for the interviews said that the thesis and any resulting publications would not include any information that could be used to identify respondents, and therefore letters have been used to identify individual FLOSS projects. These letters do not correspond to the numbers in Table 10 on page 99; it is not possible to associate interviewees with specific projects, as this might make it possible for someone familiar with the project to identify them.

Three projects listed in Table 10 on page 99 are not represented in the interviews. This is because the people selected from the projects as potential interviewees did not respond to the invitation to be interviewed, and no similar participant in the project could be identified.

As Table 11 shows, just over half (12 of 22) interviews took place using synchronous communication methods (face-to-face, telephone, and instant messaging), and the remainder were asynchronous (email). There was an even split of genders, with 12 female and 12 male interviewees. The interviewees had a range of educational qualifications; two had PhDs, three had postsecondary certificates or diplomas, one was currently studying for a post-graduate diploma, one was halfway through a Bachelor's degree, one was studying for a Master's degree, and the rest (14) had completed either Bachelor's degrees or Master's degrees.

Their experience with the FLOSS project they described varied. Two had extensive experience (over 10 years), and at the other end of the spectrum, four had approximately six months' experience. The rest were somewhere in the middle, and the median experience was between four and six years using the software.

More than half of the interviewees were associated with the same project, Project A. While this may introduce bias in the results of this stage, this high proportion was in part because this project A had the largest number of identifiable roles (project sponsor, core developer, project manager, user, system administrator, interface designer, and release manager), which ensured that the interviewees involved with the project represented a broad perspective of views, something that was not easy to determine for smaller projects. One developer had moved onto a new project, and was no longer actively involved in the project; he answered the questions based on his previous involvement, which gave a historical perspective on the software and its community. In order to minimise the effects of this bias, the types of activities identified by the interviewees representing Project A were compared to those identified by interviewees involved with other projects, and activities found in more than one project were given priority in discussing the results.

Half of the interviewees were primarily software users, while the other half had a more active role in the project. Developers, and people in related roles, such as release manager and interface designer, formed one third (8 out of 24) of the interviewees. One interviewee had been involved in different roles in two different projects, and his interview consequently covered both of these perspectives. Two interviews involved project business owners, responsible for starting new projects.

One of the challenges in identifying potential interviewees was that most of the people who could be identified for this stage of the research were likely to be satisfied with the software, rather than dissatisfied with it, because their activities were visible. People who had had a negative experience with LIM FLOSS would be more likely to stop using the software, and would therefore not participate in project email discussion lists or write about their experiences, making them difficult to identify. This suggests that the results of this stage of the research may be more useful in identifying reasons that participants are satisfied with their

experience of a FLOSS project, while factors that lead to dissatisfaction may be under-represented.

In order to locate potential interviewees who had a less than optimal experience, a library in the early stages of its implementation of one of the FLOSS integrated library systems was identified, and four of its staff were invited to be interviewed. All accepted the invitation, and were willing to discuss the negative aspects of their experience as well as the positive ones, which gave a more balanced perspective to the interviews.

4.3 STAGE 2C: WEB-BASED SURVEY

In the three weeks that the survey was available, 183 respondents completed it; several partial responses in which only the demographic questions were answered were also submitted. These were discarded before beginning the analysis; the demographic data provided suggest that these responses were primarily from younger respondents (most in the 21–25 age group). They had the same gender balance as the overall survey.

The following tables summarise the demographic characteristics of the respondents. Some respondents did not answer every question, because all of the questions were optional.

4.3.1 *Age Group*

Table 12 on the current page summarises the responses to Question 1.

Table 12: Age of respondents

AGE GROUP	n	%
20 or younger	1	0.6
21–25	5	2.8
26–30	34	18.8
31–35	29	16.0
36–40	37	20.4
41–45	28	15.5
46–50	15	8.3
51–55	15	8.3
56–60	10	5.5
61 or older	7	3.9
Total	181	100.0

As Table 12 on this page shows, two respondents did not answer this question. The results show that respondents represented all age groups,

and that the median age range was 36–40. Because all age groups are represented in the respondents, and the distribution is similar across the bands from 26–30 through 41-45, the results are unlikely to be biased toward one particular age group.

4.3.2 *Gender*

As Table 13 on the current page shows, two respondents did not answer this question.

Table 13: Gender of respondents

GENDER	n	%
Male	123	68.0
Female	58	32.0
Total	181	100.0

The results show that most respondents are men. This imbalance is common in FLOSS survey research, with previous studies of FLOSS developers, such as Ghosh et al. (2002), finding that women make up a very small proportion of survey respondents (generally less than 5%). The type of projects selected for this research has previously been underrepresented in FLOSS research, which has tended to involve large-scale surveys directed at developers, or case studies of single projects, often Linux or Apache. Lamont (2009, p.137) found that male library computer systems department heads were approximately twice as common as female ones, which is similar to the proportions found in this research. This finding suggests that the sample represents a different population than earlier FLOSS research, but that it is representative of LIM computer systems staff.

4.3.3 *Educational qualifications*

As Table 14 on the facing page shows, one respondent did not answer this question. The results show that the typical respondent has a tertiary education, and a majority has a Master's degree. This is typical of the target population for the survey, since a Master's degree is generally required for professional LIM positions.

4.3.4 *Country of residence*

Table 15 lists all countries with more than one respondent individually, with countries which had just a single respondent been included as 'Other'.

Table 14: Highest educational qualification

HIGHEST QUALIFICATION	n	%
Secondary or high school graduate	4	2.2
Postsecondary certificate or diploma	4	2.2
Undergraduate degree	32	17.4
Postgraduate certificate or diploma	19	10.4
Master's degree	113	61.7
PhD	10	5.5
Total	182	100.0

Table 15: Country of residence

COUNTRY	n	%
United States	76	41.5
India	22	12.0
New Zealand	22	10.9
United Kingdom	20	11.7
Australia	7	3.8
Canada	5	2.7
Germany	3	1.6
Argentina	2	1.1
Belgium	2	1.1
Other	24	13.1
Total	183	100.0

As Table 15 on the previous page shows, all respondents answered this question. The results show that respondents came from 33 countries, including both developed and developing countries. The largest number of respondents came from the United States, and the second largest from India. New Zealand is over-represented in the results, with 22 (10.7%) respondents. This may be because two of the widely used LIM FLOSS projects, Greenstone and Koha, originated in New Zealand, and the researcher is based there.

The 'Other' category includes six European Union (EU) countries, four European countries that are not part of the EU, four countries in Africa, four in Asia, three in South America, two in North America (Mexico and Puerto Rico), one in the Caribbean, and one in Oceania.

4.3.5 *Years using a computer*

Table 16: Years using a computer

RANGE	n	%
Less than 5 years	2	1.1
5–10 years	15	8.2
11–15 years	28	15.3
16–20 years	44	24.0
21–25 years	51	27.9
26–30 years	31	16.9
More than 30 years	12	6.6
Total	183	100.0

All respondents answered this question. The median response is 16–20 years, and 182 (88.8%) respondents had used computers for 11 or more years. This suggests that the survey results represent people who are familiar with using technology.

4.3.6 *Operating systems used*

Questions 9 and 11 of the survey asked respondents to identify the operating system(s) they used on computers provided by their employer, and on computers they owned, respectively. Because of the many possible responses, this was an open question, to allow respondents to enter as much information as they wished. Two respondents did not answer question 9 and three did not answer question 11.

Table 17: Number of operating systems used

NUMBER	n (WORK)	% (WORK)	n (HOME)	% (HOME)
1	116	64.1	111	61.7
2	39	21.5	45	25.0
3	22	12.2	20	11.1
4 or more	4	2.2	4	2.3
Total	181	100.0	180	100.0

Table 18: Operating systems used

TYPE	n (WORK)	% (WORK)	n (HOME)	% (HOME)
Windows	137	74.9	115	63.9
GNU/Linux	84	45.9	86	47.0
Mac OS	26	14.2	46	25.6

The results show that there is considerable variation in the underlying platforms used by respondents, as shown in Tables 17 and 18. Table 17 shows the number of operating systems used by respondents, Table 18 indicates the specific type of operating stems used, and Table 19 shows whether they use only proprietary operating systems, only FLOSS ones, or a mix of each. Both tables show work and home operating system use separately.

Approximately one-third of respondents used more than one operating system at work, at home, or both, meaning that the totals for the columns are more than 100%.

These results show that the respondents use a range of operating systems, and also that a higher proportion of respondents use Mac OS and GNU/Linux than typically found in business and government.

The differences between the operating systems used in the two environments shows that where respondents have a choice of which software to use (that is, on their own computers), slightly more use GNU/Linux

Table 19: Type of operating systems used

TYPE	n (WORK)	% (WORK)	n (HOME)	% (HOME)
Proprietary only	97	53.0	94	51.4
Mixed proprietary and FLOSS	55	30.1	50	27.1
FLOSS only	29	15.8	36	19.7
Total	181	100.0	183	100.0

than at work. In addition, fewer respondents use Microsoft Windows at home than at work.

4.4 UNDERSTANDING AND USE OF FLOSS

Questions 7, 8, 10, and 12 asked respondents to indicate their familiarity with free/libre and open source concepts, and the extent to which they use FLOSS software at work and on computers they owned. Table 20 on this page and Table 21 on the next page summarise their responses.

4.4.1 *Familiarity with FLOSS concepts*

Overall the results show that slightly fewer than half of the respondents have a preference for using FLOSS at work, and slightly more than half have a preference for using it on computers they own. Approximately equal proportions will consider using FLOSS at work or at home (37.7% and 35.0% respectively). Taken together, these results suggest that survey respondents are open to using FLOSS packages when they meet their needs, and this is likely to introduce a bias in the survey results. Since respondents were recruited for the survey because of their existing interest in at least one LIM FLOSS project, this result was expected, and the responses to this question confirm it.

Table 20: Familiarity with FLOSS concepts

FAMILIARITY	n	%
Not at all familiar	0	0.0
Slightly familiar	1	0.5
Somewhat familiar	7	3.8
Quite familiar	42	23.0
Very familiar	133	72.7
Total	183	100.0

All respondents answered question 7. As Table 20 on this page shows, the results indicate that a majority of respondents considered themselves to be very familiar with the idea of FLOSS.

4.4.2 *Attitude to using FLOSS*

Table 21 on the current page summarises respondent answers to questions 10 and 12, which asked them to indicate their approach to using FLOSS on work and home computers.

Table 21: Attitude to using FLOSS

ATTITUDE	n (WORK)	% (WORK)	n (HOME)	% (HOME)
It makes no difference to me	1	0.5	2	1.1
Other people make the decision for me	22	12.0	2	1.1
I prefer to use proprietary software with vendor support	2	1.1	7	3.8
I will consider a free/open source option	69	37.7	64	35.0
I give preference to free/open source whenever possible	82	44.8	83	45.4
I only use free/open source software	6	3.3	24	13.1
Other	1	0.5	1	0.0
Total	183	100.0	183	100.0

The single respondent who chose the 'Other' option for computers provided at work indicated that he had little decision making power, but did have a preference for FLOSS software if consulted. The person who chose this option for computers he owned indicated that all of his computers were provided by his employer.

4.5 SUMMARY

This chapter presented the key demographic characteristics of the projects selected for observation in Stage 1a, the Stage 1b interviewees, and the Stage 2c survey respondents. It showed that LIM FLOSS projects are varied, with differences in complexity, numbers of identifiable users, activity level, and number of developers. The Stage 1b interviewees represented six different projects, and included both technical and user roles. Respondents to the Stage 2c web-based survey were diverse, representing 33 countries. They ranged in age from under 20 (1 respondent) to over 61 (7 respondents). While they were all involved with at least one FLOSS project, they also used both free and proprietary operat-

ing systems. Slightly fewer than half indicated a preference for using free/open source software when they had a choice, and approximately one third would consider it.

5

CONTRIBUTING TO A FLOSS PROJECT

This chapter answers the first sub-question for this research project, *What types of contributions do participants make to free/libre and open source projects?* The choice of wording for this question deliberately focuses on contributions rather than roles. Its answer does not involve making an exhaustive list of activities that people participating in FLOSS projects carry out; rather, it focuses on developing a typology that can be used to classify activities, and give a rich perspective on what is involved in creating a thriving FLOSS community. Section 2.3.1 on page 18 showed that existing literature about participant roles in FLOSS projects generally uses frameworks and models that focus only on their interaction with the project's source code. The data gathered for the first stage of this research show that coding is only one dimension, and that participation in a FLOSS project community may involve a range of other activities for people who lack the skills to write or test software.

5.1 MORE THAN JUST CODE

Every FLOSS project is different. They are different in that the software has a different purpose and functionality, the developers are different, the users are different, the programming languages vary, the standards vary, etc. However, projects share a number of characteristics. In particular, they all use a FLOSS licence to release the source code. They all carry out similar activities, no matter how small or how large the project is. They all involve a community of participants interacting with each other to determine the future of the project. This stage of the current research focused on these similarities, in order to understand the most common types of contributions people make to FLOSS projects.

Using a code-centric perspective to represent FLOSS participant roles omits a number of activities, such as those that relate to supporting the community. These activities become more valuable as the software is adopted by people who are not part of the original group of developers, and its number of users increases. This chapter describes seven dimensions of participation, typical activities associated with each dimension, and four attributes of FLOSS project participation that overlay the other dimensions. The approach used to identify these dimensions is discussed at the beginning of Section 5.2 on the next page.

The goal in identifying these dimensions was to be able to categorise all of the activities that were identified in the FLOSS projects that formed part of the sample for Stage 1a of this research, or described by the interviewees in Stage 1b, while at the same time keeping it as succinct

as possible. To provide a framework that gives a generic view of the types of activities involved in a FLOSS project, the focus was on identifying themes and activities common to a range of projects, rather than activities that were found only in a single project.

5.2 TYPES OF CONTRIBUTIONS MADE TO FLOSS PROJECTS

This section describes the types of contributions interviewees made to FLOSS projects, not their specific roles. The early analysis showed that all interviewees who were identified as users in Table 11 on page 103 did more than passively use the software, which suggests that identifying groups of activities is more realistic than trying to identify distinct roles.

The range of activities identified by Stage 1b interviewees, and from the unobtrusive observation of 10 different projects in Stage 1a of this research project, were listed and then categorised by the main objective or goal of the action. Once these were listed, similarities between the objectives were identified, and seven broad categories emerged from the data. These are described below, followed by a discussion of four important attributes that were found in all of the categories.

5.2.1 *Use*

While use of the software is not inherently a 'contribution' to a project, without users the software would have no purpose, and there would be no need for the other types of activities discussed below. In other words, 'Software is for use.', to paraphrase Ranganathan's first law of library science, "Books are for use." (1963)

Users are the people for whom the software's functions are relevant, and their interactions with the software, whether frequent or infrequent, test the extent to which the software meets their requirements. Every user is likely to have subtly different needs and expectations for the software, and the larger the pool of users, the more likely it is that all features of the software will be used at some point. Half of the interviewees included using the software when asked to describe their role in the project. However, only one of them used the term 'end user' and gave the impression that use was the main focus of her interaction with the software/project. Nevertheless, even this interviewee identified changing parameters for the way the software worked as part of her role, which goes beyond basic use. The other interviewees who were classed as users in Table 11 on page 103 gave much more detail about their typical activities, which ranged from reporting problems with the software, to training colleagues in its use, to writing documentation.

In the context of library and information management application software, it is also important to note that the core developers are *not* generally users of the software. Most librarians do not have the technical

skills to develop software, and unlike many of the more widely known FLOSS projects, such as GNU/Linux and Apache, there is likely to be a clearer separation between the developers and the users in LIM FLOSS projects. This means that the relationship between people who work with the project's code, and those who use the software, is more likely to resemble that of a traditional software development project, with the users being more actively involved in specifying requirements and testing changes.

5.2.2 *Interaction with code*

There is a range of activities that fall under this heading. All of them relate to the software and its current or future functionality. Most obviously, this includes writing new code and modifying existing code to fix bugs or add new features. The most basic interaction with the source code can simply involve reading it to understand how it works, but the interview data show that most of the interaction project members have with the code goes beyond this. Much of this is intended to lead to improved functionality, whether through fixing a bug, specifying a new requirement, designing an interface, testing changes, etc. Other types of interaction with the code included reporting bugs, packaging the code for a new release, or translating the user interface into another language. All of the interviewees mentioned this type of activity, even if they didn't contribute code directly themselves. Most appreciated the benefits that they gained from code contributed by other community members.

5.2.3 *Supporting the community*

Once the user group has grown beyond the project's initiators, there is an increased need to support current community members. The most common community-related activities included writing user and system documentation and answering questions on project discussion lists and forums. Table 22 on the following page shows the full list of community-oriented activities identified by interviewees, and the number of times each was mentioned.

As projects become more established, the range of support activities offered by members of their communities grows, with experienced users teaching new ones how to use the software, for example. For two of the projects that formed part of the sample for this research, community members organised user and developer conferences, and one project had an active user group that met when convenient, usually held in association with major library conferences.

One held its first conference in 2006, and the second in 2009, with a third conference held in late 2010. A second project, first released

Table 22: List of community-oriented activities

ACTIVITY	TIMES MENTIONED
Write documentation	10
Answer questions	6
Provide training	3
Run workshops	2
Organise conference	1
Organise user interface translations	1
Set up user group	1
Translate documentation	1

several years after the previous one, held its first conference in 2009, its second in 2010, and its third in 2011. A third project has also held two conferences, in 2007 and 2009, and has recently announced a third to be held in the third quarter of 2011. In this case, the forthcoming conference will be held on a different continent to the first two, showing that the geographic distribution of members of this project's community is changing. This shows that participants in all three projects recognise the importance of face-to-face contact for community building, and in the case of the third, that it is important to hold conferences in different locations to make it possible for more, and different, people to attend. Two interviewees who were involved in the first project attended the 2006 conference, and both commented on the benefits they gained from meeting people they had previously known only virtually.

One aspect of interacting with the community that was not mentioned by any of the interviewees, but was observed in several of the projects, was mentoring new community members. This took many forms, but it often included encouraging them to contribute, or helping them understand how to contribute in a more active way. Though the time spent on mentoring activities may not always result in actual contributions to the community, it is important to both the mentor and the 'mentee', because of its potential to promote community spirit. In the FLOSS projects that were observed in Stage 1a, the ones that had formal and informal mentoring appeared more likely to grow and develop shared values among members of their communities.

5.2.4 *Outreach*

Activities that fall into this category go beyond the project's existing community, and are intended to promote the project to new users. Typical outreach activities included giving presentations and/or demonstrations at professional conferences, and writing articles for professional and practitioner journals. In some cases, these activities included giving introductions to basic FLOSS concepts, which are still not widely understood in the library profession, or in the wider community.

One example of an outreach activity intended to reach a wide audience is Software Freedom Day, held each year on the third Saturday in September. Generally FLOSS supporters within a city or region decide what, if any, activities they want to organise for this day. While much of the activity is general, intended to introduce the concepts of free/libre and open source software to members of the public, some communities also provide opportunities for demonstrations of specific software packages, and may also give away installation CD-ROMs or USB memory sticks containing FLOSS software. Because Software Freedom Day generally takes place on a Saturday, this means that most of the people involved participate in their own time, rather than as part of their paid employment.

5.2.5 *Sponsorship*

This involves seeking or providing resources to support the project, such as funding new developments, providing web hosting facilities for the community's web-based tools, etc.

The choice of the term 'sponsorship' for this category of activities was made after careful consideration of other terms, for example 'support'. Conventional IT projects sometimes use the term 'project sponsor' to indicate the business owner of an individual project, but in other contexts, such as conferences and cultural events, the term 'sponsor' is used to indicate organisations and/or individuals who have provided funds to support the event, or donated physical items, such as pens, conference bags, or USB sticks. The activities observed in the FLOSS projects in Stage 1a, and mentioned by interviewees in Stage 1b, were similar to this type of sponsorship. Therefor, the term 'sponsor' seems the most appropriate choice for activities that relate to providing a range of resources, since no project can be undertaken without resources. To have credibility, a FLOSS project needs to have a website, and methods for members of its community to communicate with each other. These methods typically include asynchronous email discussion lists, web-based discussion forums, or both, and many projects also use at least one synchronous channel, usually IRC (Internet Relay Chat). All

of these require a web hosting service so that they are available to any community member, or potential member.

While some projects use freely-available services for their web hosting, such as the ones offered by SourceForge or Google Groups, others use a dedicated project domain and other resources, which need to be funded by someone. In some cases these are provided by the individual or organisation that started the project, but for other projects these may be provided by later adopters, often companies that have been established to support the software on a commercial basis. One interviewee, a project sponsor, mentioned the importance of setting up resources to support the project early on, saying "we really realised that it needed to be supported in some way", and went on to discuss getting another user of the software to provide funding to retain the software developers on a maintenance contract, in addition to setting up user- and developer-oriented email discussion lists.

Another type of sponsorship is funding specific enhancements that are a priority for a particular organisation. While in some cases these enhancements were done in-house and not contributed back to the wider community, many sponsors chose to make their changes available to all users by having them incorporated into the next release of the software. This benefits the entire community, even though the enhancement is usually specified and funded by a single user or organisation.

5.2.6 *Management*

Management activities all shared two characteristics: they involved coordinating tasks and people, and they had a short term focus, often relating to the next release of the software. Interviewees identified fewer management activities than other types of activities, and tended to associate clusters of management tasks with specific roles. As projects grew and the associated procedures and roles became more formal, these tasks became more important. This may be because the number of people potentially affected by changes to the software is larger, and the community needed to make sure that the consequences were considered before the changes were implemented. Typical tasks in the management category included overseeing the process of preparing a new release, which often involved integrating code from a number of developers, or managing a specific aspect of a project, such as translating the user interface into other languages. Another activity that fitted into this category was determining which features to include in planned releases.

In contrast to governance activities, which are described below, management activities generally related to the near future, and/or a specific release. Another management task mentioned by one interviewee was recruiting new developers for the project, by approaching them individually to see if they would be interested in becoming involved. Most

Table 23: Management roles in FLOSS projects

ROLE	FOCUS
Coordinator	Project
Documentation manager	Project
IT project officer	Local
Operations manager	Local
Project leader	Local
Project manager	Project
QA manager	Project
Release manager	Project
Team leader	Local
Training coordinator	Local
Translation manager	Project

of the management activities identified by interviewees were given a formal title, and acknowledged by other community members. Each role was usually held by a single person at any given time, generally changing only when new versions of the software were released.

There were also a number of management roles associated with local (i.e. organisational) adoption and use of FLOSS; these included being project manager of the initial implementation within the organisation, for example, or managing a software upgrade when a new version is released.

Table 23 on the current page shows the formal management roles identified by interviewees, and whether their focus was on local use of the software, or the wider FLOSS project.

5.2.7 *Governance*

Governance activities were related to management, but had a different perspective, which is the rationale for having them in a separate category. They were less concerned with specific releases and/or deadlines, and more concerned with policies that affected the project's community. In contrast to management activities, which had a short-term focus, FLOSS governance activities involved looking at long term issues, and developing policies and procedures that were essential for the project's (or the community's) future evolution. One interviewee showed a good awareness of governance issues when discussing licensing options for software the organisation was planning to release under a FLOSS license in response to requests to make it more widely available, saying that when he reviewed the current licence provisions he found that they

Table 24: Governance activities

ACTIVITY
Choose/establish governing body
Correct misrepresentations in the media
Define community behaviour standards
Develop appointment process for formal roles
Establish copyright and licensing policy
Monitor behaviour
Monitor media
Resolve conflicts between community members

were contradictory. He noted that most of the code was released under the Apache License, but one small piece used the GNU General Public License version 2, and that according to the Free Software Foundation, these licences were incompatible.

Like management activities, activities associated with project governance became increasingly important as projects grow in size.

In some cases, governance activities also involved setting up (or choosing) an organisation to hold the project's assets in trust on behalf of the community; while the copyrighted source code itself is usually the project's main asset, other assets include domain names associated with the project, documentation, and trademarks. People involved in the project's governance also managed the process of appointing community members to formal roles, such as choosing the next release manager. In keeping with the collaborative nature of FLOSS projects, they usually designed the process so that anyone interested in the project had an opportunity to be involved in making the decision, for example by voting for their preferred candidate. In larger projects, members of the governance group were also involved in setting and documenting community standards for behaviour, or in following up complaints about unacceptable behaviour. This became more important as the project grew, and newcomers unfamiliar with the project's history and norms joined it. Some governance activities, particularly those involving conflict resolution, were managed behind the scenes, rather than in the project's public community email discussion lists or online forums.

Table 24 on this page lists the governance activities mentioned by interviewees. In all cases these activities were carried out by an interviewee whose main involvement with the project involved other types of activities. None of the interviewees was only involved in project governance; the ones who mentioned activities related to governance did so in passing, sometimes as an afterthought.

5.3 ATTRIBUTES THAT CROSS ALL DIMENSIONS

The results of the Stage 1 data analysis also revealed four attributes that provide additional context to understand roles and participation in a FLOSS project. Each of these is associated with more than one type of activity, and is described below.

5.3.1 *Organisational focus*

One aspect of participation in a FLOSS project that emerged from the interviews is that there were two aspects to most interviewees' involvement with the project: a local one that related to their role within their organisation (in other words, a role that related to their employment), and one that related to their role in the wider project community, which sits outside this organisation. *Use* is, of course, always related to the organisation (or to be more precise, not part of the wider project), but several of the people who were interviewed had an active local role (for example, being the project manager for the implementation of the software within the organisation), but little or no involvement with the wider FLOSS community. In contrast, the core developers for two of the projects included in the sample were employed to do this work, all of which was expected to be released to the wider community, and so had a purely "wider project" role.

Seven of the interviewees had a purely local focus; they used the software in-house and had no communication or involvement with the wider community. In one case, another staff member interacted with the wider project community on their behalf as necessary, while in another, the organisation had made extensive customisations to their version of the software, and the interviewee did not feel it was necessary (or even possible) to have that type of interaction, because this version of the software was now unique.

5.3.2 *Role formality*

Another characteristic of FLOSS participation that occurs across the various dimensions is the extent to which the roles interviewees held were formal or informal. A formal role had a job title and well-defined responsibilities, and was recognised by other participants in the project, some or all of whom might have been involved in the appointment process. In contrast, an informal role did not have a title, and the tasks undertaken varied from person to person. When asked how people get involved in projects, several of the interviewees used the word 'volunteer', implying that new participants took on a role that suited their current knowledge and circumstances. One user who contributed documentation said that

she saw a gap that she had the skills to fill, and took this on because she couldn't contribute in any other way.

There is a relationship between organisational focus and role formality. In general, the local roles held by the interviewees were formal, because they formed part of their paid employment. There is some scope for local roles to have a degree of informality, but this is most likely to relate to activities such as mentoring or training new users, and most participants who took on this type of informal local role also had a more clearly defined formal role.

This is not true for roles within the wider project; most of the roles participants held were informal. All of the projects that formed part of the sample for this stage of the research had only a few types of formal roles, held by a small number of people, particularly when compared with the total number of people involved with the project.

5.3.3 *Remuneration*

The third of these shared attributes is the extent to which the interviewees were paid for their work on the project. All interviewees were paid for their formal, local roles, but only some were paid for their project-related roles, whether formal or informal. Because most of the participants in this research project used or developed the software as part of their employment, they were paid for their local roles, both formal and informal ones. Nonetheless, there is no requirement for people to be paid for all of the work they do on FLOSS projects, even when they take on a formal, project-related role. One participant was paid to work as a full-time developer on the project, but also took on a formal management role in the project, which he carried out on an unpaid basis, largely in his own time. His paid work required him to write code for new features, while his management role involved managing the next release of the software, which included contributions from a number of other developers.

5.3.4 *Time commitment*

Finally, the amount of time that the interviewees spent in FLOSS project-related activities varied considerably. For some, it was the major part of their employment, and they therefore were working with the software on a full-time or near full-time basis. For others, the software was a minor part of their role, and they fitted it in around their other work. A third group, smaller than the other two, were involved with the project on a purely voluntary basis, and worked on it in their free time.

5.4 DISCUSSION

Every project included in the sample for Stage 1a was different, just as every person interviewed for Stage 1b had a different perspective on the project and roles within the project. Their involvement in the project was a unique combination of formal and informal roles; one thing that was particularly noticeable about the findings from this stage of the research was that many interviewees carried out tasks that fell into a number of dimensions, even though they identified themselves as a 'developer' or a 'user'. Even people who described themselves primarily as a user of the software also carried out outreach activities, most often giving a conference presentation about their experience, or giving one-on-one demonstrations to prospective users. One member of a FLOSS project that was part of the Stage 1a sample expressed this in a recent discussion board posting, saying "Everyone in [project X] wears at least 10 different hats :)".

Table 25 on the next page shows the distribution of types of activities by interviewee. A ✓✓ indicates their main focus in the project. In several cases an individual interviewee's activities covered six of the seven categories; in other words, they used the software or updated the code, provided support to community members, worked to promote the software to new users, provided resources (often in the form of space on a shared server), managed at least one aspect of the project, and were involved in determining overall policy for the project. These 'omni-roles' usually occurred in the smaller projects, and larger projects tended to have a clearer separation of roles, with more specialisation. However, even then interviewees whose activities were concentrated in one of these areas generally showed an awareness that the other activities were important for the health of the project community, suggesting that most FLOSS participants understand that the project involves more than just writing and testing source code. In general, the longer interviewees had been involved with the project, the broader their perspective on the range of activities carried out by other members of the project's community. This was the case for all types of primary involvement.

5.4.1 *A user-centric view of a FLOSS project*

Section 5.2 on page 114 identified seven types of activities community members contribute in a FLOSS project. These are represented in Figure 4 on page 125. The model places use at the centre, and groups the remaining six types of activities in three pairs representing the concepts of 'project fitness', 'project viability', and 'project spirit'. For illustrative purposes the three groups of activities were drawn as the same size, but in practice their proportions will vary from project to project, depending on a number of factors, such as the project's size, age, and complexity.

Table 25: Interviewee activity summary

INTERVIEW NO.	USE	CODE	SUPPORT	OUT REACH	SPONSOR	MANAGE	GOVERN
1	✓		✓	✓	✓✓		
2		✓✓	✓	✓	✓	✓	✓
3		✓	✓✓	✓	✓	✓	
4	✓				✓		
5		✓	✓	✓	✓	✓✓	
6	✓✓		✓				
7	✓✓		✓				
8	✓✓	✓					
9	✓✓		✓	✓	✓	✓	✓
10	✓✓	✓		✓		✓	
11	✓	✓	✓✓	✓			
12		✓	✓	✓	✓	✓✓	✓
13		✓	✓✓			✓	✓
14	✓✓		✓		✓		
15	✓	✓	✓✓	✓	✓		
16		✓	✓✓				
17		✓	✓✓			✓	✓
18		✓	✓✓			✓	
19	✓✓	✓	✓			✓	
20	✓	✓	✓	✓		✓✓	
21	✓✓	✓	✓	✓			
22	✓✓		✓	✓			
Total	14	15	18	12	9	11	5

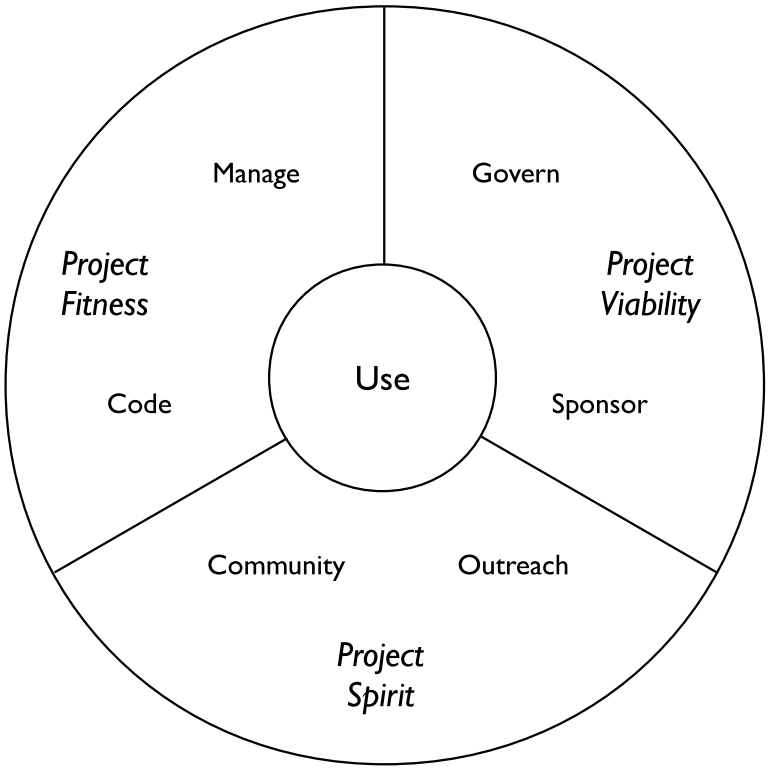


Figure 4: FLOSS contribution model

Any FLOSS project which does not have participants involved in all seven types of activities over time is likely to be unsustainable in the long term, so this framework may be helpful in considering a project's sustainability.

The three higher level categories were developed by considering whether the six activities that support use could be grouped in a way that gave additional insight into the activities identified in the FLOSS projects. This involved comparing the six categories in terms of their overall purpose, plus their long-term effects, until a meaningful combination emerged. All combinations were considered, but the one presented in Figure 4 on this page was considered to be the simplest and most meaningful. This model represents an interpretive perspective on the seven types of activities, based on the researcher's familiarity with the projects, interviewees, and data gathered in Stages 1a and 1b of this research.

Project viability

The concept of viability is familiar to everyone. The Oxford English Dictionary online defines viability as the "ability to continue or be continued". In the context of a FLOSS project, project viability relates to the expectation that community members will able to carry out their formal

and informal roles for the foreseeable future, without disruption. Governance and sponsorship activities both contribute to a FLOSS project's viability by providing policies and resources that allow community members to focus on achieving their own and the project's goals.

Project fitness

In biology, fitness is defined as "The quality of fulfilling the requirements of a particular environment for survival and reproduction; the capacity of an individual to survive and reproduce." ("Fitness", Oxford English Dictionary online). A FLOSS project's fitness is determined by activities relating to coding and management. Coding activities are necessary to improve the software's functionality, either by adding new features, or by fixing bugs. Any FLOSS project that does not involve coding activities will have static code that does not evolve to meet changing user needs, and is likely to become irrelevant or obsolete over time. Similarly, without some degree of management activity, the code changes are unlikely to be released effectively, which may cause its users to become frustrated. If this continues, users may seek alternative software and leave the community.

Project spirit

A FLOSS project's spirit is an intangible characteristic that describes the "life" of the project, and this comes from the interactions of the various members of its community with each other and with people outside the project. The extent to which community members support each other and work to attract new members will affect an individual's sense of belonging to the community. Over time it will add a sense of structure and style to the project, establishing its core values.

5.5 SUMMARY

This chapter answered the first research sub-question *What types of contributions do participants make to free/libre and open source software projects?* It identified seven distinct types of contribution: use, code, community, sponsorship, outreach, management, and governance, extending the previous code– or developer–centric models. It concluded by presenting a user-centric model of a FLOSS project, and identified pairs of contribution types that established a FLOSS project's spirit, fitness, and viability. In combination, these three characteristics may be thought of as determining a FLOSS project's sustainability.

6

INDIVIDUAL PERSPECTIVES ON SATISFACTION

This chapter begins with a discussion of the findings of the Stage 1b interviews that relate to satisfaction with the software, followed by a description of the dimensions that were used to measure satisfaction in the Stage 2 survey.

6.1 PERSPECTIVES ON SATISFACTION

As in the previous chapter, statements in the interview transcripts relating to satisfaction were coded and the data grouped by broad theme. This was a less structured process than the one used to analyse participation, because the reasons given for interviewee's satisfaction and dissatisfaction were more varied, and reflected individual perspectives more strongly, making the patterns and relationships more subtle.

6.1.1 *Documentation*

It will come as no surprise to anyone who has used FLOSS in any context that poor quality documentation was the single aspect mentioned most often, by both developers and users, when discussing aspects of the project that affected their satisfaction (or in this case their dissatisfaction). Most interviewees who mentioned documentation (both developers and users) said that it was weak, and that there was room for improvement. However, they also acknowledged that this was typical, with one interviewee who had experience of several FLOSS projects saying "Documentation's always a problem".

One interviewee suggested that the only way the documentation would be improved would be if the developers worked on it in their own time, implying that no one in the community was prepared to sponsor writing manuals. This interviewee then went on to say that this was unlikely to happen because community members had other commitments and priorities. However, two interviewees said that the documentation for the software they were using was good, but out of date. One interviewee implied that this was common, saying "things like documentation and installation scripts always lag slightly behind". Other interviewees acknowledged that it was difficult to keep documentation up to date when the software was changing frequently, with one saying "As soon as you did a manual you'd have to change it the next day".

Despite this widespread recognition that user-oriented project documentation could be improved, only two interviewees had contributed

documentation to the project they were involved with; one had written a stand-alone user guide, and the other had contributed to a project wiki. Other interviewees referred to writing in-house documentation, but did not mention contributing this back to the wider project.

One interviewee mentioned that proprietary software documentation is often written to support the vendor's formal training programs, and suggested that the lack of documentation for FLOSS projects was a consequence of the community's informal approach to training. However, he also said that the software was intuitive and that formal training was unnecessary, suggesting that he didn't view the lack of documentation as a significant issue.

One question that two of the interviewees raised was whether community members had the right skills to write good documentation. One suggested that projects should try to attract technical writers to improve the documentation, while the second thought that users would also need to be involved, since they understood the workflows required to make effective use of the software. This interviewee commented that there was good information available about the technical aspects of installing the software on different platforms, because the developers themselves wrote about their experiences on the project's email discussion list. However, he stated that users did not document their experiences in the same way.

6.1.2 *Community helpfulness*

Another characteristic that contributed to interviewees' satisfaction with the FLOSS project they were discussing was the community itself. Most of the interviewees gave examples of good communication between community members, usually on the project mailing lists, and said that this contributed to their satisfaction with the project. One interviewee, who saw himself as an in-house user of the system, relied on another staff member to interact with the wider project community; he said that in his view this was not being handled particularly effectively, with the consequence that their implementation of the software had more problems than necessary. Several of the developers interviewed from the same FLOSS project said that they were disappointed that users tended not to answer questions on the project's email discussion list, suggesting that users expected this project's core developers to answer even simple questions. However, none of them had taken any steps to try to increase user involvement in answering questions.

This example, along with the previous one, raises an interesting point: even when FLOSS project participants perceive something as a problem, they will not necessarily take any action to resolve it. It is not clear why this is the case, since at least one of the projects with which interviewees

were involved has a very open culture, and was generally described as being friendly and welcoming to new members.

Other people, usually those who were in project initiator roles or were early adopters of the software, mentioned seeing the community grow as something that gave them satisfaction. One said that seeing versions of the interface in different languages, even though she couldn't read them herself, was a "thrill".

6.1.3 *Software characteristics*

Several interviewees mentioned the software's functionality, and/or its ease of use, as contributing to their satisfaction with it, while others felt that the opportunity for them to change the software to suit local conditions was its most important characteristic. One interviewee said that she was disappointed with the software because the functionality was very similar to their previous proprietary system, and her view was that the organisation had believed that the FLOSS package would be "better than what we had".

A different group of interviewees had customised the FLOSS package they were using so much that they weren't able to upgrade to the most recent release, but they did not see this as a significant issue, since they did not perceive the new features in this release as important or relevant to them.

Reliability was another aspect of the software that several interviewees mentioned. One said that "It's been going the whole time—it's very reliable", while another said that their system had been running continuously for three months with no problems. This interviewee then said that it was very easy to recover from failures because of the system's architecture, in contrast to what was required for the proprietary software his organisation had previously used, which he described as "painful".

6.1.4 *Cost*

Cost was mentioned by four interviewees. One relied on commercial support for the software, saying that it represented value for money (in comparison with using proprietary software). A second said that she began using the software because she could no longer afford the fees charged for proprietary software. A third interviewee said the lack of license fees meant that they could run development and training versions of their system alongside their production one, which he saw as a key benefit of using FLOSS.

The most extensive discussion of cost came from an interviewee who said that cost was not a factor in their decision to move to a FLOSS package; he explained that they expected their overall costs to be similar

to using proprietary software, but that they would be spending the funds on customisation and fixing bugs, which he saw as being a better use of the money than contributing to a private company's profits.

6.1.5 *Personal benefits*

Two interviewees identified personal benefits that resulted from their involvement with a FLOSS project, which increased their satisfaction. In one case, the interviewee attended a conference and was able to meet leading figures in the wider FLOSS community because of his involvement with the project; the second person had been invited to be on a state-wide library technology committee, and felt that this was because of her adoption of FLOSS software for her library, which her peers perceived as being innovative.

6.1.6 *Complexity*

Two people mentioned complexity as an issue; one said that the underlying technology software stack needed for the software was complex, and had a steep learning curve; the other stated that the software was difficult to install, and that the developers needed to pay more attention to packaging it as a 'one-click' installation.

6.1.7 *Other comments*

Several aspects of a FLOSS project were mentioned by only a single interviewee. The first discussed quality assurance for new releases, which he felt could be improved, since he had found a number of bugs in the most recent release. Another interviewee described a lack of information about the project's future as a source of dissatisfaction, and made a number of suggestions about how this could be improved using social media such as blogs. He felt that regular updates from the project's key decision-makers would contribute to an enhanced feeling of being part of the community. This comment related to a project that had few users involved in answering questions on the project's email discussion list, and suggests that product openness may be important in encouraging contributions from software users.

6.1.8 *Attitude*

One finding that emerged from some interviews was the importance of the interviewees' attitude towards FLOSS in determining their overall satisfaction with the software/project. This was most evident in a series of four interviews that related to the same implementation of one LIM FLOSS package. One interviewee had a technical background, and is best

described as the project's internal 'champion.' He was frank in admitting that the project's implementation had not been 100% successful, but explained that their ability to manage the project in-house and make changes to meet local needs meant that he was positive about its future. The other three interviewees from this organisation were all users; two of them had had formal roles on the project's implementation team, while the third's work was directly affected by the implementation. All three of the users said that they were initially enthusiastic about the project, and understood how using FLOSS was different from their previous proprietary system. Similarly, all three were willing to discuss the issues with the system's implementation, and its effects. However, while two of them (A and B) were still positive about the system's potential, the third (C) expressed significant dissatisfaction, saying "I've been really disappointed with it". This was despite acknowledging that most of the problems she had identified had been sorted out quickly by their in-house developer.

One of the reasons for C's disappointment was that the software didn't meet her expectations in terms of initial functionality; she had been expecting something that was significantly better (from her perspective) that the system it replaced. In addition, C didn't enjoy being able to be involved in determining its future, and felt that she (and the people she worked with) lacked the skills to make good decisions about what needed to be changed, saying "I feel we're trying to do it when we're not really qualified". C had a strong preference for purchasing software that didn't need any local changes, and that provided all of the essential functionality 'out of the box'. She also expressed concern about the amount of time that was required to identify the enhancements that were needed to meet the library's requirements, implying that this would not be needed if they had licensed a proprietary system. Her perspective was that by choosing a FLOSS package, the organisation had made a poor decision, saying "we really are re-inventing the wheel". In contrast, A and B, who were also affected by the implementation issues, were still optimistic about the system's future. In particular, B, who had been the staff member most affected by the issues, identified a number of problems that had created a stressful work environment for her team, but still said "I like the idea of the open source software" and "I'm sure things will come right in time". A's perspective was that having the ability to make local changes was very important, and this compensated for the problems, because of the potential to have something that could be tailored to meet the library's specific needs in the future. In addition, A referred to himself as an "advocate" for the software, saying that part of his position involved promoting the software to other staff in the organisation.

One explanation for the the difference between C's assessment of her satisfaction with the software compared to that of A and B is her underlying attitude to FLOSS. This is particularly evident when compar-

ing her background to that of B. They had similar levels of education and experience; both had the same library-related qualification, and both had over 10 years' experience of working with proprietary library software. They had both used the FLOSS package they were discussing for the same amount of time, and had experienced the same range of problems. But the interviews showed that one was positive about the system's future, while the other was very negative.

Indirect evidence of interviewees' attitude to the FLOSS package they used was also shown by their actions in promoting it to prospective users; in other words, activities that are part of the 'outreach' category. Over half of the interviewees had either given demonstrations of the software to small groups of people, presented a paper at a conference, or written an article about it. As might be expected, the longer interviewees had been involved with the project the more likely it was that they had undertaken some outreach activities; however, one person who had been using the software for one year (less than most of the other interviewees) was particularly active in promoting it, having given several conference papers and demonstrations. Her second email interview was delayed because she was demonstrating the software in another city when I sent the questions, and did not have time to respond promptly.

Interviewees definitely hoped that other people would be encouraged to adopt the software because of their recommendations. One person expressed strong disappointment that no other libraries in her region had adopted the software she had demonstrated, even though it was easy to install and use, and also provided functionality that was not part of standard ILS offerings.

One explanation for these types of voluntary activities is that the interviewees felt a commitment to the software, because of their positive experiences with it. They wanted to encourage other people to gain the same type of experience by sharing their stories. This highlights a difference between FLOSS and proprietary software: public servants may be discouraged from promoting proprietary products, because this could be seen as a conflict of interest under the applicable code of conduct. However, because promoting a FLOSS package does not involve recommending a specific supplier, there is no conflict of interest.

Several other interviewees also mentioned that FLOSS was important to them for philosophical reasons. One developer made the point that he had worked on open source projects exclusively since 1997, and another said that he and other members of his project team were "open source fans", releasing their software under a FLOSS license because they believed it was the right thing to do.

6.2 measuring satisfaction with a floss project

The previous section shows that there are a wide range of factors that contribute to participant satisfaction with a FLOSS project, and that these do not overlap with the existing measures of satisfaction with information systems/software discussed in Section 2.7 on page 32.

In order to answer the main research question for this research, it was therefore necessary to develop a scale to measure participant satisfaction with their experience of a FLOSS project. This research used a characteristic-based approach to measure satisfaction. Section 2.7.3 on page 43 showed that there was considerable variation in the underlying dimensions of satisfaction measured by different instruments developed in previous research. The individual dimensions from previous research were compared to the broad themes identified in Section 6.1 on page 127 in order to develop an initial list of items considered to be particularly appropriate in a FLOSS context.

This list contained the following 11 items (in alphabetical order):

1. easy to add new features;
2. easy to configure to meet local needs;
3. easy to install;
4. easy to learn;
5. easy to use;
6. free from bugs;
7. functionality;
8. helpfulness of community;
9. quality of documentation;
10. release frequency; and
11. reliability.

Items 5, 6, 7, and 11 appeared on the list of system dimensions of satisfaction, and were also mentioned by interviewees, suggesting that they were important characteristics and should be included in this research. Items 1, 2, 3, 8, and 9 were mentioned by more than one interviewee, and were included because they were judged to be key characteristics of a FLOSS environment, based on the researcher's understanding from her ongoing document review of project email discussion lists. Item 4, easy to learn, was included because one interviewee noted that the FLOSS system had a 'steep learning curve', and this was a characteristic not reflected in the other items. Finally, item 10 was included because the FLOSS projects selected for ongoing document review in Stage 1a varied

considerably in their release frequency, which was another characteristic not reflected in the other items.

The initial list was reviewed by a FLOSS developer, and several FLOSS users, to ensure that it had face validity, and that it covered dimensions they considered to be most important. Bryman and Bell (2007, p.165) defined face validity as the extent to which the scale measures the underlying concept, and said that one way of establishing it is to ask other people. One additional item was added as a result of this review, *security and access control*[1]. In addition, the meaning of 'reliablity' was clarified with the addition of "(i.e., doesn't freeze, crash, or lose data)" to make it clear that it was the reliability of the software that was being assessed, as opposed to the reliability of the information the system provided. The final list of attributes used in the survey was:

1. easy to add new features;
2. easy to configure to meet local needs;
3. easy to install;
4. easy to learn;
5. easy to use;
6. free from bugs;
7. functionality;
8. helpfulness of community;
9. quality of documentation;
10. release frequency;
11. reliability (i.e. doesn't freeze, crash, or lose data); and
12. security and access control.

Definitions of these attributes were not included in the survey itself, since this would have made it longer, which could have decreased the response rate. This means that individual respondents may have interpreted the meaning of the attribute slightly differently, leading to ambiguity in the results. However, since none of the people involved in the pilot test raised any questions about the meaning of any of these items, the list was considered to be sufficiently clear on its own.

1 This was treated as a single item, since LIM FLOSS projects typically treat both as part of the system administration module. In multi-user systems, they refer to the ability to create individual user accounts with different levels of access to the various system modules.

6.3 SUMMARY

This chapter discussed the results of the Stage 1b interviews that illustrated interviewees' individual perspectives on satisfaction. It showed that documentation was the aspect of the FLOSS project that was discussed most often, followed by community helpfulness and specific software characteristics. Their underlying attitude towards FLOSS appeared to have a significant influence on some interviewees' satisfaction, even when they had similar experiences with the software. It concluded with a discussion of the development of a scale to measure participant satisfaction with a FLOSS project for the Stage 2 web-based survey.

7

SURVEY RESULTS AND MODEL TESTING

This chapter begins by reviewing the research model presented in Figure 2 on page 59, and then presents the results of the web-based survey, beginning with descriptive statistics showing the pattern of responses. The chapter concludes with the inferential statistics that test the constructs, model and hypotheses.

7.1 RESEARCH MODEL REVIEW

The research model presented in Figure 2 on page 59 included nine constructs that were considered potential predictor variables influencing participant satisfaction with a FLOSS project. The individual perspectives on satisfaction discussed in Chapter 6 on page 127 included additional constructs, such as attitude. Each new construct was considered for inclusion in a revised model, particularly in terms of its relevance to the main research question, its relationship to constructs in the preliminary research model, and the nature of its influence on satisfaction. This last point applied Oliver's distinction between factors that influence someone's initial decision to select a product or service ("choice selectors"), and those that influence satisfaction ("satisfaction drivers") (2010, p.37) to each new concept.

The interviewees generally indicated that being able to control costs was one of their reasons for choosing a FLOSS product, which implies that it is more likely to be a choice selector than a satisfaction driver. In addition, survey respondents would need to have budget authority in order to be able to provide meaningful answers to questions about cost, which might have discouraged or disqualified potential survey respondents. Therefore cost was not included as a component of the model.

Attitudes toward the potential benefits of FLOSS were also considered for inclusion in the model. This is also likely to be a choice selector rather than a satisfaction driver, though it could also be what Oliver (2010) termed a "dual influence factor", meaning that it affects both initial choice and subsequent satisfaction. Since the primary purpose of the research was to measure satisfaction drivers, attitude was not added to the model. A further reason for excluding attitude from the model was that all respondents had already chosen to use FLOSS software, which suggested that the range of responses would be limited. To see if this was the case, questions about attitude to FLOSS were included in the survey, to measure the spread of attitudes across respondents.

Figure 5 on the next page shows the final version of the research model tested in this research, including the number of items for each factor included in the survey. There are no changes from the preliminary model in Figure 2 on page 59. The hypothesis numbers are not included.

7.2 SURVEY RESULTS PART 1

This section presents the results of the web-based survey.

7.2.1 *Project name*

Question 13 asked respondents to name the project on which they would base their answers to subsequent questions. Some respondents entered a local variation of the project name, rather than the name of the underlying project, and five others listed more than one project. Since the purpose of this research was not to compare individual projects, but rather to identify which factors contribute to respondents' satisfaction with the software, these responses were included in the subsequent analysis. One reason for this was that the document analysis of some projects indicated that a few organisations integrated two or more FLOSS projects into a single local application, and it might not have been possible for any respondents from these organisations to restrict their responses to a single project.

The list of software that was generated from this question included integrated library systems (Koha, Evergreen, PhpMyBiblio, repository software (DSpace, EPrints, Kete), wiki engines (ikiwiki, MediaWiki), digital library software (Greenstone), blogging software (WordPress), web content management software (Drupal, Joomla!, Mambo), and signon management software (Shibboleth). In the cases where the respondent had used a local name or a generic term such as 'institutional repository', the underlying software could not be identified, and it was assumed that the specific implementation was based on a FLOSS package. Koha was named most often, by 31 respondents, followed by DSpace (21). The distribution had a long tail of 25 projects that were named by just one person.

7.2.2 *Length of time using or contributing to the project*

Table 26 on page 140 shows the length of time respondents had been using or contributing to the project. Only a small proportion, 6.0% (11 respondents) had been involved with the project for more than 6 years, while 13.1% (24) people had been involved with it for less than 6 months. The largest proportion of respondents was in the 2 to 4 years category, suggesting that the typical respondent was familiar with the software, and had a realistic perspective on its development history and future.

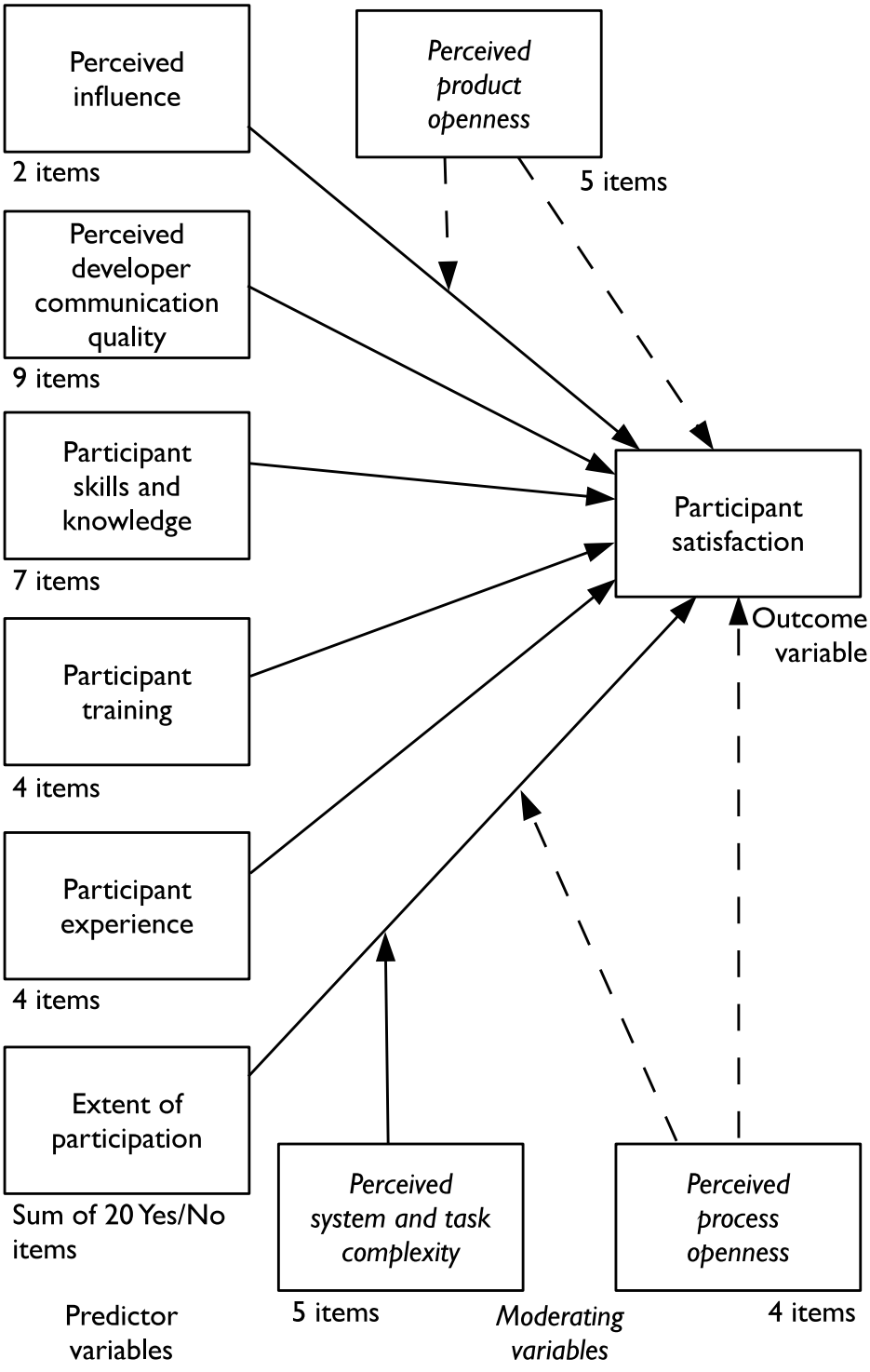


Figure 5: Research model with items

Table 26: Length of time using or contributing to project

LENGTH OF TIME	n	%
Less than 6 months	24	13.1
Between 6 months and one year	31	16.9
1–2 years	47	25.7
2–4 years	50	27.3
4–6 years	20	10.9
6–8 years	5	2.7
More than 8 years	6	3.3
Total	183	100.0

7.2.3 *Project roles*

Questions 15 and 16 asked respondents to indicate their current role in the project, and to indicate any other roles they had held, if any. As with the list of projects in Section 7.2.1 on page 138, there was considerable variation in the data respondents provided in answering these questions, and as the literature review found, there was little standardisation in the terms respondents used to identify their current and previous roles. A majority of respondents (142, or 78.9%) indicated a single current role, while 32 (17.8%) indicated two roles, and 5 (2.8%) three. One respondent named four current roles: implementer, documentor, integrator, user.

A total of 61 different current roles were identified by the survey respondents. The most common was 'user', named by 69 (38.3%) respondents; the next most common roles were developer (29 or 16.1%), maintainer (25 or 13.9%), and trainer (11 or 6.1%). Fifty respondents (27.3%) indicated that they were solely users, with no other current role, though 20 of these (10.9%) had previously held other roles, such as developer or trainer. Several respondents indicated formal positions, such as library manager (2 respondents), repository manager (2 respondents), systems administrator (4 respondents), while others identified specific types of contributions to a FLOSS project, such as bug reporter (1), bug triager[1] (1), content creator (1), or website designer (1). A few respondents added adjectives to qualify their roles, naming them as 'advanced user', 'experimental user', or 'high end user'. One respondent was a self-described '[Project] ambassador: news/marketing/advocacy'.

The most common combination of roles was 'developer/maintainer', which suggests that respondents see the two roles as distinct and needing to be specified separately, or that they see it as a single role without

1 This term means "assessment according to quality" ('triage', OED Online 20 October 2010), and is more commonly used in military, emergency room, or disaster response contexts, where it refers to classifying injured or ill people according to the severity of their condition.

an agreed single descriptive term. Only two respondents used the term 'core developer' and one the term 'committer', suggesting that this terminology for project roles is not widely used in the projects represented in the responses. Overall the roles relating to working with code were generic, and only a few individual responses named specific roles such as 'bug triager' or 'interface designer'. This confirms the findings of Stage 1 of this research, which suggested that most FLOSS projects have only a small number of formal roles and positions, and that most participants have an informal role in the community.

The roles included in the responses to question 16, which asked about other roles respondents had held in the project, followed a similar pattern, but were slightly more varied, including 'development sponsor' and 'event organizer'. One person indicated that she had been on the board of the project's non-profit governing foundation.

All of the types of contributions identified in Section 5.2 on page 114 were included in the responses, with roles relating to use, interaction with code, and interaction with community the most frequent. The least frequent type of contribution was governance, mentioned by only one person. Both types of organisational focus were mentioned, with a number of respondents naming roles that were related to their local work with the software, such as metadata specialist, library team leader, and manager, library services. Some of the respondents showed that they were aware of this distinction; in one example, the respondent qualified the role of developer by adding "in terms of customizing the software for my library".

7.2.4 *Hours per week spent working on the project (internal/shared version)*

Table 27 on the following page shows the average number of hours per week respondents spent working with the software in the last six months, for both a local implementation and for the shared/community version. The results show that respondents are generally more involved with their local implementation than the community project: the median response for time spent in a role relating to the local implementation was 5–10 hours, and for the community project less than 5 hours. Fewer than 10% of the respondents spent more than 30 hours per week in a role relating to either version. This suggests that most people use a range of other software, work on other FLOSS projects, or have other jobs, rather than having a full-time role relating to the software they identified in their response to Question 13.

7.2.5 *Paid project time*

Question 18 asked respondents to indicate what proportion of their time working on the project had been paid for. Table 28 on the next page

Table 27: Hours per week spent working on the project

HOURS PER WEEK	n (LOCAL)	% (LOCAL)	n (SHARED)	% (SHARED)
None	13	7.1	55	30.4
less than 5 hours	53	29.0	75	41.4
5–10 hours	39	21.3	19	10.5
11–20 hours	40	21.9	15	8.3
21–30 hours	20	10.9	8	4.4
more than 30 hours	18	9.8	9	5.0
Total	183	100.0	181	100.0

shows that a majority of respondents (85.1% or 154) were paid for at least some of the time they spent working with the software. However, the results also show that 59.7% (108) are paid for less than 80% of the time they work on the project, suggesting that they have a commitment or interest in the project that goes beyond their formal employment.

Table 28: Paid proportion

PAID PROPORTION	n	%
None	27	14.9
Less than 20%	30	16.6
Between 20% and 50%	21	11.6
Between 50% and 80%	30	16.6
Between 80% and 100%	73	40.3
Total	181	100.0

7.2.6 *Activities carried out*

Question 19 of the survey asked respondents to indicate which activities they had carried out with the software/project by choosing from a list of 20 generic activities. These activities covered five of the seven types of activity identified from the qualitative stage of this project. Management and governance activities were not included because only a small number of interviewees indicated they were involved in them. Table 29 on page 144 lists them in descending order of frequency. The results show that the five most common activities survey respondents were engaged in cover four of the categories (Use, Code, Community, and Outreach), with over 70% of respondents indicating that they had

customised the software for local conditions, and two-thirds engaging in outreach activities. The least frequent activities were fixing bugs and adding new functionality, and only 29% (53) of respondents had provided resources to support the project. One explanation for the low frequencies reported for fixing bugs and adding new functionality is that these require considerable familiarity with the way the software is designed and structured, and only a small number of respondents had this knowledge. However, contributing local changes back to the project was more frequent than either fixing bugs or adding new functionality. Since the respondents represented many countries, some of whose first language is not English, as shown in Table 15 on page 107, some of these local changes may have been translations of the user interface, which respondents are unlikely to have considered bug fixes or adding new functionality.

Question 20 gave respondents an opportunity to describe other ways they had contributed to the project, and 41 respondents (22.4%) took advantage of this. In most cases they provided additional detail about the activities they had chosen in response to question 19, but a small number of respondents identified activities that were not on the list. These were two people who had taken part in governance activities, one who had translated the interface to another language, one who ran a local user group, and one who had funded specific changes and then contributed them back to the community. Since these additional activities represented less than 3% of respondents, no further analysis was done of these responses.

7.2.7 *Impact of Training*

Table 30 on page 145 shows respondents' assessment of the impact of any training they had undertaken on their use of the software. There is a noticeable difference in the proportion of people who commented on formal training versus those who had undertaken self-study, with roughly 62% responding about formal training, and over 90% on self-study. This suggests that self-study is the most common way of learning to use a LIM-related FLOSS package. A majority of those who commented on formal training chose 'somewhat' or lower as their response with only 16.8% (19) indicating that training provided by outside organisations had affected their use of the software 'considerably' or 'extensively'. The results for in-house training are only slightly more positive, with 24.3% (26) choosing 'considerably' or 'extensively'. In contrast, approximately three-quarters of respondents felt that their self-study had a 'considerable' or 'extensive' impact, for both tutorials or online help, or manuals and other documentation.

Table 29: Activities carried out

ACTIVITY	n	%	TYPE
Used the software	171	93.4	Use
Installed the software	135	73.8	Code
Joined the project's email discussion list/forum	135	73.8	Community
Customised the software to meet local needs, either yourself, or by having a developer do so	129	70.5	Code
Promoted the project by talking about it to others, for example at a conference	123	67.2	Outreach
Evaluated existing software functionality	117	63.9	Use
Upgraded the software to a more recent release	114	62.3	Code
Asked a question on the project's email discussion/list forum	112	61.2	Community
Studied the source code to see how it works	106	57.9	Code
Written documentation to help others use the software	97	53.0	Community
Reported a bug to the system developers	92	50.3	Code
Answered a question on the project's email discussion list/forum	87	47.5	Community
Requested an enhancement from the system developers	88	48.1	Code
Contributed local changes back to the project	64	35.0	Code
Distributed the software to others	64	35.0	Community
Promoted the project by writing about it for publication	63	34.4	Outreach
Organised an event relating to the project, such as a meeting or conference	62	33.9	Outreach
Provided resources to support the project, such as hosting an email discussion list, forum or wiki	53	29.0	Resources
Written software to add new features	49	26.8	Code
Fixed one or more bugs	45	24.6	Code

Table 30: Impact of Training

TYPE	NOT AT ALL	VERY LITTLE	SOMEWHAT	CONSIDERABLE	EXTENSIVE
Provided by outside organisations (n = 113)	46.0% (52)	19.5% (22)	17.7% (20)	11.5% (13)	5.3% (6)
In-house (n = 115)	38.3% (44)	15.7% (18)	21.7% (25)	16.5% (19)	7.8% (9)
Self-study using tutorials or online help (n = 169)	3.6% (6)	9.5% (16)	18.3% (31)	36.1% (61)	32.5% (55)
Self-study using manuals or other documents (n = 164)	6.1% (10)	9.8% (16)	18.3% (30)	34.8% (57)	31.1% (51)

This table shows that self-study using tutorials or online help was more common, and perceived as more effective, than the other two types of training. This is shown clearly in Figure 6 on page 147.

Question 22 asked respondents to identify any other training they received that affected their use of the software. Very few people answered this question, with several commenting that no training was available. Others noted that training in using other software had been useful in understanding the FLOSS package they were commenting on, and several mentioned receiving demonstrations or training from other users of the software. Three mentioned formal education, such as an MLIS or an MSc in Computer Science.

7.2.8 *Satisfaction with software features*

Question 23 asked respondents to indicate their general level of satisfaction with 12 aspects of the project. Table 31 on the following page summarised the results. The results show that satisfaction with specific features varies. Some respondents are satisfied with a particular aspect of their chosen FLOSS project, while others are dissatisfied with it.

Figure 7 on page 148 shows the box plots of respondents' satisfaction with each of the features, while Figure 8 on page 148 shows a box plot for the mean satisfaction. The box plots show that individual ratings of satisfaction with project characteristics vary, with a number of outliers. The means for satisfaction range from 3.06 for documentation to 4.05 for reliability; the standard deviations are between .8 (functionality) and

Table 31: Satisfaction with project features

Feature	Not at all satisfied	Slightly satisfied	Somewhat satisfied	Quite satisfied	Completely satisfied
easy to add new features (n = 151)	6.6% (10)	20.5% (31)	29.1% (44)	29.8% (45)	13.9% (21)
easy to configure to meet local needs (n = 174)	6.3% (11)	16.7% (29)	31.0% (54)	33.3% (58)	12.6% (22)
easy to install (n = 156)	12.8% (20)	16.7% (26)	21.2% (33)	30.1% (47)	19.2% (30)
easy to learn (n = 179)	2.8% (5)	13.4% (24)	31.3% (56)	33.5% (60)	19.0% (34)
easy to use (n = 182)	2.7% (5)	9.9% (18)	30.8% (56)	34.6% (63)	22.0% (40)
free from bugs (n = 181)	5.5% (10)	11.6% (21)	26.5% (48)	47.0% (85)	9.4% (17)
functionality (n = 181)	1.7% (3)	3.3% (6)	23.2% (42)	56.4% (102)	15.5% (28)
helpfulness of community (n = 166)	1.2% (2)	7.2% (12)	16.9% (28)	39.8% (66)	34.9% (58)
quality of documentation (n = 181)	6.6% (12)	24.3% (44)	33.7% (61)	27.1% (49)	8.3% (15)
release frequency (n = 159)	3.8% (6)	13.8% (22)	30.2% (48)	32.7% (52)	19.5% (31)
reliability (n = 181)	2.2% (4)	3.9% (7)	13.8% (25)	47.0% (85)	33.1% (60)
security and access control (n = 165)	4.2% (7)	12.1% (20)	17.0% (28)	41.8% (69)	24.8% (41)

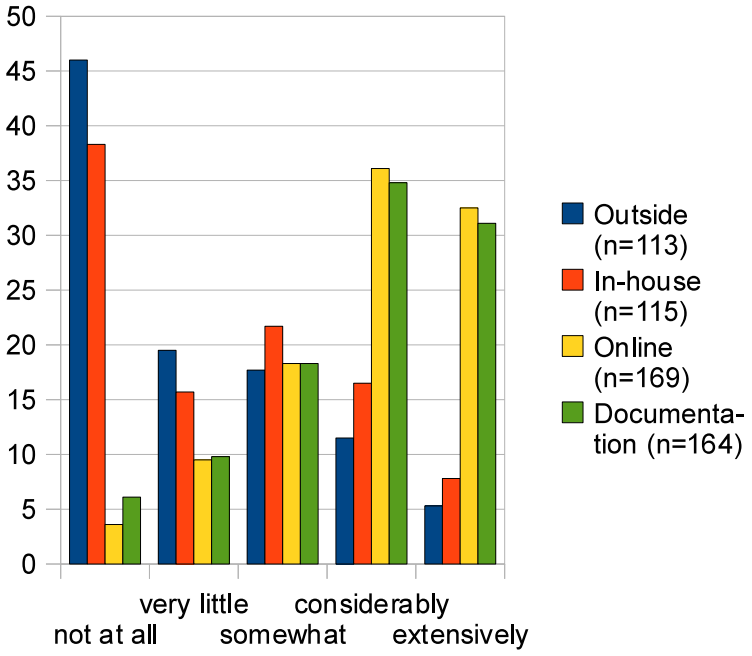


Figure 6: Impact of Training (%)

1.3 (easy to install). Overall, reliability, helpfulness of community, and security and access control are rated most highly, while documentation and ease of adding new features receive the lowest mean satisfaction scores. When the mean satisfaction is calculated from the individual scores, the resulting distribution is closer to normal, with only a single outlier as shown in Figure 8 on the next page.

7.2.9 *Perceived experience relative to others involved in the project*

Table 32 on page 149 summarises respondents' answers to question 24, which asked them to rate four aspects of their own experience relative to other people involved in the project. The overall results show that a majority of respondents felt they had as much or more experience than others. Using a FLOSS package is likely to require more experience than proprietary software, in part because the projects often lack good documentation, as discussed in Section 6.1.1 on page 127. Figure 9 on page 149 shows a box plot of the distributions for each aspect of experience, which shows that respondents with low levels of perceived experience relative to others using this type of software are outliers.

Although this weighting to more experienced respondents introduces a bias in the results, the means for experience ranged from 3.0 (as a member of a software development project) to 3.9 (using computers in general). This shows that overall, respondents had a range of different types of experience relative to other members of the FLOSS project community.

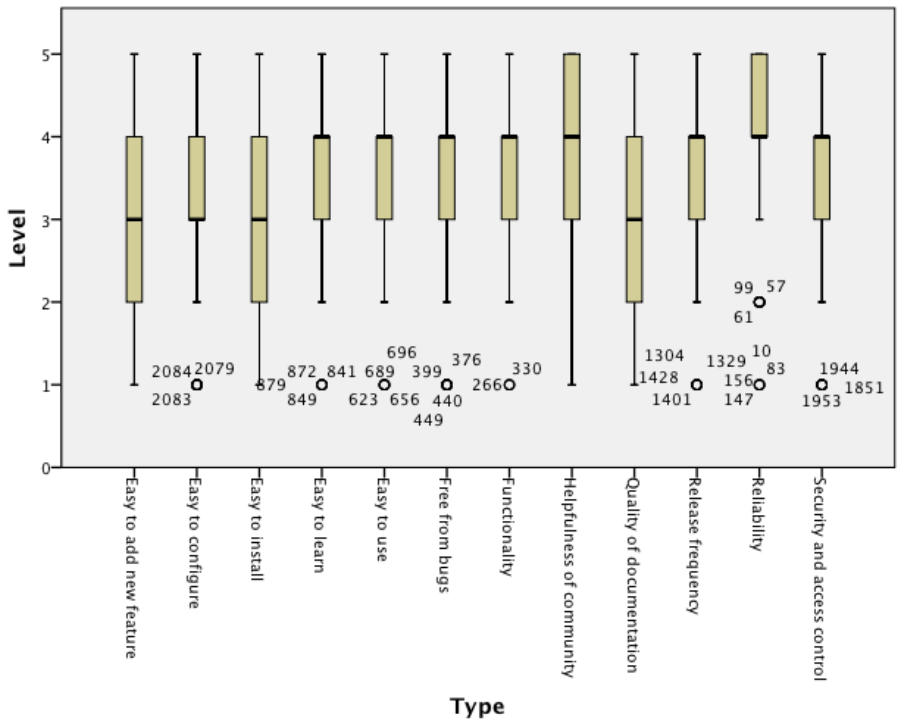


Figure 7: Box plots of satisfaction with project characteristics

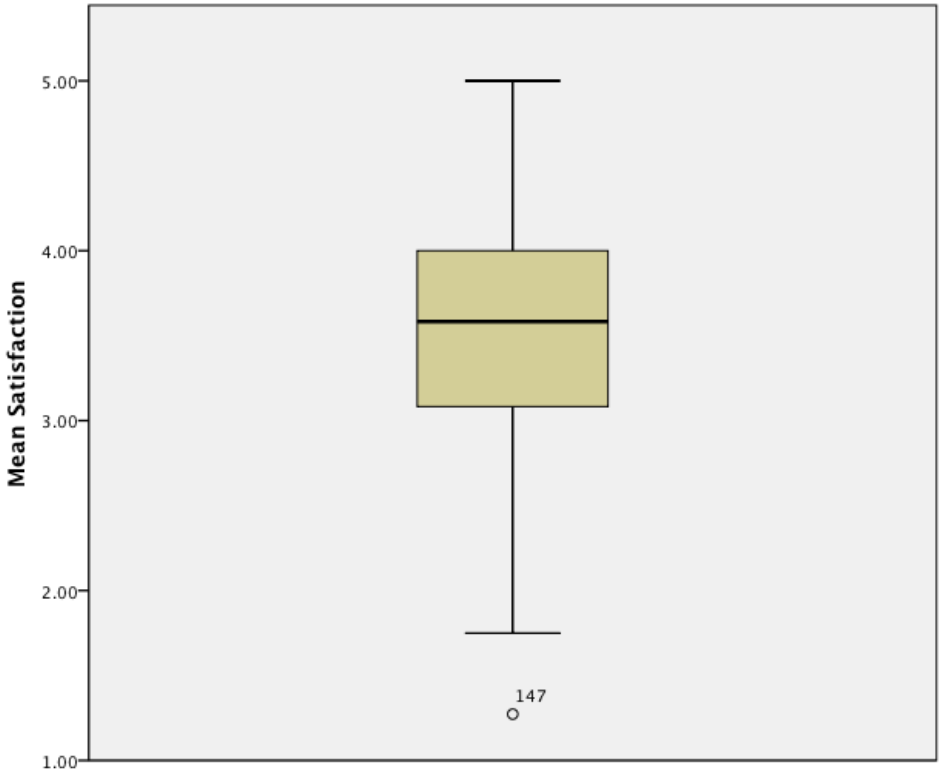


Figure 8: Box plot of mean overall satisfaction

Table 32: Experience relative to others involved in the project

Aspect	Considerably less	Slightly less	About the same	Slightly more	Significantly more
Using this type of software (n = 179)	6.7% (12)	17.9% (32)	27.9% (50)	24.0% (43)	23.5% (42)
Using this particular software package (n = 178)	10.1% 18)	15.1% (27)	27.5% (49)	21.9% (39)	25.3% (45)
Using computers in general (n = 177)	2.3% (4)	3.4% (6)	28.2% (50)	33.9% (60)	32.2% (57)
As a member of a software development project (n = 178)	23.0% (41)	16.3% (29)	21.3% (38)	20.2% (36)	19.1% (34)

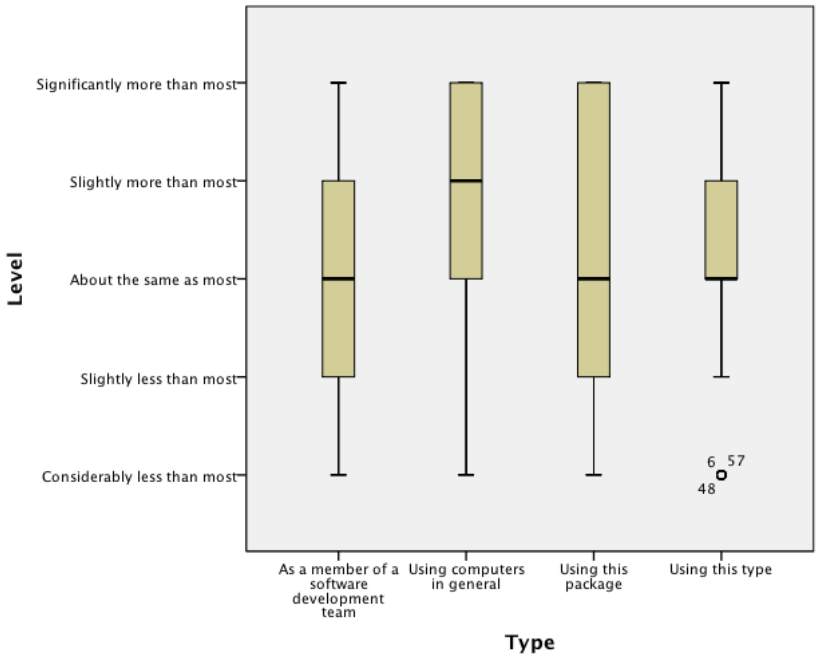


Figure 9: Box plot of experience relative to others involved in the project

7.2.10 *Characteristics of developer communication*

Respondents were asked to indicate their agreement with nine statements about the project's developers, and the results are summarised in Table 33 on the current page. The results show a strong positive rating, with fewer than 10% of respondents choosing 'Disagree' or 'Strongly disagree' for any of the statements.

Table 33: Developer communication characteristics

CHARACTERISTIC	STRONGLY DISAGREE	DISAGREE	NEUTRAL	AGREE	STRONGLY AGREE
Sensitive to other's needs (n = 159)	1.9% (3)	4.4% (7)	19.5% (31)	44.7% (71)	29.6% (47)
Get right to the point (n = 150)	0.7% (1)	6.0% (9)	19.3% (29)	38.7% (58)	35.3% (53)
Pay attention to what other people say (n = 155)	0.6% (1)	1.3% (2)	16.8% (26)	47.1% (73)	34.2% (53)
Deal effectively with others (n = 150)	0.7% (1)	5.3% (8)	21.3% (32)	45.3% (68)	27.3% (41)
Easy to understand (n = 155)	0.6% (1)	7.1% (11)	22.6% (35)	42.6% (66)	27.1% (42)
Say the right thing at the right time (n = 146)	2.1% (3)	4.8% (7)	34.9% (51)	37.0% (54)	21.2% (31)
Easy to communicate with (n = 152)	1.3% (2)	7.2% (11)	14.5% (22)	41.4% (63)	35.5% (54)
Respond to messages quickly (n = 142)	1.4% (2)	7.5% (11)	17.0% (25)	35.4% (52)	38.8% (57)
Express ideas clearly (n = 153)	0.7% (1)	3.9% (6)	24.2% (37)	39.9% (61)	31.4% (48)

7.2.11 *Project culture*

Question 26 covered several aspects of the project's culture, including encouragement to contribute to the project and its development cycle and plans. Table 34 on the facing page summarises the responses to this question. It is clear from the responses that people feel there is general encouragement for community members to contribute to the project, but there is also a sense that people who have contributed do not necessarily feel that their contributions are valued, since 'Neutral' is the most frequent response to this statement. In contrast, there is a strong sense that survey respondents value the contributions other people have

made to the project, with 91.9% (144) of respondents choosing 'Agree' or 'Strongly agree'.

Table 34: Project culture

CHARACTERISTIC	STRONGLY DIS-AGREE	DISAGREE	NEUTRAL	AGREE	STRONGLY AGREE
I feel encouraged to contribute (n = 178)	0.6% (1)	3.4% (6)	20.8% (37)	42.7% (76)	32.6% (58)
Anyone is encouraged to contribute (n = 177)	1.1% (2)	2.8% (5)	19.2% (34)	47.5% (84)	29.4% (52)
Only a few people are allowed to contribute (n = 173)	27.7% (48)	46.2% (80)	13.9% (24)	10.4% (18)	1.7% (3)
I find other people's contributions valuable (n = 176)	0.0% (0)	0.6% (1)	17.6% (31)	48.9% (86)	33.0% (58)
Other people find my contributions valuable (n = 171)	1.8% (3)	2.3% (4)	49.1% (84)	31.0% (53)	15.8% (27)
Information about future developments is easy to find (n = 176)	1.7% (3)	13.1% (23)	32.4% (57)	39.8% (70)	13.1% (23)
The future development plans are clear (n = 174)	2.3% (4)	10.9% (19)	40.2% (70)	33.9% (59)	12.6% (22)
The project has infrequent, formal releases of new versions (n = 170)	9.2% (16)	18.5% (32)	29.5% (51)	31.8% (55)	11.0% (19)
The project has frequent releases of incremental versions (n = 170)	1.2% (2)	11.8% (20)	30.0% (51)	37.6% (64)	19.4% (33)

7.2.12 *Influence on software features/functionality*

Questions 27 and 28 asked respondents to assess the amount of influence they have had on the software locally and on the version available for downloading by others. Table 35 on the next page presents the results. It shows that 65.1% (118) of users feel that they have had much or very much influence on the local version, with almost the same proportion (65.4%, or 119) assessing their impact on the shared version as 'very little or none'.

Table 35: Influence on software features/functionality

VERSION	NONE	VERY LITTLE INFLUENCE	MODERATE INFLUENCE	MUCH INFLUENCE	VERY MUCH INFLUENCE
Internal (n = 180)	5.0% (9)	12.8% (23)	16.7% (30)	27.9% (51)	37.2% (67)
Shared (n = 181)	34.8% (63)	30.9% (56)	19.3% (35)	9.4% (17)	5.5% (10)

7.2.13 *Perceived complexity*

The final question of the survey asked about the perceived system complexity and the perceived task complexity.

Perceived system complexity

Two sub-questions asked respondents to assess the complexity of the software's requirements and its design. The results are summarised in Table 36 on this page. They show that 43.9% (78) of the respondents judged the software they chose as the basis for their responses was as complex ('Agree' or 'Strongly agree'), while 36.5% (65) did not feel this was the case ('Disagree' or 'Strongly disagree'). Slightly more (48.4%, or 86) felt that it had a complex design, compared with 30.4% (54) who felt that it didn't.

Table 36: System complexity

	STRONGLY DIS-AGREE	DISAGREE	NEUTRAL	AGREE	STRONGLY AGREE
Complex requirements (n = 178)	6.2% (11)	30.3% (54)	19.7% (35)	34.3% (61)	9.6% (17)
Complex design (n = 178)	7.9% (14)	22.5% (40)	21.3% (38)	36.0% (64)	12.4% (22)

Task complexity

Question 29 also had two sub-questions that asked respondents to judge the complexity of the tasks they completed using the software. Table 37 on the next page summarises the results.

Overall the results show that a majority of respondents agreed or strongly agreed with the statements, indicating that they judge the tasks they carry out with the software to be straightforward.

Table 37: Task complexity

WHEN WORKING WITH THIS SOFTWARE ...	STRONGLY DIS-AGREE	DISAGREE	NEUTRAL	AGREE	STRONGLY AGREE
I have clear planned goals and objectives (n = 180)	0.6% (1)	4.4% (8)	13.9% (25)	55.6% (100)	25.6% (46)
I know what I am responsible for (n = 178)	0.0% (0)	0.6% (1)	12.9% (23)	59.0% (105)	27.5% (49)
I know exactly what other people expect of me (n = 178)	0.0% (0)	5.1% (9)	25.3% (45)	48.9% (87)	20.8% (37)

7.2.14 *Other comments*

The final question in the online survey gave respondents an opportunity to provide additional comments about their use of the software, their involvement in the project, or reasons for their satisfaction/dissatisfaction. Approximately 40% (72 of 183) of the survey respondents provided a comment, though five of these just said 'No' or 'None'. Of the respondents who provided meaningful data, four commented on the survey itself, with one saying it was too long and this was a barrier to completion (though this person did complete the survey), and one said that the survey seemed to assume that they "liked the software". Two pointed out that the question about developer communication was difficult to answer for FLOSS projects that had a modular structure with many thousands of contributors of themes and plug-ins. This is a legitimate criticism of the survey design; in addition, this comment is a good illustration of the range of FLOSS projects used in information management and the differences between them.

The remaining 63 comments had considerable variation, with the most common themes relating to the respondent's local implementation of the software, the respondent's role in the project, and the experience of dealing with the project community. The most common type of comment provided additional detail about the respondents' experiences of using the software and/or their perceptions of its strengths and weaknesses, particularly the aspects they were dissatisfied with, or felt could be improved. Others provided additional context for their responses, particularly to do with their role in the project and their satisfaction with it. One comment focused on the ability to customise the software, saying "The ability to customize/modify the source code to the software has made all the difference in the world. Like other packages, it only addressed 95% of our needs. But I could add the remaining 5% myself". Two respondents commented on aspects of the software/project that

had changed in ways that decreased their satisfaction with it, which confirms that their reported satisfaction was a cumulative judgement based on experience. One said that her responses would have been very different a year ago, because the developers had become less responsive over time.

The most dissatisfied respondent (who rated all of the project characteristics in question 23 'Slightly satisfied' or 'Not at all satisfied'), had not managed to install a working version of the software, despite following the instructions "to the letter". Other people with low satisfaction ratings commented on their perception that the software was complex, and difficult to change. One person noted that none of his questions to the project's email discussion list had been answered, and said that while he supported FLOSS in general, he would only recommend this project with several caveats, and that he was no longer considering using it himself. This shows the importance of participant satisfaction in promoting the software to potential new users, since people who are dissatisfied with it are likely to make only qualified recommendations.

7.3 SCALE CONSTRUCTION

This section presents the results of the tests for the reliability and validity of the measurement scales, and concludes with a description of which items were retained and how the final values were calculated to test the revised research model and hypotheses. These tests were carried out in order to ensure that the data had sufficient reliability to be used in the subsequent analysis, particularly the multiple regression. Some of the scales had been used in previous research, and had been shown to have Cronbach's alpha values close to or above the recommended threshold value of .70, as indicated in Section 3.4.3 on page 87. However, their reliability was retested for this research, to confirm that the scales retained their psychometric properties in this context.

Scale construction followed an iterative process. First, Cronbach's alpha was constructed for the initial scales, to test the internal reliability of each scale and identify any items that reduced reliability. The items included in the scale were modified as necessary to achieve a satisfactory initial value. Once the components of each scale had been identified, a factor analysis on the remaining items checked that each item had a satisfactory loading on its primary factor. Items that did not have satisfactory loadings on their primary factor were removed from the scales, and Cronbach's alpha was recalculated as necessary. This process was repeated until both approaches suggested that the data were as reliable as possible. Once this was done, the mean value was calculated for each scale from the remaining items, and this value was then used in the subsequent regression calculations.

7.3.1 *Knowledge and skills*

Question 6 measured participant knowledge and skills, asking respondents to rate their knowledge and skills with seven aspects of technology and software development, using a 5-point scale ranging from 'minimal' to 'extensive'. The initial Cronbach's alpha value based on standardised values was .83 which was higher than the recommended threshold of .70. Table 38 on the current page shows the corrected item–total correlations and estimated value of Cronbach's alpha if the item was removed from the scale. Knowledge and use of hardware has the lowest corrected item–total correlation, and therefore the results of the factor analysis for this item were examined closely to see how well it loaded on the overall factor. This is discussed in Sections 7.3.12 on page 167 and 7.3.15 on page 169.

Table 38: Knowledge and Skills Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Knowledge and use of hardware	.38	.82
Knowledge and use of operating systems	.58	.79
Knowledge and use of one or more programming languages	.44	.82
Knowledge and use of LIM application software	.52	.80
Ability to provide information to develop LIM application software	.70	.77
Ability to define LIM software requirements	.72	.76
Ability to assess LIM application software features	.67	.78

7.3.2 *Training*

Question 21 measured participant training, with 4 items that asked respondents to indicate how any training they received affected their use of the software using a 5-point scale ranging from 'not at all' to 'extensively'. Participants could also choose a 'n/a' response to indicate that the question did not apply to them. The initial Cronbach's alpha value based on standardised values was .52, which was lower than the

recommended threshold value of .70. Table 39 on this page shows the corrected item–total correlations and estimated value of Cronbach's alpha if the item was removed from the scale.

Table 39: Training Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Training provided by outside organisations	.34	.43
In-house training	.24	.52
Self-study using tutorials or online help	.37	.40
Self-study using manuals or other documentation	.32	.45

Because the corrected item–total correlations were all close to or below the recommended threshold value of .30 and and the alpha value would not have been raised by removing any items, the construct was removed from the model, and not used in subsequent analysis.

The scale used in this research was based on Guimaraes, Staples, and McKeen (2003), who used five questions that covered general courses at a community college or university, training provided by vendors or outside consultants, in-house courses, self-study using tutorials, and self-study using manuals and printed documents. Their results had an alpha of .83, which they interpreted as indicating suitable reliability. However, after considering the wording of the individual items and the results of this research, it became clear that the questions did not express a coherent underlying construct, since differences in location or source of training may not be related to training effectiveness. The data from the current study suggest that a different approach to measuring the effectiveness of training is needed, particularly for a FLOSS context. This is supported by the differences in response numbers shown in Table 30 on page 145, which suggests that survey respondents' training experience was too varied for this approach to measuring training effectiveness to be appropriate.

7.3.3 *Satisfaction*

Question 23 measured participant satisfaction, asking respondents to rate their satisfaction with 12 dimensions of the software and project using a 5-point scale ranging from 'not at all satisfied' to 'completely satisfied'. The initial Cronbach's alpha value based on standardised values

was .88, which was higher than the recommended threshold value of .70. Because the corrected item–total correlations were all higher than the recommended threshold value of .30 and the alpha value would not have been raised by removing any items, all 12 items were included in the scale. Table 40 on this page shows the corrected item–total correlations and estimated value of Cronbach's alpha if the item was removed from the scale.

Table 40: Satisfaction Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Easy to add new features	.63	.86
Easy to configure to meet local needs	.70	.86
Easy to install	.66	.86
Easy to learn	.54	.86
Easy to use	.61	.86
Free from bugs	.63	.86
Functionality	.66	.86
Helpfulness of community	.42	.88
Quality of documentation	.55	.86
Release frequency	.57	.87
Reliability	.44	.87
Security and access control	.47	.87

7.3.4 *Experience*

Question 24 measured participant experience, asking respondents to rate four aspects of their experience relative to their perceptions of other people involved in the project. The initial Cronbach's alpha value based on standardised values was .82, which was higher than the recommended threshold value of .70. Because the corrected item–total correlations were all higher than the recommended threshold value of .30 and and the alpha value would not have been raised by removing any items, all 4 items were included in the scale. Table 41 on the next page shows the corrected item–total correlations and estimated value of Cronbach's alpha if the item was removed from the scale.

Table 41: Experience Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Using this type of software	.75	.71
Using this software package	.70	.73
Using computers in general	.55	.80
As a member of a software development project	.57	.81

7.3.5 *Developer communication quality*

Question 25 measured developer communication quality, asking respondents to rate nine characteristics of the developers' communication with the community using a 5-point scale ranging from 'strongly disagree' to 'strongly agree'. The initial Cronbach's alpha value was .94, which was higher than the recommended threshold value of .70. Because the corrected item–total correlations were all higher than the recommended threshold value of .30 and and the alpha value would not have been raised by removing any items, all nine items were included in the scale. Table 42 on this page shows the corrected item–total correlations and estimated value of Cronbach's alpha if the item was removed from the scale.

Table 42: Developer Communication Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Sensitive to other's needs	.70	.94
Get right to the point	.78	.93
Pay attention to what others say	.83	.93
Deal effectively with others	.82	.93
Easy to understand	.74	.93
Say the right thing at the right time	.80	.93
Easy to communicate with	.80	.93
Respond to messages quickly	.69	.94
Express ideas clearly	.80	.93

7.3.6 *Process openness*

Question 26 items a–e measured process openness, asking respondents to rate the project's openness to contributions from themselves and others, using a 5-point scale ranging from 'strongly disagree' to 'strongly agree'. Item 26c was reverse coded for the scale evaluation. The initial Cronbach's alpha value was .76, which was higher than the recommended threshold value of .70. Table 43 on the current page shows the initial corrected item–total correlations and estimated value of Cronbach's alpha if the item was removed from the scale.

Table 43: Process Openness Scale Validation 1

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
I feel encouraged to contribute	.67	.65
Anyone is encouraged to contribute	.61	.67
Only a few people are allowed to contribute	.46	.73
I value other contributions	.53	.71
Others value my contributions	.35	.76

Although the corrected item–total correlations were all higher than the recommended threshold value of 0.30, the value for 'Others value my contributions' was .35, as shown in Table 43 on this page. Because this is close to the minimum acceptable value, Cronbach's alpha was recalculated for a modified version of the scale, leaving out 'Others value my contributions', resulting in a value of .77, a slight improvement on the earlier one. The recalculated results are shown in Table 44 on the next page. The discrepancy between the value respondents placed on other project participants' contributions and their perception that their own contributions are valued suggests that their contributions are valued more than they realise.

7.3.7 *Product openness*

Product openness was measured in Question 26 items f and g , which asked respondents to rate the availability of information about the future development plans of the project using a 5-point scale ranging from 'strongly disagree' to 'strongly agree'. The initial Cronbach's alpha value was .87, which was higher than the recommended threshold of .70. Table 45 on the following page shows the initial corrected item–total correlations, which were both higher than the recommended threshold

Table 44: Process Openness Scale Validation 2

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
I feel encouraged to contribute	.64	.66
Anyone is encouraged to contribute	.64	.66
Only a few people are allowed to contribute	.51	.74
I value other contributions	.47	.75

of 0.3. Because this scale had only two items, there is no meaningful value for Cronbach's alpha if the item was deleted.

Hulin and Cudeck (2001) suggested caution when using Cronbach's alpha with two-item scales, since it can underestimate the reliability of the scale. In this case the value is above the recommended threshold, implying that the scale is sufficiently reliable to be retained in the analysis.

Table 45: Product Openness Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Information about future development plans is easy to find	.77	n/a
The future development plans are clear	.77	n/a

7.3.8 *System complexity*

Question 29 items a and b measured system complexity, asking respondents to rate the complexity of the software using a 5-point scale ranging from 'strongly disagree' to 'strongly agree'. The initial Cronbach's alpha value was .81, which was higher than the recommended threshold of .70. Table 46 on the next page shows the corrected item–total correlations, which were both higher than the recommended threshold of 0.30. Because this scale had only two items, there is no meaningful value for Cronbach's alpha if the item was deleted.

With only two items, Hulin and Cudeck's cautions again needed to be considered (2001). Since the value of Cronbach's alpha is higher

than the recommended threshold value, the scale was considered to be sufficiently reliable to be retained in the analysis.

Table 46: System Complexity Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Software has complex requirements	.68	n/a
Software has a complex design	.68	n/a

7.3.9 *Task complexity*

Question 29 items c–e measured task complexity, which related to the degree of certainty and structure respondents had in the tasks they carried out with the software/project. These asked respondents to rate the complexity of the tasks they carry out using the software using a 5-point scale ranging from 'strongly disagree' to 'strongly agree'. The data were reverse-coded so that high values indicated more complexity. The initial Cronbach's alpha value was .72, which was higher than the recommended threshold of .70. Table 47 on this page shows the corrected item–total correlations and estimated value of Cronbach's alpha if the item was removed from the scale. Because the corrected item–total correlations were all higher than the recommended value of .30 and and the alpha value would not have been raised by removing any items, all 3 items were included in the initial factor analysis.

Table 47: Task Complexity Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Clear planned goals	.46	.71
Know what I am responsible for	.62	.53
Know what other people expect of me	.53	.62

7.3.10 *Initial factor analysis*

A factor analysis was done for the independent variables using Principal Component Analysis (PCA) with Oblimin rotation and Kaiser normali-

sation. The Kaiser-Meyer-Olkin (KMO) measure of sampling adequacy was .81, which is 'meritorious' (de Vaus 2002, page 137); in addition, the KMO values for individual items were all higher than 0.50, which indicates an adequate sample size for factor analysis (Field 2009, p.659). Bartlett's test of sphericity $\chi^2(465) = 2483.30, p < .001$, indicated that the correlations between items were sufficient for PCA to be an appropriate technique (Field 2009, p.671). The rotation converged in 12 iterations, identifying eight factors with eigenvalues greater than Kaiser's criterion of 1. These eight factors explained 73.5% of the variance. Table 49 on page 164 shows the structure matrix from the factor analysis, and Table 50 on page 165 has the pattern matrix. Field also recommended including both the pattern and structure matrixes when using oblique rotation, since relationships between factors can sometimes affect the pattern matrix (2009, p. 666). The structure matrix, which takes these relationships into account, therefore provides additional validation for the results of the factor analysis. Only factors with loadings greater than .40 have been included in the tables, following Field's recommendation (2009, p.669). To simplify the tables, the statements for each item have been assigned codes, which are shown in Table 48 on the next page.

Table 51 on page 166 contains a matrix showing the correlations between the eight components that were identified in the factor analysis. It shows that there were small correlations (all less than .30) between the eight factors. This confirms that using an oblique rotation technique was suitable for the data, because the factors are not independent (Field 2009, p.668). This correlation between the variables is not a significant concern, because they are all in the small to medium range.

Analysing the structure and pattern matrices showed that all items associated with the experience, developer communication, process openness, product openness, and system complexity scales had their maximum loading on the expected components, confirming that they were acceptable as factors. However, the knowledge and experience items were split between two components, with knowledge of hardware and knowledge of operating systems loading most strongly on one factor, and knowledge of LIM application software, system design, software requirements, and assessing LIM software on a second. Item KnowSkill3 relating to knowledge of one or more programming languages appeared to load on two separate factors, one that related to technical knowledge and skills, and the other to experience. This suggests that there are two underlying factors in the data, representing 'technical skills and knowledge' and 'LIM-specific skills and knowledge'.

Cronbach's alpha was recalculated for these as separate scales. Details of these calculations are in Sections 7.3.11 on page 166 and 7.3.12 on page 167.

Table 48: Item statement codes

CODE	STATEMENT
DevComm1	Sensitive to other's needs
DevComm2	Get right to the point
DevComm3	Pay attention to what other people say
DevComm4	Deal effectively with others
DevComm5	Easy to understand
DevComm6	Say the right thing at the right time
DevComm7	Easy to communicate with
DevComm8	Respond to messages quickly
DevComm9	Express ideas clearly
Exp1	Using this type of software
Exp2	Using this software package
Exp3	Using computers in general
Exp4	As a member of a software development project
ProdOpen1	Information about future development plans is easy to find
ProdOpen2	The future development plans are clear
ProcOpen1	I feel encouraged to contribute
ProcOpen2	Anyone is encouraged to contribute
ProcOpen3	Only a few people are allowed to contribute
ProcOpen4	I value other contributions
SysComplex1	Software has complex requirements
SysComplex2	Software has a complex design
TaskComplex1	Clear planned goals
TaskComplex2	Know what I am responsible for
TaskComplex3	Know what other people expect of me
KnowSkill1, TechKnow1	Knowledge and use of hardware
KnowSkill2, TechKnow2	Knowledge and use of operating systems
KnowSkill3, TechKnow3	Knowledge and use of one or more programming languages
KnowSkill4, LimKnowSkill1	Knowledge and use of LIM application software
KnowSkill5, LimKnowSkill2	Ability to provide information to develop LIM application software
KnowSkill6, LimKnowSkill3	Ability to define LIM software requirements
KnowSkill7, LimKnowSkill4	Ability to assess LIM application software features

Table 49: Factor analysis structure matrix

ITEM	1	2	3	4	5	6	7	8
DevCom3	.84							.41
DevCom4	.84							
DevCom6	.83							
DevCom9	.83							
DevCom7	.82							
DevCom2	.81							
DevCom5	.80							
DevCom1	.74							.52
DevCom8	.70							
KnowSkill6		.92						
KnowSkill7		.89						
KnowSkill4		.84						
KnowSkill5		.83						
SystComplex2			.90					
SystComplex1			.86					
ProdOpen2				.89				
ProdOpen1				.87				
TaskComplex3				-.58			.49	
Exp4					-.83			
Exp1		.50			-.81			
Exp2		.48			-.81			
Exp3					-.74			
KnowSkill3					-.57	.55		
KnowSkill1						.90		
KnowSkill2						.86		
TaskComplex1							.80	
TaskComplex2				-.47			.72	
ProcOpen4							-.60	.46
ProcOpen3								.80
ProcOpen2				.50				.80
ProcOpen1	.41			.47				.69

Table 50: Factor analysis pattern matrix

ITEM	1	2	3	4	5	6	7	8
DevComm5	.85							
DevComm6	.82							
DevComm2	.82							
DevComm9	.81							
DevComm3	.80							
DevComm7	.80							
DevComm4	.79							
DevComm1	.69							
DevComm8	.65							
KnowSkill6		.90						
KnowSkill7		.89						
KnowSkill4		.85						
KnowSkill5		.76						
SysComplex2			.88					
SysComplex1			.86					
ProdOpen2				.88				
ProdOpen1				.84				
TaskComplex3				−.48				
Exp4					−.83			
Exp3					−.76			
Exp2					−.75			
Exp1					−.74			
KnowSkill1						.90		
KnowSkill2						.78		
KnowSkill3						.41		
TaskComplex1							.79	
TaskComplex2							.62	
ProcOpen4							−.51	
ProcOpen3								.77
ProcOpen2								.70
ProcOpen1								.55

Table 51: Component correlation matrix

component	DevComm	KnowSkill1	SysComp	ProdOpen	Exp	KnowSkill2	TaskComp	ProcOpen
DevComm	1.00	.04	−.11	.25	−.05	.07	−.21	.19
KnowSkill1	.04	1.00	−.06	.06	−.27	.16	−.21	.07
SysComp	−.20	−.06	1.00	−.10	−.12	.06	.07	−.16
ProdOpen	.25	.06	−.10	1.00	−.01	.12	−.20	.19
Exp	−.05	−.27	−.12	−.01	1.00	−.23	.12	−.06
KnowSkill2	.07	.16	.06	.12	−.23	1.00	−.08	.13
TaskComp	−.21	−.21	.07	−.20	.12	−.08	1.00	-.20
ProcOpen	.19	.07	−.16	.19	−.06	.13	−.20	1.00

7.3.11 *Technical knowledge and skills scale validation*

This new scale had three items, all relating to technical knowledge and skills. The initial value of Cronbach's alpha was .77. Table 52 on the current page shows the corrected item–total correlations, which were all higher than the recommended threshold of 0.30. However, it also shows that removing the third item would increase reliability slightly, and so Cronbach's alpha was recalculated for the remaining two items.

Table 52: Technical Knowledge and Skills Scale Validation

item	corrected item–total correlation	Cronbach's alpha if item deleted
Knowledge and use of hardware	.53	.67
Knowledge and use of operating systems	.74	.49
Knowledge and use of one or more programming languages	.48	.80

The recalculated alpha was .80. Table 53 on the facing page shows the corrected item–total correlations for this revised calculation, which were both higher than the recommended threshold of 0.30. Because this scale had only two items, there is no meaningful value for Cronbach's alpha if the item was deleted. Because of this improvement in alpha, this third item was considered for removal from the scale. In particular, the results of the second iteration of the factor analysis, discussed in Section 7.3.15 on page 169, were examined closely to see how the factor loadings for the item relating to programming languages compared to

the other two items considered part of the technical knowledge and skills scale.

Table 53: Technical Knowledge and Skills Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Knowledge and use of hardware	.68	n/a
Knowledge and use of operating systems	.68	n/a

7.3.12 *LIM-specific knowledge and skills scale validation*

This new scale had four items, all relating to application software for the LIM field. The initial value of Cronbach's alpha was .89, which was higher than the recommended threshold value of .70. Because the corrected item–total correlations were all higher than the recommended threshold value of .30 and and the alpha value would not have been raised by removing any items, all four items were included in the scale. Table 54 on this page shows the corrected item–total correlations and estimated value of Cronbach's alpha if the item was removed from the scale.

Table 54: LIM-specific Knowledge and Skills Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Knowledge and use of LIM application software	.68	.89
Ability to provide information to develop LIM application software	.74	.87
Ability to define LIM software requirements	.85	.83
Ability to assess LIM application software features	.80	.85

7.3.13 *Process openness scale confirmation*

The factor analysis also showed that one item expected to be on the Process Openness scale (ProcOpen 4: I value other people's contributions) loaded most strongly on the Task Complexity scale, with a loading of .60, compared with .46 for process openness. This item was therefore dropped from the scale, and Cronbach's alpha was recalculated for the revised scale, giving a value of .76. The results of the recalculation are shown in Table 55 on the current page. These show that removing the third item would increase alpha slightly, and so Cronbach's alpha was recalculated for the remaining two items.

Table 55: Revised Process openness Scale Validation 1

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
I feel encouraged to contribute	.64	.59
Anyone is encouraged to contribute	.62	.62
Only a few people are allowed to contribute	.48	.79

The recalculated alpha was .79. Table 56 on this page shows the corrected item–total correlations for this revised calculation, which were both higher than the recommended threshold of 0.30. Because this scale had only two items, there is no meaningful value for Cronbach's alpha if the item was deleted. Because of this improvement in alpha, this third item was considered for removal from the scale. In particular, the results of the second iteration of the factor analysis in 7.3.15 on the next page were examined closely to see how the factor loadings were for the item 'Only a few people are allowed to contribute' compared to the other two items considered part of the Process openness scale.

Table 56: Revised Process openness Scale Validation 2

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
I feel encouraged to contribute	.65	n/a
Anyone is encouraged to contribute	.65	n/a

7.3.14 *Task complexity scale confirmation*

The factor analysis showed that one item expected to be on the Task Complexity scale (TaskComplex3: I know exactly what other people expect of me) loaded most strongly on the Process Openness scale, with a loading of .58, compared with .49 on Task Complexity. This item was therefore dropped from the scale, and Cronbach's alpha was recalculated for the revised scale, giving a value of .63, which is lower than the recommended threshold of .70. A summary of the results of the recalculation is in Table 57 on this page. Since the recalculation resulted in a lower value of alpha than the original, the wording and relevance of the items relating to task complexity were considered further, particularly because the items associated with this factor had the lowest communalities in the factor analysis. De Vaus (2002, p.137-138) defined communality as the extent to which the variance in the item is explained by the extracted factor. A high value indicates that the factor and the item have a good fit, while a low value indicates that the variable should be dropped from the analysis. The communalities for all of the other items in the factor analysis were above 0.50, with most above .70. However, the communalities for the items on the Task Complexity scale were all below 0.50, suggesting that the data from these items did not relate to a single underlying construct, and the construct was therefor removed from the model.

Table 57: Revised Task complexity Scale Validation

ITEM	CORRECTED ITEM–TOTAL CORRELATION	CRONBACH'S ALPHA IF ITEM DELETED
Clear planned goals	.46	n/a
Know what I am responsible for	.46	n/a

7.3.15 *Final factor analysis*

Field (2009, p.679) recommended repeating the factor analysis once the final scales have been determined to confirm that deleting items has not changed the factor structure. The results of this identified seven factors, as expected. The results showed that one of the items considered for removal, KnowSkill3, relating to knowledge and use of one or more programming languages, continued to have similar loadings on two different factors, and this was therefore dropped from the scale. The other item considered for removal, ProcOpen3, loaded cleanly on the

same factor as the other two ProcOpen items, and it was therefore retained in the subsequent analysis.

The final factor analysis had a Kaiser-Meyer-Olkin (KMO) measure of sampling adequacy of .803, which is 'meritorious' (de Vaus 2002, page 137); in addition, the KMO values for individual items were all higher than 0.50, which indicates an adequate sample size for factor analysis (Field 2009, p.659). Bartlett's test of sphericity $\chi^2(325) = 2224.29, p < .001$, indicated that the correlations between items were sufficient for PCA to be an appropriate technique (Field 2009, p.671). The rotation converged in 11 iterations, identifying seven factors with eigenvalues greater than Kaiser's criterion of 1. These seven factors explained 76.3% of the variance. Table 49 on page 164 shows the structure matrix from the factor analysis, and Table 50 on page 165 has the pattern matrix. Only factors with loadings greater than .40 have been included in the tables.

Table 58 on the next page and Table 59 on page 172 show the results of the final factor analysis. To make the distinctions between the factors clearer, the KnowSkill items have been renamed LIMKnowSkill and TechKnow.

The results show that all items have the highest loadings on their primary factors in both the pattern matrix and the structure matrix, though there are some that also loaded on other factors, particularly in the structure matrix. The primary loadings in the pattern matrix are all higher than .45, suggesting that the items for each construct have adequate convergent validity (Hair et al. 2006, p.128). In addition, the cross loadings in the structure matrix are all lower than .60, suggesting that the factors have adequate discriminant validity (Hair et al. 2006, p. 130).

Some of the items relating to developer communication also loaded on the product and process openness, and some of the items relating to experience also loaded on LIM skills and knowledge. It is reasonable to expect to find relationships between these constructs, since perceptions of product and process openness may be affected by the way the developers communicate with other members of the project. Experience is also likely to increase skills and knowledge. However, the tables also show that most of the items load onto a single component, suggesting that the other components are independent, as could be expected.

7.3.16 *Other variables*

The model included two additional components that were not treated as scales in the same way as other variables. These two items were Extent of Participation, and Perceived Influence.

Extent of participation was calculated as the total number of different activities the respondent had carried out. Question 19 had a list of the

Table 58: Final factor analysis pattern matrix

	1	2	3	4	5	6	7
DevComm5	.89						
DevComm2	.85						
DevComm7	.82						
DevComm6	.82						
DevComm9	.81						
DevComm4	.77						
DevComm3	.77						
DevComm1	.68						.41
DevComm8	.64						
LimKnowSkill3		.90					
LimKnowSkill4		.88					
LimKnowSkill1		.83					
LimKnowSkill2		.81					
SysComplex2			.92				
SysComplex1			.87				
ProdOpen1				.92			
ProdOpen2				.86			
Exper4					−.83		
Exper2					−.76		
Exper1					−.75		
Exper3					−.75		
TechKnow1						.93	
TechKnow2						.79	
ProcOpen3							.84
ProcOpen2							.67
ProcOpen1							.59

Table 59: Final factor analysis structure matrix

	1	2	3	4	5	6	7
DevComm5	.84						.42
DevComm2	.83						
DevComm7	.83						
DevComm6	.83						
DevComm9	.83						
DevComm4	.82						
DevComm3	.81						
DevComm1	.74						.55
DevComm8	.71						
LimKnowSkill3		.92					
LimKnowSkill4		.89					
LimKnowSkill1		.84					
LimKnowSkill2		.83					
SysComplex2			.92				
SysComplex1			.88				
ProdOpen1				.92			
ProdOpen2				.90			
Exper4		.51			−.84		
Exper2		.49			−.83		
Exper1					−.82		
Exper3					−.75		
TechKnow1						.94	
TechKnow2						.87	
ProcOpen3							.83
ProcOpen2				.51			.78
ProcOpen1	.40			.49			.72

Table 60: Final component correlation matrix

component	Dev Comm	LIM Know Skill	Sys Comp	Prod Open	Exp	Tech Know Skill	Proc Open
DevComm	1.00	.04	−.13	.29	−.03	.05	.22
LIMKnowSkill	.04	1.00	−.05	.03	−.29	.15	.08
SysComp	−.13	−.05	1.00	−.06	−.07	.08	−.19
ProdOpen	.29	.03	−.06	1.00	−.03	.08	.19
Exp	−.03	−.29	−.07	−.03	1.00	−.22	−.05
TechKnowSkill	.05	.15	.08	.08	−.22	1.00	.13
ProcOpen	.22	.08	−.19	.19	−.05	.13	1.00

20 most common activities identified in the first stage of this project, and respondents indicated which of these they had done.

Perceived influence was more challenging to measure, because participants could influence their local version of the software, or the community version shared with others. The difference between these values was expected to vary between participants, depending on where they were most active. As Table 35 on page 152 shows, most respondents indicated that they had more influence on their local version than on the shared version. Using a mean of the two values, as originally planned, would have tended to lower the values. Therefore this construct was measured using the maximum of the two values.

7.4 SCALE CHARACTERISTICS

Table 61 on the following page shows the number of items, Cronbach's alpha, mean, standard deviation, and the correlations between each scale and their one-tailed significance.

It shows that correlations between the variables are mostly low to moderate (between .10 and .29), with some that are moderate to substantial (between .30 and .49), according to de Vaus (2002, p.272). As expected, the most significant relationships are between the outcome variable and a majority of the predictor variables, with the strongest relationships between process openness and satisfaction (.50), and developer communication and satisfaction (.45). Both relationships are significant at $p<.01$ or lower. Technical knowledge and skills has the lowest correlation with satisfaction (.08, not significant), falling in the middle of the range de Vaus terms 'trivial' (2002, p.272). In addition, there are some moderate correlations between several of the predictor variables. The highest are between extent of participation and expe-

Table 61: Scale characteristics

	scale	no. of items	alpha	mean	s.d.	correlations[a] 1	2	3	4	5	6	7	8	9
1	Satisfaction (outcome variable)	12	.88	3.53	.71									
2	Technical knowledge and skills	2	.80	3.75	.88	.08								
3	LIM knowledge and skills	4	.89	3.95	.86	.14*	.25**							
4	Experience	4	.82	3.43	1.02	.19**	.29**	.37**						
5	Developer communication	9	.94	3.98	.74	.45**	.08	.05	.08					
6	Process openness	3	.79	4.03	.74	.50**	.17*	.18*	.16*	.38**				
7	Product openness	2	.87	3.49	.86	.38**	.07	.02	.11	.33**	.43**			
8	System complexity	2	.81	3.21	1.06	−.35**	.20**	−.14*	.03	−.18*	−.23**	.06		
9	Extent of participation	1	n/a	10.92	4.87	.30**	.23**	.32**	.47**	.15**	.22**	.15*	.03	
10	Maximum influence	1	n/a	3.87	1.13	.29**	.12*	.24**	.28**	.22**	.21**	.24**	−.07	.38**

[a] *=p<.05 **=p<.01 or lower (one tailed)

rience (.37), and product openness and process openness (.38), again significant at p<.01 or lower.

System complexity behaves differently from the other variables. It has a moderate negative relationship with satisfaction, and with process openness, but a positive correlation with technical knowledge and skills. These relationships are all significant at p<.01 or lower. It also has low negative correlation with LIM knowledge and skills, developer communication, and process openness, but no significant correlation with experience or product openness. This negative relationship with satisfaction is plausible, since people who perceive the software as complex are likely to be less satisfied than people who do not; it is also reasonable that people who feel that the project's activities are less open to outsiders will perceive increased complexity. Finally, people with higher technical skills are likely to be working on more complex software, so this relationship is also reasonable.

Table 61 on the preceding page also shows a number of low to moderate correlations between the other variables, some of which are also statistically significant. There are correlations between experience and knowledge and skills, between extent of participation and experience, and knowledge and skills. People develop their skills and knowledge in part by gaining experience with software, and these correlations confirm that relationship. Both process openness and product openness have a moderate positive correlation with developer communication, significant at p<.01 or lower. This is again logical, since perceptions of process and product openness are only likely to occur when there is good communication with developers. Extent of participation also has a moderate correlation with maximum influence, significant at p<.01 or lower. This confirms that participants need to be involved in the project in some way in order to influence its direction.

Although these correlations show that there are relationships between some of the predictor variables, none of these is in the substantial range, and the data were therefore judged to be suitable for regression analysis.

7.5 MODEL TESTING

Stepwise multiple regression was used to test the relationships between the variables and their influence on participant satisfaction. de Vaus explained that the goal of stepwise regression is to produce the simplest model that explains the variance in the dependent variable, and eliminate variables that have little or no effect (2002, p.361–362). Field noted that stepwise regression also shows the increase in variance each additional independent variable adds to the model (2009, p.213).

The first step was to ensure that the data met the requirements for carrying out multiple regression: first, that the sample size was large enough, and second, that multicollinearity was not an issue. Field (2009,

p.222) said that the number of independent variables and the goal of the regression calculation both determine the minimum sample size. For testing an overall model, the desired number is $50 + 8k$, where k is the number of variables; for testing the individual predictors, the minimum recommended sample size is $104 + k$. This research was interested in both, so the minimum number of cases needed is the larger of the two values, which is 122. The number of cases used in the analysis was 154, which was above this number.

Multicollinearity occurs when there are significant correlations between the predictor variables that could affect the results. The data were examined for potential multicollinearity problems using the process outlined in de Vaus (2002, p.345).

This began by examining the bivariate correlations between the predictor variables. Field suggested that correlations should be below .80 (2009, p.224). As Table 61 on page 174 shows, all of the correlations were low to moderate, and none was high or very high. The next step was to consider the diagnostic statistics resulting from the multiple regression calculation. The two key statistics are the variable inflation factor (VIF) and the tolerance measure. de Vaus recommended that VIF values should be less than 5, and tolerances higher than .20 in order to be confident that multicollinearity is not an issue (2002, p.345). All of the predictor variables met these requirements, as shown in Table 62 on the current page.

Table 62: Multicollinearity diagnostics

VARIABLE	VIF	TOLERANCE
Mean process openness	1.45	.69
Mean developer communication	1.25	.80
Mean complexity	1.12	.89
Extent of participation	1.07	.94
Mean product openness	1.34	.74
Maximum technical knowledge	1.15	.66
Mean LIM knowledge	1.29	.68
Mean experience	1.29	.68
Maximum influence	1.24	.69

The final step required testing the data to ensure that they had a normal distribution. This was done by examining the distribution pattern of the residual values for the variance in satisfaction that remained after the multiple regression calculation. It involved examining a histogram

and scattergram of the residuals (Figure 10 on this page and Figure 11 on the next page).

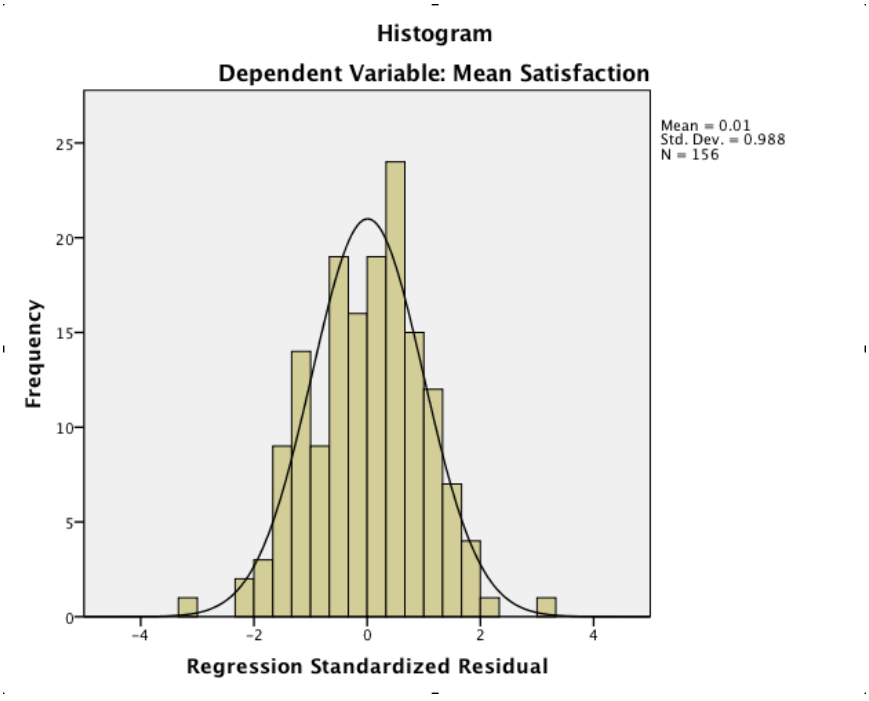


Figure 10: Residual histogram following stepwise regression

Both of these show that the residuals have a normal distribution, with no obvious skew. Further support for the independence of the predictors came from the value of the Durbin–Watson statistic, which tests whether the residual errors are correlated. The value was 2.09; Field says that if this value is close to 2, it indicates that the assumption that they are not correlated has been met (2009, p.236).

7.5.1 *Regression results*

The results of the stepwise regression showed that five variables explained 43.5% of the variance in satisfaction, as shown in Table 63 on the following page. The ANOVA output, which tests the significance level, showed that all results were significant at .000, meaning that the null hypothesis is not supported, and that the change in R^2 appears to be due to a real pattern in the data (de Vaus 2002, p.377).

Mean process openness was the most influential predictor of mean satisfaction, accounting for 25% of the variance, followed by mean

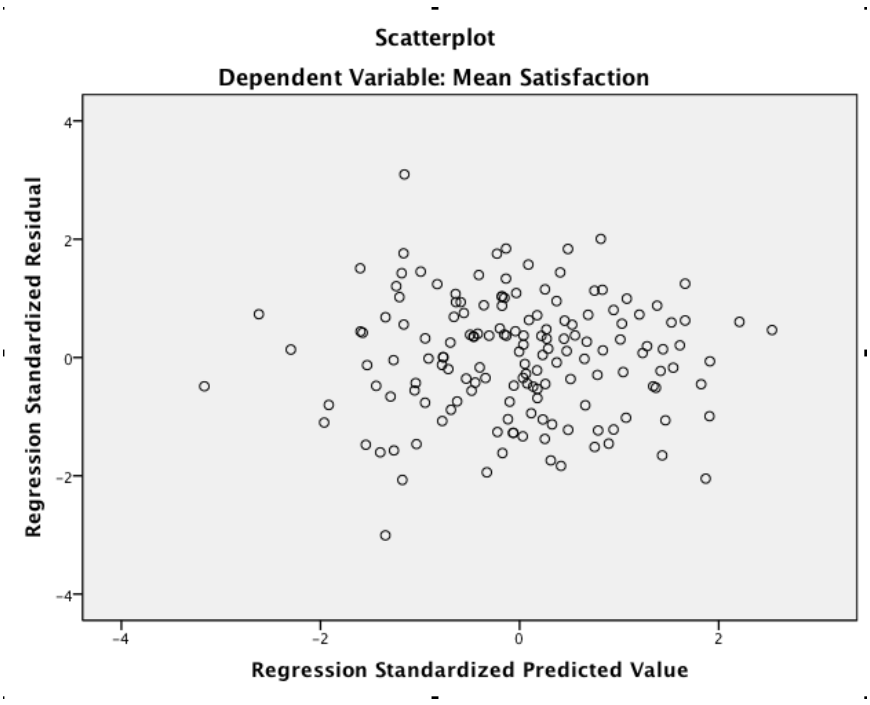


Figure 11: Residual scattergram following stepwise regression

Table 63: Model Summary

MODEL	R	R^2	ADJ. R^2	S. E. EST.	ΔR^2	Δ F	DF1	DF2	SIG. Δ F
1[a]	.50	.26	.25	.62	.26	52.06	1	152	.000
2[b]	.58	.34	.33	.58	.08	18.43	1	151	.000
3[c]	.62	.38	.37	.57	.05	11.62	1	150	.001
4[d]	.65	.42	.41	.55	.04	10.41	1	149	.002
5[e]	.67	.45	.44	.54	.03	8.01	1	148	.005

a (Constant), Process Openness
b (Constant), ProcessOpenness, Developer Communication
c (Constant), Process Openness, Developer Communication, Complexity
d (Constant), Process Openness, Developer Communication, Complexity, Extent of Participation
e (Constant), Process Openness, Developer Communication, Complexity, Extent of Participation, Product Openness

developer communication (a further 7.7%), and mean complexity (a further 4.5%). The final form of the regression equation is

$$\begin{aligned} Satisfaction = 1.51 &+ (.22 * ProcessOpenness) \\ &+ (.22 * DeveloperCommunication) \\ &- (.19 * Complexity) \\ &+ (.03 * ExtentOfParticipation) \\ &+ (.16 * ProductOpenness) \end{aligned}$$

Table 64 on the current page shows the coefficients, standard error, beta, T, and significance level for each variable.

Table 64: Coefficients

VARIABLE	B	S. E.	BETA	T	SIG.
(Constant)	1.51	.35		4.26	.000
Process Openness	.22	.07	.23	3.10	.002
Developer Communication	.22	.07	.22	3.29	.001
Complexity	−.19	.04	−.28	−4.40	.000
Extent of Participation	.03	.01	.20	3.20	.002
Product Openness	.16	.06	.20	2.83	.005

The beta values represent standardised regression coefficients for all variables, and are not dependent on the measurement units of the variables. They show that complexity has the largest impact on satisfaction, relative to the other variables, and product openness and extent of participation the smallest.

7.5.2 *Power analysis*

Table 63 on the facing page showed that adjusted R^2 was .44 for the final model. For this effect size, the power level is over .90, from the table in Ellis (2010, p.62), since the sample size of 154 was larger than the minimum of 61 needed for a power level of .90. This means that the sample size in this research was large enough to detect a minimum effect size of .25 at a power level of .90 (Ellis 2010, p.140), and the results can be considered to represent a real effect in the population.

7.5.3 *Moderated regression analysis*

The research model that was tested in this research identified three possible moderating variables: system complexity, process openness, and product openness. They were considered as independent predictor variables in the previous section. Table 64 on the current page shows

that b_2 was not zero for all three hypothesised moderator variables, which already suggests that they are not pure moderators. Nonetheless, the regression analysis was repeated, in order to test the hypothesised research model fully. This section presents the results of testing whether these variables acted as moderators of other relationships, rather than affecting satisfaction directly.

Testing for moderation involves comparing three regression equations (Sharma, Durand, and Gur-Arie 1981, p.295):

$$y = a + b_1 x \tag{7.1}$$

$$y = a + b_1 x + b_2 z \quad \rightarrow \quad R^2_{add} \tag{7.2}$$

$$y = a + b_1 x + b_2 z + b_3 xz \quad \rightarrow \quad R^2_{mult} \tag{7.3}$$

where y is the outcome variable, that is satisfaction, x is the predictor variable, and z the proposed moderator variable.

The first aspects of the results that needed to be considered were the b_i values. If z is a pure moderator, b_2 will be zero; if it is a pure predictor variable, with no moderating effects, b_3 will be zero. If neither b_2 nor b_3 is zero, then z may be what Sharma, Durand, and Gur-Arie term a quasi-moderator (1981, p.295), because it has an effect on both the outcome and predictor variables.

However, on their own these tests are not sufficient to determine whether a variable is acting as a predictor or a moderator. Carte and Russell (2003) said that to determine whether differences in the b_i values are significant, it is also necessary to consider whether the change in R^2 is significant. This is determined by calculating the value of F:

$$F_{(df_{mult}-df_{add},N-df_{mult}-1)} = \frac{(R^2_{mult} - R^2_{add})/(df_{mult} - df_{add})}{(1 - R^2_{mult})/(N - df_{mult} - 1)}$$

If F is significantly greater than 1, then z is a moderator variable.

The first step involved calculating the xz values as the product of the two independent variables being tested, and then centring the resulting values by subtracting their means. This was done in order to minimise the risk of multicollinearity problems (McKeen, Guimaraes, and Wetherbe 1994, p.441).

The second step involved carrying out a regression analysis to determine the values of a, b_1, b_2, and R^2 for the specific combinations of interest.

Finally, the individual cross-products xz were tested in a more complete model that incorporated the predictor variables identified in 7.5 on page 175, to see if their behaviour changed.

Impact of system complexity on extent of participation and satisfaction

The research model being tested in this research identified system complexity as a moderator of the relationship between participation and satisfaction. The results of the regression calculations are shown in Table 65 on the current page and Table 66 on this page.

Table 65: System complexity and participation model summary

MODEL	R	R^2	ADJ. R^2	S. E. EST.	ΔR^2	Δ F	DF1	DF2	SIG. Δ F
1[a]	.27	.08	.07	.67	.08	14.25	1	176	.000
2[b]	.43	.18	.18	.63	.11	23.65	1	175	.000
3[c]	.44	.19	.18	.63	.00	0.94	1	174	.335

a (Constant), Extent of Participation
b (Constant), Extent of Participation, Mean Complexity
c (Constant), Extent of Participation, Mean Complexity, Extent of Participation*Mean Complexity

Table 66: System complexity and participation regression coefficients

EQUATION	REGRESSION EQUATION	F CHANGE	SIG.
1	Satis = 3.12 + (.04*Extent of Participation)	14.25	.000
2	Satis = 3.77 + (.04*Extent of Participation) − (.22*Complexity)	23.65	.000
3	Satis = 4.38 + (.02*Extent of Participation) − (.32*Complexity) + (.01*Extent of Participation*Complexity)	0.94	.335

The results suggest that Model 2 and Equation 2 are a better fit to the data than Model 3 and Equation 3, since there is no significant change in R^2 between Model 2 and Model 3. In addition, the F change between models 2 and 3 is small, and Equation 3 is not statistically significant. The value of b_3 is also close to 0. Similar values were found when Extent of Participation*Complexity was tested in the larger model. These results suggest that in this context, System Complexity is acting as an independent variable, not a moderator variable.

Impact of process openness on activity count and satisfaction

The results of the regression calculations are shown in Table 67 on the next page and Table 68 on the following page.

The results suggest that Model 2 and Equation 2 are a better fit to the data than Model 3 and Equation 3, since there is no significant change in R^2 between Model 2 and Model 3. In addition, the F change between models 2 and 3 is small and Equation 3 is not statistically significant.

Table 67: Process openness and participation model summary

MODEL	R	R^2	ADJ. R^2	S. E. EST.	ΔR^2	Δ F	DF1	DF2	SIG. Δ F
1[a]	.27	.07	.07	.68	.07	13.90	1	176	.000
2[b]	.48	.23	.22	.62	.16	35.19	1	175	.000
3[c]	.48	.23	.22	.62	.00	0.38	1	174	.537

a (Constant), Extent of Participation
b (Constant), Extent of Participation, Mean Process Openness
c (Constant), Extent of Participation, Mean Process Openness, Extent of Participation*Mean Process Openness

Table 68: Process openness and participation regression coefficients

EQUATION	REGRESSION EQUATION	F CHANGE	SIG.
1	Satis = 3.13 + (.04*Extent of Participation)	13.90	.000
2	Satis = 1.72 + (.02*Extent of Participation + (.39*ProcessOpenness)	35.19	.000
3	Satis = 2.464 − (.01*Extent of Participation + (.29*ProcessOpenness) + (.01*Extent of Participation*ProcessOpenness)	0.38	.537

The value of b_3 is also close to zero. Similar values were found when Extent of Participation*ProcessOpenness was tested in the larger model. These results suggest that in this context, Process Openness is acting as an independent variable, not a moderator variable.

Impact of product openness on influence and satisfaction

The results of the regression calculations are shown in Table 69 on the current page and Table 70 on the facing page.

Table 69: Product openness and influence model summary

MODEL	R	R^2	ADJ. R^2	S. E. EST.	ΔR^2	Δ F	DF1	DF2	SIG. Δ F
1[a]	.28	.08	.08	.67	.08	15.334	1	174	.000
2[b]	.41	.17	.16	.64	.09	18.019	1	173	.000
3[c]	.41	.17	.15	.64	.00	0.052	1	172	.819

a (Constant), Maximum influence
b (Constant), Maximum influence, Mean Product Openness
c (Constant), Maximum influence, Mean Product Openness, Maximum influence*Mean Product Openness

The results suggest that Model 2 and Equation 2 are a better fit to the data than Model 3 and Equation 3, since there is no significant change

Table 70: Influence and product openness regression coefficients

EQUATION	REGRESSION EQUATION	F CHANGE	SIG.
1	Satis = 2.90 + (.17*MaxInfluence)	15.33	.000
2	Satis = 2.21 + (.14*MaxInfluence) + (.24*ProductOpenness)	18.02	.000
3	Satis = 2.48 + (.10*MaxInfluence) + (.20*ProductOpenness) + (.01*TotalActivity*ProcessOpenness)	0.05	.819

in R^2 between Model 2 and Model 3. In addition, the F change between models 2 and 3 is small and Equation 3 is not statistically significant. The value of b_3 is also close to zero. Similar values were found when TotalActivity*ProcessOpenness was tested in the larger model. These results suggest that in this context, Product Openness is acting as an independent variable, not a moderator variable.

7.5.4 *Type of participation and satisfaction*

The proposal for this thesis assumed that it would be possible to assign specific roles to survey respondents, in order to see if the factors that influence satisfaction vary between roles. As the results of the Stage 1b interviews suggested, and the data provided in response to questions 15 and 16 confirmed (discussed in Section 7.2.3 on page 140), there is no agreed set of roles that participants in a FLOSS project take on, and many people are involved in more than one type of activity. However, the attributes identified in Section 5.3 on page 121 provide a useful framework for comparing different aspects of participation and satisfaction, as an alternative to considering participation by roles.

The survey data made it possible to classify respondents according to three of these attributes: organisational focus, remuneration, and time commitment. The data on roles were not structured in a way that made it possible to distinguish between formal and informal project roles, meaning that the effect of differences in this attribute could not be tested further.

One point to note when reading the current section is that splitting the data file reduces the number of responses being tested for each value, which means that some of them fall below Field's recommended minimum sample size for multiple regression, discussed in Section 7.5 on page 175. This means that the effect sizes may be overstated, and caution is required in generalising these results.

Organisational focus and satisfaction

Respondents' organisational focus was determined by comparing their responses to questions 17 a and b. If their response indicated that they spent more hours per week working on their local implementation, they were classed as having a local focus; if they spent more hours per week working on the community version, they were classed as having a community focus; and if the hours were the same range, they were classed as 'equal'. Table 71 on this page shows the results of this classification. Since the number of people who spent equal amounts of time with both aspects of the project was small, this group was combined with those who spent more time working on the community project for the subsequent analysis. The resulting two groups were named 'local' and 'non-local'.

Table 71: Organisational focus

FOCUS	N	%
Local	104	56.8
Equal	12	6.6
Project	67	36.6
Total	183	100.0

Once the two groups were classified, a regression analysis was carried out by splitting the data based on the value of organisational focus. To make it easy to compare the two groups, the five factors identified in Section 7.5.1 on page 177 were entered as a single block for the analysis

Table 72: Organisational focus model summary

MODEL	R	R^2	ADJ. R^2	S. E. EST.	ΔR^2	Δ F	DF1	DF2	SIG. Δ F
1[a]	.60	.36	.32	.56	.36	11.03	5	98	.000
2[b]	.65	.43	.39	.54	.43	10.84	5	73	.000

a Local focus (n=104)
b Non-local focus (n=79)

The Durbin–Watson values for the two models were 2.022 and 2.035 respectively; according to Field (2009, p.200) their closeness to 2.0 indicates that the residuals are uncorrelated. Other indicators of the model's reliability are also appropriate, since all VIFs were close to 1.0 and all tolerances were above 0.50.

The results show that the model is a slightly better fit for survey respondents with a non-local organisational focus, accounting for 38.7% of the variance in satisfaction vs 32.7%. In addition, the contribution of

Table 73: Local focus coefficients

VARIABLE	B	S. E.	BETA	T	SIG.
(Constant)	2.00	.49		4.07	.000
Process Openness	.13	.09	.14	1.47	.145
Developer Communication	.29	.10	.27	2.84	.005
Complexity	−.22	.06	−.34	−3.81	.000
Activity Count	.01	.01	.08	.95	.345
Product Openness	.11	.07	.14	1.53	.13

Table 74: Non-local focus coefficients

VARIABLE	B	S. E.	BETA	T	SIG.
(Constant)	1.15	.47		2.43	.018
Process Openness	.33	.10	.33	3.28	.002
Developer Communication	.13	.08	.15	1.58	.119
Complexity	−.10	.06	−.15	−1.62	.111
Activity Count	.03	.01	.21	2.17	.033
Product Openness	.19	.08	.24	2.36	.021

each component of the model varies with the organisational focus, with perceived complexity and developer communication having the most impact (based on the beta values) for respondents with a local focus. Process and product openness were the most important predictors of overall satisfaction for respondents with a non-local focus.

Remuneration and satisfaction

Respondents' remuneration focus was determined from their response to question 18, which asked what proportion of their time spent working on the project was part of their paid employment. Respondents were classified into two groups: those who spent more than 50% of their time working on the project as part of their paid employment, and those who spent 50% or less. Table 75 on the following page shows the results of this classification. These indicate that a significant proportion of respondents contributed to the project on their own time, rather than as part of their employment, suggesting that they have a personal interest in the project's success.

Once the two groups were classified, a regression analysis was carried out by splitting the data based on the value of remuneration category. The five factors identified in Section 7.5.1 on page 177 were entered as a single block for the analysis, to make it easy to compare the two groups.

Table 75: Remuneration category

REMUNERATION CATEGORY	N	%
50% or less	79	43.2
More than 50%	104	56.8
Total	183	100.0

Table 76: Remuneration category model summary

MODEL	R	R^2	ADJ. R^2	S. E. EST.	ΔR^2	Δ F	DF1	DF2	SIG. Δ F
1[a]	.67	.44	.41	.54	.44	11.7	5	73	.000
2[b]	.58	.34	.31	.58	.34	10.19	5	98	.000

a 50% or less (n=79)
b More than 50% (n=104)

The Durbin–Watson value for the two models were 1.746 and 2.044 respectively; according to Field (2009, p.200) their closeness to 2.0 indicates that the residuals are uncorrelated. Other indicators of the model's reliability are also appropriate, since all VIFs were close to 1.0 and all tolerances were above 0.50.

Table 77: 50% or less coefficients

VARIABLE	B	S. E.	BETA	T	SIG.
(Constant)	1.23	.50		2.45	.017
Process Openness	.16	.11	.16	1.53	.130
Developer Communication	.28	.10	.28	2.88	.005
Complexity	−.16	.07	−.22	−2.33	.023
Activity Count	.03	.01	.21	2.28	.025
Product Openness	.22	.08	.29	2.73	.008

The results show that the model is a somewhat better fit for survey respondents who are paid for less than 50% of the time they work on the project, accounting for 40.7 % of the variance in satisfaction vs 30.8%. In addition, the contribution of each component of the model varies with the remuneration category, with developer communication and product openness shown as the most important predictors of overall satisfaction for respondents who are paid for a smaller proportion of their time. In contrast, complexity and process openness have the largest impact for respondents who are paid for most of their time working on the project.

Table 78: More than 50% coefficients

VARIABLE	B	S. E.	BETA	T	SIG.
(Constant)	2.06	.48		4.34	.000
Process Openness	.26	.09	.27	2.98	.004
Developer Communication	.14	.09	.15	1.64	.109
Complexity	−.20	.06	−.30	−3.56	.001
Activity Count	.02	.01	.17	2.00	.049
Product Openness	.06	.07	.08	.85	.396

Time commitment and satisfaction

Respondents' time commitment to the project was determined from their responses to question 17 a and b, which asked them to categorise the number of hours per week they spent working on their local implementation of the software, and the same for the shared version. Individual responses were converted to the mid-range of the category, and added together to get a number approximating their total hours per week. Based on this, respondents were ranked and then classified into two equal size groups based on the median value.

Once the two groups were identified, a regression analysis was carried out by splitting the data based on the value of remuneration category. The five factors identified in Section 7.5.1 on page 177 were entered as a single block for the analysis, to make it easy to compare the two groups.

Table 79: Time commitment model summary

MODEL	R	R^2	ADJ. R^2	S. E. EST.	ΔR^2	Δ F	DF1	DF2	SIG. Δ F
1[a]	.59	.35	.32	.55	.35	9.28	5	85	.000
2[b]	.64	.40	.37	.57	.40	11.68	5	86	.000

a Below median (n=91)
b Above median (n=92)

The Durbin–Watson value for the two models were 1.676 and 2.050 respectively; according to Field (2009, p.200) their closeness to 2 indicates that the residuals are uncorrelated. Other indicators of the model's reliability are also appropriate, since all VIFs were close to 1.0 and all tolerances were above 0.50.

The results show that the model is a somewhat better fit for survey respondents who are above the median in terms of their time commitment, accounting for 37.0% of the variance in satisfaction vs 31.5%. In addition, the contribution of each component of the model varies with the time commitment category. Complexity was the most important pre-

Table 80: Below median

VARIABLE	B	S. E.	BETA	T	SIG.
(Constant)	1.70	.50		3.39	.001
Process Openness	.26	.10	.26	2.53	.013
Developer Communication	.18	.10	.18	1.83	.071
Complexity	−.18	.06	−.26	−2.84	.006
Activity Count	.03	.01	.22	2.40	.019
Product Openness	.11	.08	.13	1.35	.180

Table 81: Above median

VARIABLE	B	S. E.	BETA	T	SIG.
(Constant)	1.50	.47		3.17	.002
Process Openness	.19	.09	.21	2.18	.032
Developer Communication	.20	.09	.20	2.22	.029
Complexity	−.16	.06	−.23	−2.60	.008
Activity Count	.03	.01	.19	2.20	.031
Product Openness	.18	.08	.22	1.36	.021

dictor for both categories, but process openness was more important for respondents below the median (i.e., those who spent less time) than for those above it. Product openness was more important for respondents above the median than those below the median.

7.6 HYPOTHESIS TESTING

This section discusses whether the hypotheses proposed in the research model are confirmed by the data, based on the results of the regression analysis presented in Section 7.5 on page 175.

H1 The higher the perceived participant influence, the higher the participant satisfaction.

NOT SUPPORTED The results of the multiple regression analysis showed that perceived participant influence had no significant effect on participant satisfaction. Therefore, it was not a predictor variable.

H2 The higher the perceived quality of developer communication, the higher the participant satisfaction.

SUPPORTED The results of the multiple regression analysis showed that perceived developer communication quality has a standardised beta coefficient of .22, significant at $p<.01$. This

means that the hypothesis is supported, with perceived developer communication quality accounting for 8% of the variation in satisfaction.

H3 There is a positive relationship between participant skills and knowledge and participant satisfaction.

NOT SUPPORTED The factor analysis showed that the questions about participant skills and knowledge represented two underlying constructs, technical knowledge and skills and LIM knowledge and skills. Neither had any significant effect on participant satisfaction in the regression equation.

H4 There is a positive relationship between participant training and participant satisfaction.

UNTESTED This hypothesis could not be tested because the results of the Cronbach's alpha and factor analysis showed that this construct lacked reliability, and was therefore not able to be included in the statistical analysis.

H5 There is a positive relationship between participant experience and participant satisfaction.

NOT SUPPORTED The results of the multiple regression analysis showed that participant experience had no significant effect on participant satisfaction.

H6 There is a positive relationship between the extent of participation and participant satisfaction.

SUPPORTED The results of the multiple regression analysis showed that extent of participation has a standardised beta coefficient of .20, significant at $p<.01$. This indicates that the hypothesis is supported, with extent of participation accounting for 4% of the variation in satisfaction.

H7A The greater the perceived system complexity, the greater the relationship between extent of participation and participant satisfaction.

NOT SUPPORTED The results showed that system complexity is an independent predictor variable for satisfaction, rather than a moderator variable. This means that they support a revised hypothesis that there is a negative relationship between perceived system complexity and participant satisfaction (H7C).

H7B The greater the perceived task complexity, the greater the relationship between extent of participation and participant satisfaction.

UNTESTED H7B could not be tested, because the results of the Cronbach's alpha and factor analysis showed that this construct lacked reliability, and was therefore unable to be included in the statistical analysis.

H8A The higher the perceived process openness, the greater the relationship between extent of participation and participant satisfaction.

NOT SUPPORTED The results show that perceived process openness is not a moderator for the relationship between extent of participation and satisfaction.

H8B There is a positive relationship between process openness and participant satisfaction.

SUPPORTED The results of the multiple regression analysis showed that process openness had a standardised beta coefficient of .23, significant at $p<.01$. This indicates that the hypothesis is supported, with process openness accounting for 25% of the variation in satisfaction.

H9A The higher the perceived product openness, the greater the relationship between perceived participant influence and participant satisfaction.

NOT SUPPORTED The results show that perceived product openness does not moderate the relationship between extent of participation and satisfaction, but that it acts as a predictor variable for satisfaction.

H9B There is a positive relationship between product openness and participant satisfaction.

SUPPORTED The results of the multiple regression analysis showed that product openness had a standardised beta coefficient of .20, significant at $p<.01$. This indicates that the hypothesis is supported, with product openness accounting for 3% of the variation in satisfaction.

7.7 REVISED RESEARCH MODEL

This section presents a revised research model, based on the results of the regression testing presented in Section 7.5 on page 175. Figure 12 on the next page illustrates the model, and shows the hypotheses, beta values, and adjusted R^2. Overall the model accounts for 44% of the variance in survey respondents' satisfaction.

The revised model suggests that a more appropriate hypothesis for the relationship between system complexity and satisfaction is:

H7C (new) The greater the perceived system complexity, the lower the participant satisfaction.

The results showed that system complexity had a standardised beta coefficient of $-.28$, significant at $p<.01$. System complexity accounts for 5% of the variation in satisfaction.

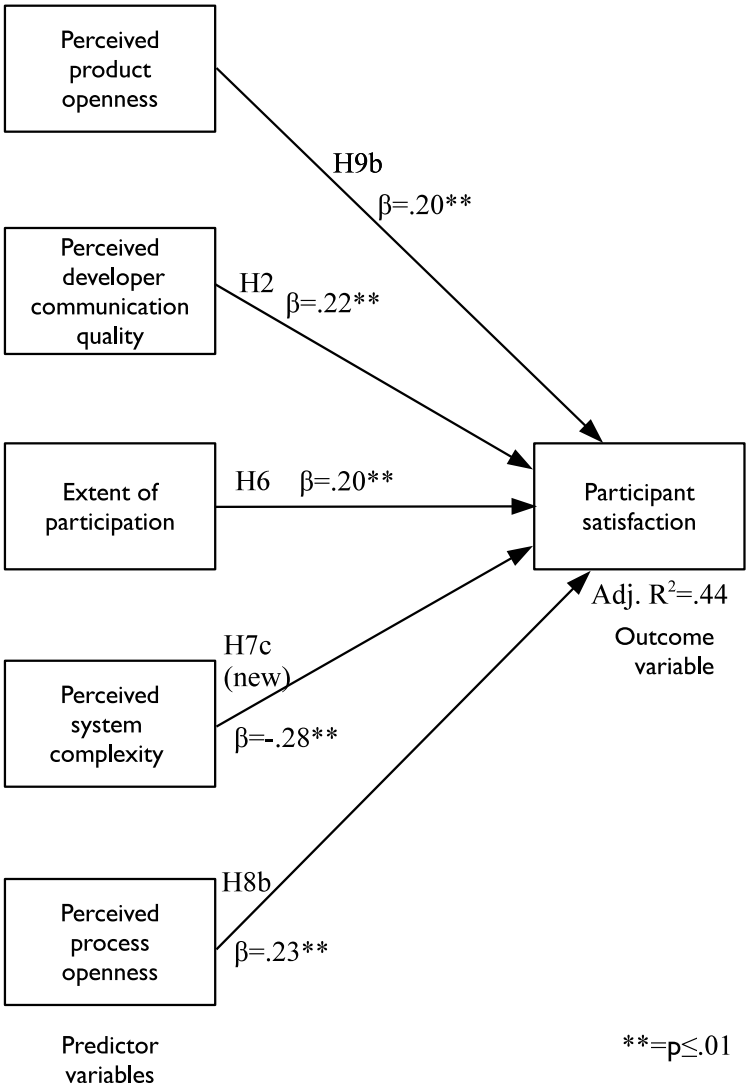


Figure 12: Revised research model

7.8 SUMMARY

This chapter presented the results of the quantitative analysis of the web-based survey, including testing the reliability and validity of the constructs and revising them as necessary. It then described the results of the stepwise multiple regression, which showed that five constructs accounted for 44% of the variability in satisfaction. Perceived process openness was the single best predictor of satisfaction, but perceived complexity had the largest effect with a beta of −.28 ($p \leqslant .01$). The results of the moderated regression analysis showed that none of the proposed moderator variables had a significant moderating effect, and that they were acting as predictor variables.

8

DISCUSSION AND INTERPRETATION OF FINDINGS

8.1 INTRODUCTION

This chapter discusses the research findings and their significance. It begins with a discussion of the user-centric model of a FLOSS project and its implications. This is followed by a discussion of the results of the quantitative survey, including the measurement of the constructs, the research model, and the hypothesis testing. The chapter discusses the revised revised research model presented in Figure 12 on page 191, followed by a discussion of the results of the hypothesis testing and their implications. It concludes with additional findings that were not part of the original research objectives, but emerged from the demographic characteristics of the survey respondents.

8.2 A USER-CENTRIC VIEW OF A FLOSS PROJECT

Section 5.4.1 on page 123 presented a user-centric view of the activities involved in a FLOSS project. In contrast with the typical code- or developer-centric models found in the information systems literature, this alternative model presents a broader view of the activities project participants carry out to achieve the project's goals. It is an example of what Gregor termed a level I theory, intended to describe and analyse a phenomenon (2006, p.620).

8.2.1 *Perspective is important*

Taking an alternative perspective can change the way people perceive a phenomenon. This is true for the physical environment, where people who climb to the top of a hill have a different view than people at a lower level. In a similar vein, in Edwin Abbott's *Flatland*, a two-dimensional square has difficulty understanding the concept of three dimensions until it is shown that the one-dimensional point cannot conceive of two dimensions (1953).

These examples show that basing a description of a phenomenon on limited data may result in a flawed or partial understanding of the phenomenon. This partial understanding can be useful as long as its limitations are recognised and acknowledged. For example, the code-centric model of a FLOSS project structure may be more relevant for projects where the developers are also users of the software. One benefit

of the alternative model proposed here is that it places the earlier code-centric descriptions of FLOSS project structures in a broader context.

Gregor (2006, p.623–624) notes that descriptive theories are useful for classifying and comparing phenomena. The proposed user-centric model provides a number of options for this. The simplest would be to use it as a framework for measuring the extent of FLOSS project participant activity in each of the six categories: code, management, community, sponsorship, outreach, and governance. This would show whether any types of activity are being overlooked. As Section 2.3.1 on page 18 showed, most current FLOSS research places emphasis on the code and governance categories, with the result that the other types of activity are given little attention. This has the potential to create the misleading impression that only people with good coding skills can contribute to a project, when other types of activity, such as sponsorship and outreach, are also important to long-term project sustainability. The code-centric model also conveys the impression that what participants value is contributing code; Studer's research, which asked participants in the KDE project to indicate how much prestige they assigned to 11 different activities, found that while code was assigned the most prestige, coordination, which falls in the 'manage' category, was also highly ranked (2007). This suggests that the FLOSS contribution model presented here is a better match to the way project participants view participation options.

In addition, grouping the activities into the three categories of project fitness, project viability, and project spirit provides a higher-level framework for classifying and comparing FLOSS projects based on the extent of participant activities in these groups. If projects are studied over time, this has the potential to aid in identifying different patterns of FLOSS project evolution. This would also allow researchers to examine the relationship between the patterns of activity across the categories and the extent of project growth. For example, projects with weak or non-existent governance may be less productive because participants' energy could be deflected into repetitive arguments about processes or acceptable behaviour, rather than into constructive contributions in any of the six outer categories.

8.2.2 *Terminology matters*

In 1943, George Orwell wrote an essay titled *Politics and the English Language*, in which he argued that writers must choose their words carefully to encourage clear thinking. In a similar vein, Richard Stallman discourages the use of the term 'intellectual property', which he argued causes confusion by grouping three dissimilar legal concepts (copyright, trademarks, and patents) and thereby suggesting that they are equivalent (2010). The research literature on FLOSS projects at times

shows similar confusion, for example, by using the term 'developers' to refer to all members of a project's community, or by implying that all contributors are unpaid volunteers.

The results of this research have shown that participation includes seven types of activity, only one of which involves interacting directly with the code. The results of both stages of this research show that the term 'developers' applies only to a subset of a project's total community. By using this term, researchers imply that they are only interested in one type of participant—the developers who work with the project's code. If that is indeed the case, then the term 'developers' is appropriate, but if it is not, their results may be biased because they do not include the non-developers' perspective. If the target population for the research is the wider project community, a more inclusive term like 'participant' or 'contributor' would be more appropriate. This research followed that practice, which may be a factor that led to the higher proportion of women among its survey respondents than in previous FLOSS surveys. In addition, using the phrase 'open source' reinforces a developer-centric view, because it is natural to assume that only developers are interested in access to source code, on the grounds that users lack the skills to understand or change it, and are generally not interested in doing so.

A second term that is often used in FLOSS research literature is 'volunteer', which can also be misleading. A strict interpretation of the word 'volunteer' implies that it refers only to the extent to which participants choose to work on FLOSS projects, as opposed to doing so because they are required to as part of their paid employment. However, in modern use the term 'volunteer' also implies that contributors are not paid for their work; when people speak of the 'voluntary sector', they generally mean people who donate their time free of charge to charitable projects.

Most research articles imply that FLOSS 'volunteers' are also unpaid, illustrated by the following quotes: "The projects are distinctive in that they rely on the efforts of a community of volunteer software users and developers instead of paid managers and employees" (Shah 2006, p.1000); "OSS [open source software] is usually developed by a community of voluntary participants" (Xu, Jones, and Shao 2009, p.151); "Participation is voluntary, and participants do not receive direct compensation for their work" (Hars and Ou 2002, p.25); and "a central dilemma of OSS development is what motivates developers to contribute their time and skills 'for free' " (Markus 2007, p.153).

These generalisations place emphasis on the extent to which FLOSS participants are *not* paid for their work on the projects. A majority of respondents to the web-based survey in this research were paid for at least some of the time they worked on the FLOSS project, with only 14.9% receiving no payment. In most cases they would have been paid by their employer, because they participate in the FLOSS project as part of their employment during their regular work hours. However, only

40.3% were paid for over 80% of their time, suggesting that many were not paid for *all* of the time they spent contributing to the project. This is a characteristic behaviour of many professionals, who often work more than a standard 40-hour week, without direct remuneration for their 'overtime'. In addition, many professionals engage in activities which are not part of their normal duties in order to extend their skills and knowledge, and respondents may see their involvement in a FLOSS project as a form of professional development.

Because of differences in the way they collected their data, it is difficult to determine how many respondents to previous surveys were paid for their work on FLOSS projects. David and Shapiro (2008) reported on employment status, but did not associate this with work on a specific FLOSS project. Their findings showed that 72% of their respondents were employed, 23% were students, and 5% were unemployed (2008, p.372). This distribution pattern is similar to previous studies, in particular Ghosh et al.'s 2002 study, which showed that 79% of respondents were employed, 17% students, and 4% unemployed. Kroah-Hartman, Corbet, and McPherson's 2009 study of contributions to the Linux kernel concluded that "over 70% of all kernel development is demonstrably done by developers who are being paid for their work" (p.10). These examples, and the findings from this research, suggest that emphasising the extent to which FLOSS project contributors are not paid misses the point, and that a better approach might relate to the extent to which their contributions are part of a formal role, whether within the FLOSS project itself, or as part of their employment.

8.3 FACTORS THAT INFLUENCE FLOSS PARTICIPANT SATISFACTION

This section discusses the findings of the quantitative survey, beginning with the measurement scales, and concluding with the research model and hypotheses.

8.3.1 *The measures*

The results discussed in Section 7.3 on page 154 showed that most scales performed as expected in terms of reliability (measured using Cronbach's alpha) and validity (measured using factor analysis). Specifically, perceived developer communication quality, participant experience, extent of participation, and perceived system complexity all had Cronbach's alpha values greater than the recommended threshold value of 0.70, and had their maximum loadings on separate factors in the factor analysis. In addition, the two new measurement scales, perceived product openness and perceived process openness all met the recommended minimum threshold of .70 for Cronbach's alpha, and mapped clearly onto separate factors in the factor analysis. However, the items associ-

ated with three constructs had less satisfactory results. In particular, the measures of training and task complexity were problematic, and were dropped from the model because the data were not reliable. Secondly, the measure of knowledge and skills appeared to represent two underlying concepts: LIM knowledge and skills, and technical knowledge and skills.

The research also developed a new scale to measure the outcome variable participant satisfaction, which was above the recommended threshold value of .70 for Cronbach's alpha.

Training scale

The items used for the training scale were first developed by Nelson and Cheney (1987), and used by Guimaraes, Staples, and McKeen (1983), who found that it had a Cronbach's alpha of .83. However, in the current study it did not exhibit the same psychometric properties, having an alpha value of .52, and low values for Corrected item–Total correlation. This suggested that the scale is not measuring a clearly-defined underlying construct. This conclusion is supported by the descriptive results for the scale shown in Table 30 on page 145. These show different patterns for the various types of training specified in the individual items, with the two items that relate to formal training having mainly low scores, and the two items relating to self-study having mainly high scores. This suggests that the scale is not measuring overall training effectiveness, but how similar the various training options are to each other in terms of effectiveness.

The respondents in the Guimaraes, Staples, and McKeen (1983) study were all primary users of software developed in-house, and could reasonably be expected to have come from similar backgrounds, and had similar training options for the software. The context for the current study, FLOSS packages used in library and information management, is considerably more diverse, which is illustrated by the range of countries represented in the survey responses shown in Table 15 on page 107, and also the range of projects that are represented, discussed in Section 7.2.1 on page 138. The training options that were available to survey respondents are likely to be equally diverse, leading to the observed lack of reliability in the training measurement scale. Further research is needed to identify more suitable items to measure training effectiveness in a FLOSS context.

Task complexity scale

The task complexity construct was measured using items from a scale developed by Rizzo, House, and Lirtzman (1993, first published in 1970) that related to role conflict and ambiguity. Though Rosenkrantz, Luthans, and Hennessey (1983) showed that it could be used in a range of fields with acceptable psychometric properties, the data from the cur-

rent study did not demonstrate these properties. In particular, the item relating to knowing what other people expected of them loaded most strongly on the process openness scale, and the remaining two items had communalities below .50, suggesting that they were measuring different underlying concepts.

Task complexity and its relationship to satisfaction with software/systems have previously been measured in contexts that differ from the current study in two ways: first, they involved software developed in-house, rather than FLOSS, and they involved people who used the information provided by the system, rather than using the system to provide information to others. It is possible that this difference in context affected the way respondents interpreted the scale, and it suggests that further research into task complexity, as perceived by FLOSS LIM project participants, is needed.

Knowledge and skills scale

The knowledge and skills scale was based on a scale first developed by Torkzadeh and Lee (2003). Though the items that related to this construct had a Cronbach's alpha value of .83, which is above the minimum recommended value of .70, the factor analysis showed that the items mapped as separate factors, which were named 'Technical knowledge and skills' and 'LIM knowledge and skills'. The most likely explanation for the difference in the properties of the scale in this research is that the respondents to this survey were more diverse than those for previous research that has used the scale. In earlier research, the respondents have been primarily software users, and it is reasonable to expect that their responses to all of the questions showed similar patterns. However, the items that formed what this research termed 'Technical knowledge and skills' had the lowest factor loadings in Torkzadeh and Lee's original study, close to .70, while the other items had higher loadings (over .80), which suggests that they may represent a slightly different, but related underlying construct. In this research, some of the respondents were primarily developers, whose knowledge of LIM applications (covered by the final four items) may have been very different to their knowledge of hardware, operating systems, and programming languages, which may have led to the clearer separation in the factor analysis.

8.3.2 *The revised research model*

The results of the regression analysis suggest that the model presented in Figure 12 on page 191 is a better representation of the relationships between the individual predictor variables and the outcome variable than the original model presented in Figure 2 on page 59. Overall the revised model explained 44% of the variation in satisfaction, which is a moderate effect size according to Cohen (1992). The beta values

show that perceived system complexity is the most important factor influencing satisfaction, with a value of −.28, while perceived product openness and extent of participation are the least important with values of .20. However, beta values for all the constructs fall within a relatively small range (absolute values between .20 and .28), and all are significant at $p \leqslant .01$. This shows that they all play a role in determining a FLOSS project participant's overall satisfaction.

One implication is that people involved in FLOSS project governance and management activities may need to consider all of the factors if they wish to raise participant satisfaction with the project, since focusing on just one or two may not resolve underlying issues with other aspects. In particular, if perceived system complexity is high, improving process or product openness may not result in increased levels of overall satisfaction.

8.3.3 *The hypotheses*

The original research model incorporated nine hypotheses, only four of which were supported by the data. This section discusses the results for each of the hypotheses in turn.

H1 The higher the perceived participant influence, the higher the participant satisfaction.

The results showed that participant influence did not have a significant effect on satisfaction, and the construct was therefore removed from the model. Influence related to the extent to which the respondent's ideas and suggestions were implemented in the software, or how these affected decisions relevant to the project. Previous research examining the relationship between influence and satisfaction has had mixed results. McKeen, Guimaraes, and Wetherbe (1994) found a statistically significant, positive correlation between the two, while Guimaraes, Staples, and McKeen (2003) found no statistically significant relationship. The results from this research are closer to those of the later study. The reasons for this are unclear. One possibility is that the consensus approach to decision-making found in many FLOSS projects means that participants do not expect to be able to influence key decisions. This could mean that they consider any influence they have had to be unimportant when they assess their level of satisfaction with the project. In addition, the way this research measured influence was simple, involving two items, and using the maximum value in the regression analysis. Further research into the nature of influence in FLOSS communities might lead to the development of a more robust set of items to measure influence, which would allow the relationship to be retested.

H2 The higher the perceived quality of developer communication, the higher the participant satisfaction.

The results showed that perceived developer communication quality had a direct influence on satisfaction ($\beta = .22$, significant at $p \leqslant .01$). Perceived developer communication quality related to the extent to which the developers were perceived as responding quickly and appropriately, as well as how easy they were to understand. Since members of a FLOSS project community are distributed geographically, typically relying on asynchronous communication to make decisions, it makes intuitive sense that a higher perceived quality of communication with developers will be associated with a higher level of satisfaction.

One survey respondent provided a comment that indicates how important this aspect of a FLOSS project was to him, saying:

> "For open source library software to really thrive, there must be more consideration given by the community to the newbie experience. If a potential user has taken the time to research the software, downloaded and installed it, and then has questions that they post about problems, bugs, or configuration issues, the community should have some level of commitment to respond. The lack of response is a clear signal to me that the software is simply not ready for my organization, or me personally, to invest in."

This comment makes the point that first impressions count by emphasising that good communication is particularly important for new users of the software. Another survey respondent had a more positive view of developer communication (for a different project), saying

> "[T]he community is VERY helpful, and answers or pointers are a few minutes away via IRC or via online forums and/or documentation. I am glad to be part of such a wonderful community."

This comment was made in a reference to difficulties understanding the software's workflow, and it illustrates the positive effect that good communication can have on members of the FLOSS project's community.

Previous research examining the relationship between perceived developer communication quality and satisfaction has had mixed results. McKeen, Guimaraes, and Wetherbe (1994) found a low, but statistically significant, positive correlation between the two, while Guimaraes, Staples, and McKeen (2003) found no statistically significant relationship. The results from this research are closer to those of the earlier study. McKeen, Guimaraes, and Wetherbe did

not report beta values, meaning that it is not possible to compare the current results directly with theirs. However, the results of this study suggest that if members of a FLOSS project's community perceive that the quality of their communication with the project's developers is high, their level of satisfaction will be higher.

This has practical consequences for members of a project's community. It suggests that they need to monitor communication channels regularly, and ensure that questions, particularly those from newcomers, are answered clearly and promptly. Though this may be difficult to monitor if questions are answered privately, rather than copied to the email discussion list or forum, a policy that all responses are copied to the list or forum, unless a private reply is requested, is common practice in FLOSS projects. Information about the importance of this practice should be part of the project's core information about support options.

H3 There is a positive relationship between participant skills and knowledge and participant satisfaction.

This hypothesis was not supported, and therefore the skills and knowledge construct was removed from the model. Previous research into the relationship between participant skills and knowledge and satisfaction has had varying results, with Blili, Raymond, and Rivard (1998) and Jang (2009) finding evidence to support a statistically significant, positive relationship, while Palvia and Palvia (1999) found no significant relationship. The results of this research are most similar to those of Palvia and Palvia, suggesting that in a FLOSS context, other factors have more influence on satisfaction than participant skills and knowledge. The FLOSS projects that survey respondents identified were all collaborative, involving contributions from many people, rather than being primarily the work of a single individual. The results suggest that the respondents may have placed more value on the results of this collaboration than on their own contributions.

H4 There is a positive relationship between participant training and participant satisfaction.

This hypothesis could not be tested because the results of the Cronbach's alpha and factor analysis showed that this construct lacked reliability. This may have been because training in the use of FLOSS projects is seldom available, meaning that participants had no meaningful way of assessing its impact. The participant training construct was therefore removed from the revised model. In order to test this relationship, further research will be needed to identify a suitable measure for training in the context of a FLOSS project. Previous research into the relationship between training and satisfaction has had varying results, with Sharma

and Yetton (2009) finding that correlations ranged from −.05 to .51. This relatively wide range suggests that there may have been issues with the measurement scales used in the various studies, and which supports the need for more research into this.

H5 There is a positive relationship between participant experience and participant satisfaction.

This hypothesis was not supported, and therefore the participant experience construct was removed from the revised model. Like perceived influence and perceived quality of developer communication, previous research into the relationship between experience and satisfaction has had mixed results. Guimaraes, Staples, and McKeen (2004) found that there was a moderate–substantial .33 correlation between experience and satisfaction, significant at $p \leqslant .01$, while Lawrence and Low (1993) found no significant relationship. The results from this research are closest to those of Lawrence and Low. The interpretation of this finding in a FLOSS context is difficult to establish, though the results shown in Table 32 on page 149 indicate that most respondents saw themselves as having the same or more experience than other people involved in the project. It is possible that testing the original model with a larger and more diverse sample would have different results, which suggests that further research, with a different sample, would be useful.

H6 There is a positive relationship between the extent of participation and participant satisfaction.

The results showed that extent of participation had a direct influence on satisfaction ($\beta = .20$, significant at $p \leqslant .01$). This is consistent with the results of previous research, which consistently showed a positive relationship between participation and satisfaction, though the strength of the relationship varied. The moderate relationship found in the current research is slightly below the integrated effect size of .33 He and King found in their meta-analysis of 48 studies that examined the relationship between participation and satisfaction (2008). Since the studies He and King included all involved conventional, in-house software development projects, the similarity of the results suggests that participation plays a similar role in a FLOSS project, and that participants will be more satisfied if they are given opportunities to contribute to the project. This suggests that people involved in FLOSS project governance, management, and community support need to identify options for different types of participation and make them known to members of their community. Lists of suggested contributions can include activities in any of the six categories: code, community, governance, management, outreach, and sponsorship. This should allow

potential contributors to identify an option that is a good match to their skills, which would then lead to wider participation. For example, new users could document their experiences, and the questions they had about the software. This would contribute to the information available to potential users, and over time these contributions have the potential to build into an effective user guide.

H7A The greater the perceived system complexity, the greater the relationship between extent of participation and participant satisfaction.

This hypothesis was not supported. The results showed that system complexity was an independent predictor variable for satisfaction, rather than a moderator variable. The model was therefore revised to show a direct relationship between perceived system complexity and satisfaction. This differs from McKeen, Guimaraes, and Wetherbe's 1994 finding that system complexity moderated the relationship between participation and satisfaction. In the current research, system complexity had a direct influence on satisfaction ($\beta = -.28$, significant at $p \leqslant .01$). The difference in behaviour of this variable may be attributable to the difference in context between the current study and McKeen, Guimaraes, and Wetherbe's. Their research concerned conventional in-house software development, where users receive support from other members of the project team. The same type of support is not necessarily available for participants in a FLOSS project, which may change the relationship between perceived system complexity and satisfaction. Survey respondents who saw the software they were involved with as complex had lower satisfaction. This issue is an inherent part of the FLOSS context, illustrated by the following quote from a survey respondent:

> "[T]he very factor that is most appealing, the ability to customize, seems to render the products' [sic] documentation ambiguous at best and unhelpful at worst."

It shows that there is a tension between the freedom to change the software to meet local needs, and the added complexity this can add to the environment. Because of this, it is perhaps not surprising that perceived complexity had the highest beta value in the multiple regression, which shows that it has the most impact on satisfaction in comparison to the other factors.

H7B The greater the perceived task complexity, the greater the relationship between extent of participation and participant satisfaction.

This hypothesis could not be tested, because the results of the Cronbach's alpha and factor analysis showed that this construct lacked reliability. The scale used for the construct was originally

developed for use in measuring task ambiguity, and may not have been appropriate in a FLOSS context. The task complexity scale was a modified version of the 6-item scale developed by Rizzo, House, and Lirtzman (1993), which had previously been found to have an alpha value of between .78 and .81. McKeen, Guimaraes, and Wetherbe used it successfully in their 1994 study of user satisfaction with in-house application software. However, in this research, survey respondents based their responses on a range of FLOSS packages used for different business purposes; some were using transaction-processing systems like Koha, others digital library/repository software like Greenstone and DSpace, and a third group based their answers on web content management systems or wiki engines. The specific tasks respondents carried out with these packages would have varied considerably, which may be one explanation for the lack of reliability in the data. Another explanation may be that the complexity of the tasks they carried out varied with their roles. In order to test this relationship, further research will be needed to identify an effective measure of task complexity in the context of a FLOSS project, and whether there is a relationship between tasks carried out and role, whether formal or informal.

H8A The higher the perceived process openness, the greater the relationship between extent of participation and participant satisfaction.

The results showed that perceived process openness was not a moderator for the relationship between extent of participation and satisfaction, and this relationship was therefore removed from the revised model. Since this relationship had not been tested previously, this hypothesis was one of two possibilities identified from the literature review. The results for the second possibility, H8B, are discussed below.

H8B There is a positive relationship between process openness and participant satisfaction.

The results showed that process openness had a direct influence on satisfaction ($\beta = .23$, significant at $p \leqslant .01$). This suggests that perceived process openness acts as a predictor variable for satisfaction, with the moderated regression results showing that process openness accounted for 25% of the variation in satisfaction. Process openness was defined as the extent to which participants felt that they were able to participate in project decision-making, and that their contributions were welcome and valued. This finding provides empirical evidence that a characteristic unique to a FLOSS context, process openness, has a direct and significant correlation with satisfaction. Its relationship with satisfaction suggests that people involved with FLOSS project governance and community

building need to make sure that their project is presented in a way that new participants perceive as welcoming, and that contributions are appreciated and visibly recognised. This suggests that activities that build and support the project's community may be as important as those that contribute to writing code, particularly as projects grow.

H9A The higher the perceived product openness, the greater the relationship between perceived participant influence and participant satisfaction.

The results showed that perceived product openness is not a moderator for the relationship between extent of participation and satisfaction, and this relationship was therefore removed from the revised model. Since this relationship had not been tested previously, this hypothesis was one of two possibilities identified from the literature review. The results for the second possibility, H9B, are discussed below.

H9B There is a positive relationship between product openness and participant satisfaction.

The results showed that product openness had a direct influence on satisfaction ($\beta = .20$, significant at $p \leqslant .01$), which suggests that perceived product openness acts as a predictor variable for satisfaction. Product openness was defined as the extent to which information about the project's future plans was available to members of its community. This finding provides empirical evidence that survey respondents' satisfaction was higher if they felt they were informed about the project's future development path. Its relationship with satisfaction suggests that people involved with FLOSS project government and management need to ensure that they make their plans for the project's future clear to members of their community.

8.3.4 *New moderating variables*

Section 7.5.4 on page 183 discussed three characteristics of a FLOSS project community members' participation that had the potential to affect their level of satisfaction, and the relative importance of the factors that affect this. These were organisational focus, extent of remuneration, and the amount of time they spent working on project activities. Carrying out multiple regression analysis on subgroups of the data identified differences in beta values for the predictor variables between groups.

Process openness was most important for respondents who had a non-local (i.e. community) focus, or were paid for 50% or less of their time working on the project, while perceived developer communication quality was most important for respondents who had a local focus, or were

paid for more than 50% of their time. This suggests that these two characteristics may be potential moderator variables for the relationships between the predictor variables and satisfaction.

The differences between groups based on the amount of time respondents spent working on the project were smaller, which suggests that this is unlikely to be a significant moderator variable.

8.3.5 *Other findings*

Although the research objectives did not include describing the characteristics of participants in LIM FLOSS projects, or identifying the number of available projects, this emerged as an additional finding. The results of the survey showed that respondents came from 33 different countries. Though these were predominately English-speaking, a number of other countries, both in Europe and the developing world, were also represented. This means that efforts to translate user interfaces to other languages are increasingly important to some projects. It may also raise the importance of high quality communication with members of a project's community, since at least some of these respondents are likely to have limited English language skills. For example, using casual or colloquial English on email discussion lists may confuse non-native English speakers. Community members may also need to respond carefully to unclear messages that come from non-native English speakers, who may be doing their best to communicate despite their lack of fluency in English.

The other aspect of the survey findings that deserves comment is the number of projects respondents identified. Respondents named a total of 37 individual FLOSS projects relating to library and information management, and the distribution followed a 'long-tail' pattern, with 23 projects named by just one respondent. Some of these projects were general-purpose information management software, such as wiki engines or web content management systems. By using this type of FLOSS package, members of the LIM profession benefit from a wider community and are able to do things they could not otherwise do. However, other software was specific to the LIM field, and some of the projects appeared to be similar in scope. While this shows that FLOSS projects are becoming popular in the LIM field, it also raises questions about the extent to which the field can sustain a growing number of projects. The list of projects could be used as a starting point for further research, since it could be used as a wider population to track the evolution of projects over time, which might identify critical success factors for a LIM FLOSS project's long-term survival.

8.4 SUMMARY

This chapter discussed the research findings and their significance. It showed that the user-centric FLOSS contribution model presented a broader perspective on the activities carried out by FLOSS project participants than the more common code-centric model, and that this has implications for the language researchers use. The results from the quantitative survey show that perceived complexity plays a different role in a FLOSS context that in a conventional software development project, and that characteristics of the project's community are more important as well. The global reach of many LIM FLOSS projects also has implications for the way project members communicate with each other. Some survey respondents benefited from the popularity of generic information management software projects, such as wikis and web content management systems, but the relatively high number of LIM-specific FLOSS projects raises questions about the ability of the field to sustain them.

9

CONCLUSION

9.1 INTRODUCTION

This chapter begins by summarising the research, including the research questions, the research design, and the findings, followed by a discussion of its contributions to research and to practice. It ends with a discussion of the limitations of the research, and outlines further research to extend the findings.

9.2 RESEARCH OVERVIEW

This section presents the research background, the conceptual model, the research design, and the key findings.

9.2.1 *Research background*

This research was prompted by a popular story about Richard Stallman's reason for starting the free software movement: his frustration with his inability to change the software used to run a new printer in MIT's Artificial Intelligence Laboratory in the early 1980s. The result of this frustration was codified in the Free Software Foundation's four software freedoms:

- the freedom to run the program, for any purpose (freedom 0);
- the freedom to study how the program works, and change it to make it do what you wish (freedom 1);
- the freedom to redistribute copies so you can help your neighbour (freedom 2); and
- the freedom to distribute copies of your modified versions to others (freedom 3).

The Open Source Initiative was established in the late 1990s, to give the free software movement more appeal to business. As a result, free/libre and open source software is now well-established as an alternative to closed, proprietary software.

Previous research had indicated that participation in software development was one of a number of factors that influenced user satisfaction, but this research had not been extended to FLOSS projects. This led to the main research question: *What factors influence participant satisfaction with a free/libre and open source application software project?*

9.2.2 *Research model*

A review of the research literature on participation in FLOSS projects showed that much of the focus had been on software developers and interaction with source code, and that little attention had been paid to community-oriented activities. It also showed that there was little standardisation in the terms used to identify project roles. This led to the first sub-question for this research: *What types of contributions do participants make to free/libre and open source software projects?*

A review of the information systems literature showed that a number of factors had been shown to have a direct influence on user satisfaction with software, though the extent of this influence varied. The individual factors were: participation in software development, perceived developer communication quality, participant skills and knowledge, perceived influence, experience, and training. In addition, system and task complexity had been shown to moderate the relationship between participation and satisfaction. A research model was developed based on these factors; it also incorporated two characteristics unique to a FLOSS context: perceived product openness and perceived process openness.

9.2.3 *Research design*

The research used a sequential, mixed methods approach. The first qualitative stage involved interviews with 24 participants who were involved with a range of roles in seven LIM FLOSS projects, complemented with ongoing document review of 10 LIM FLOSS projects. The purpose of this stage was to gain an understanding of the types of contributions they made to the projects, and to confirm that the concepts included in the research model were relevant to a FLOSS context.

The second stage of the research involved a quantitative, web-based survey. Invitations were sent to five email discussion lists, and subsequently forwarded to at least two other email discussion lists and promoted on several library blogs. A total of 183 usable responses was received, from people in 33 countries. The data from this were used to test the model.

9.2.4 *Research findings and model revision*

The findings from the first, qualitative stage of the research were used to develop a user-centric contribution model of a FLOSS project, showing seven types of activity, plus four attributes of participation that spanned all activity types. They were also used to review and confirm the preliminary research model, and develop measurement scales for the new constructs.

The analysis of the second, quantitative stage began with an assessment of the reliability of each measurement scale and a factor analysis of the items associate with the predictor variables, followed by multiple regression and moderated regression analysis. These findings showed that perceived system complexity had the largest effect on satisfaction ($\beta = -.28, p \leqslant .01$), while perceived process openness accounted for the most variance in satisfaction ($R^2 = .25, p \leqslant .01$). Overall, characteristics related to the project and its community were more influential on overall satisfaction than personal characteristics such as skills and knowledge, perceived influence, and experience. A two-word summary of the main research findings into factors that influence FLOSS participant satisfaction is 'Community matters'. Projects that were perceived as being open to participation with high quality communication resulted in higher levels of satisfaction than projects that were perceived as being more closed.

One finding that differs from previous research is the role of perceived system complexity. McKeen, Guimaraes, and Wetherbe (1994) found that system complexity moderated the relationship between participation and satisfaction. However, in the FLOSS context of this research system complexity acted as a predictor variable for satisfaction. It emerged as the most influential factor in comparison to the others that were tested in the model. Respondent comments made it clear that perceived system complexity had both benefits and drawbacks. One benefit of complexity was that the FLOSS package could be tailored to the specific individual or organisational needs, but the drawback of this was that it tended to decrease satisfaction.

The final outcome of this research was a revised research model, showing that five predictor variables had a statistically significant influence on satisfaction, accounting for 44% of the overall variance in satisfaction. This model is a parsimonious one, designed in include the smallest number of statistically significant constructs to explain the maximum amount of variance in the outcome variable.

9.3 RESEARCH CONTRIBUTIONS

The results of this research contribute to both theory and practice. This section first discusses its theoretical contribution, followed by its implications for participants in FLOSS projects.

9.3.1 *Theoretical contributions*

The main theoretical contribution of this research comes from its exploration of satisfaction in the context of FLOSS projects. Existing models of satisfaction, and its measurement, have been developed almost exclusively in the context of in-house software development, with a small

amount of research that looked at user-developed applications of proprietary commercial products, most often spreadsheets. The results make it clear that community aspects of participating in a FLOSS project are more significant than in more closed projects. Some of these, such as product and process openness, are not relevant to in-house projects, but are an additional and important part of a FLOSS project.

A second contribution of this research is the development of a user-centric FLOSS participation model, that incorporates more types of contributions than usually considered. This presents a broader perspective on what participation involves than the previous developer- or code-centric models. One potential use of this model is to provide a descriptive framework to compare projects in terms of relative amount of time their participants spend on on the different types of activities over time, which would give a greater understanding of the ways in which different projects evolve.

Another contribution of this research is the development of a scale to measure satisfaction with a FLOSS project. The items used for this had a Cronbach's alpha of .88, and the resulting distribution of values had a Komogorov–Smirnov significance of .44, indicating a normal distribution. This scale incorporated elements of existing satisfaction scales, plus others that Stage 1b interviewees identified as being important in a FLOSS context.

This research also demonstrated that a form of cluster sampling can be effective in gathering data from participants in a range of FLOSS projects. Previous FLOSS research has tended to use single projects as the source of their data, or data harvested from repositories. The respondents to this research represented 44 identifiable LIM FLOSS projects, which increases the generalisability of the findings.

Finally, this research showed that Nelson and Cheney's instrument used to measure training effectiveness did not work well in a FLOSS context, and raised questions about its overall reliability. It also found that Rizzo, House, and Lirtzman's role conflict and ambiguity scale (1993, first published in 1970), though previously used effectively as a surrogate for task complexity, did not work well in a FLOSS context. New measures will need to be developed for both of these constructs, and retested in a FLOSS context.

9.3.2 *Contributions to practice*

The results of this research will also be of interest to members of FLOSS project communities. Its most valuable contribution comes from its use of empirical evidence gathered from a range of projects to show which aspects of a project's culture have the most effect on participant satisfaction. Although most of the presentations at recent practitioner conferences, such as linux.conf.au and OSCON, had a technical focus,

the titles of some sessions show that there is growing awareness that community is important. Selected examples of such sessions from the 2010 linux.conf.au and OSCON conferences include:

- Build your own contributors, one part at a time;
- Lessons learned from a growing project;
- Making yourself popular: A guide to social success in (and for) the Linux community;
- Mentoring for fun and profit;
- Open source for newbies: Attracting and retaining talented people for your project; and
- The secrets of building and participating in open source communities.

The abstracts for these sessions make it clear that they are based on anecdotes and personal experience, which suggests that the current research can be used to provide evidence of where FLOSS projects should focus their efforts to improve overall participant satisfaction.

Some specific suggestions extrapolated from the hypotheses supported by the results of this research include:

- ensure that the project's 'About' page and documentation include information about what types of contributions are most needed, and how to contribute (based on H6);
- acknowledge and celebrate contributions, so that people who do contribute feel appreciated and motivated to continue (based on H8B);
- monitor questions in the project's email discussion list and/or forums, particularly those from newcomers, to ensure that they are answered (based on H2);
- provide information to the project's community about the project's future development, perhaps in the form of a 'road map' that lists the planned changes and enhancements (from H9B);
- ensure that documentation is up-to-date, and that aspects of the software that may be perceived as complex are explained clearly (based on H7C); and
- find out what barriers participants encounter when making a contribution to the project, and take steps to minimise or eliminate them (based on h6).

The results from this research will also be useful to practitioners in the field of library and information management, particularly those interested in becoming more involved in relevant FLOSS projects. In

particular, it gives them a framework to identify different ways in which they could participate in the projects, as well as guidance on what to look for when seeking a project open to participation from new users.

9.4 LIMITATIONS OF THE RESEARCH

All research has limitations. The most significant delimitation of this research is that it focused on a subset of FLOSS projects, namely applications used in library and information management, generally adopted at an organisational level. This delimitation means that the results are unlikely to be generalisable to the wider population of FLOSS projects. In particular, they may not be generalisable to FLOSS projects that are not intended for end-users, such as the Linux Kernel project, the Apache project, and other infrastructure projects. However, the results may apply to other types of FLOSS application projects which are intended to be used by end users to carry out their work or business tasks, particularly since some of the FLOSS applications named by survey respondents are already used outside the LIM field, such as the WordPress blogging software and MediaWiki.

The second significant limitation of the research is a self-selection bias due to its use of a general invitation to participate on email discussion lists, rather than inviting an identifiable sample with a known relationship to the wider population. This means that the respondents to the Stage 2c survey may over-represent people who are satisfied with the software, since people who are not at all satisfied may not have been subscribed to the project's email discussion list. Even if they were, they might not have been motivated to respond to the survey. This limitation also means that it is not possible to calculate a response rate for the survey, since an unknown number of people received an invitation to participate. However, the demographic characteristics of the respondents reported in Section 4.3 on page 105 show that they came from different countries, ranged in age and educational qualifications, and included both men and women. While this does suggest that there is no identifiable bias towards a specific type of respondent, one key difference between the samples for both stages of this research and those in previous research is the higher proportion of women (50% in Stage 1, and 32% in Stage 2). There may be several reasons for this difference. First, the researcher was identified as female, and it is possible that prospective interviewees and survey respondents were influenced by this, leading to a higher number of responses from women. All but one of the people who did not respond to the invitation to participate in an interview were male, which supports this suggestion. In addition, the library and information management profession is generally considered to be largely female, so it is reasonable to assume that a higher proportion of women received the invitation to complete the survey

than in previous FLOSS research. This suggests that the population for this research may be significantly different to that of previous FLOSS research, which also limits the generalisability of the results to other types of FLOSS projects.

A further source of bias in the results of Stage 1 is the high proportion (roughly half) of the interviewees who were involved with the same project. This could have introduced bias towards the attitudes and practices of that project. The researcher attempted to limit this bias by ensuring that the types of activities included in Figure 4 on page 125 were discussed by interviewees, or identified in the document review, from more than one project, to increase the generalisability of the model.

The nature of satisfaction means that many of the research measures involved perceptions, rather than objective measures. This means that the results reflect the subjective perspectives of individual survey respondents, and there may be inconsistencies in the way each interpreted the measurement scale. This limitation was addressed by using multiple measures for the constructs where possible.

Measurement issues are another limitation to the findings, since some constructs were dropped from the model before the regression testing. This was because their scales did not meet the minimum recommended thresholds for reliability, or because they failed to load cleanly on a single factor in the factor analysis.

A further limitation of the research was the use of multiple regression as the main data analysis technique. The results of multiple regression indicate that there is a relationship between the variables, but do not imply causation. Further research, using techniques such as structural equation modelling (SEM), would allow causal relationships to be examined.

This thesis did not take into account the life-cycle stage of the FLOSS projects nominated by survey respondents. This may have limited its ability to distinguish differences in satisfaction between respondents whose activities were primarily oriented to code, and those who were primarily users, since Subramanayn, Weisstein, and Krishnan (2010) found that the life-cycle stage in which developers and users participated affected their satisfaction.

Finally, the research was conducted in English, which may have meant that FLOSS participants who are not fluent in English were unlikely to respond to the survey. This means that the findings are likely to reflect the perspectives of the English-speaking participants.

9.5 FUTURE RESEARCH

All research also raises new questions, or identifies possibilities for further research. Because this research was among the first to examine

FLOSS project participant satisfaction, there are a number of aspects that can be expanded or clarified. They include:

- Extend the concepts of process and project openness, in order to refine and clarify the concepts, and develop more comprehensive measurement scales.
- Conduct a study of developer communication characteristics and how they are perceived by FLOSS project members, in order to develop guidelines for good practice.
- Examine the training options available for a range of FLOSS projects in different fields, in order to develop a reliable measurement scale, and carrying out a survey to validate it.
- Identify barriers to participation in FLOSS projects, and ways they can be minimised or resolved.

  This appears to be particularly significant for documentation, since a number of interviewees and survey respondents identified documentation as a weakness, but did not themselves take any action to improve its quality. This suggests that they felt the barriers to contributing documentation were higher than the benefits of doing so.
- Extend the research model by including attitude to FLOSS.

  This could also examine the impact of this attitude on initial selection and subsequent satisfaction with a FLOSS project.
- Identify factors which influence the extent to which FLOSS projects apply the concept of process openness to their tools and documentation.

  This could include examining the licenses for the bug tracker and communication channels, as well as the licenses used for the project's documentation.
- Examine participants' satisfaction with different types of projects, such as their size, activity level, and life-cycle stage, to see if there are any significant differences.

  This would build on Subramanayn, Weisstein, and Krishnan's (2010) research examining the relationship between satisfaction and participation in different stages of the software life-cycle.

Another option for further research would be to examine satisfaction with proprietary software projects, in order to determine which of the constructs used in this research apply in that context. Much of the previous research on user satisfaction with software was carried out in the 1980s and 1990s, when the structure of the software industry was very different to that of the 2000s.

9.6 SUMMARY

This chapter concludes this thesis by summarising the research topic and research design. It identifies the contributions to theory and to practice made by the findings, and discusses the limitations of the research. It concludes with some suggestions for further research to extend and clarify the research findings.

APPENDICES

A

STAGE 1 HUMAN ETHICS APPLICATION

Victoria
UNIVERSITY OF WELLINGTON
Te Whare Wānanga
o te Ūpoko o te Ika a Māui

INFORMATICS HUMAN ETHICS COMMITTEE

Application for Approval of Research Projects

Please email applications to your supervisor, who will then email it to an Informatics HEC member for a preliminary review.

Note: The Human Ethics Committee attempts to have all applications approved within 6 working days, but a longer period may be necessary if applications require substantial revision.

1 NATURE OF PROPOSED RESEARCH:

(a) Student Research (delete one)

(b) If Student Research Degree …Ph.D…. Course Code …INFO 690…

(c) Project Title: …Participant satisfaction with open source software: Stage 1

……

2 INVESTIGATORS:

(a) Principal Investigator

Name ……Brenda Chawner……

e-mail address …brenda.chawner@vuw.ac.nz……

School/Dept/Group …Information Management……

(b) Other Researchers	Name	Position
……..		…….
……..		…….

(c) Supervisor (in the case of student research projects)

…Gary Gorman/Sid Huff…… Professor….

3 DURATION OF RESEARCH

(a) Proposed starting date for data collection – After HEC approval has been granted.
(Note: that NO part of the research requiring ethical approval may commence prior to approval being given)

(b) Proposed date of completion of project as a whole …December 2007…

4 PROPOSED SOURCE/S OF FUNDING AND OTHER ETHICAL CONSIDERATIONS

(a) Sources of funding for the project
Please indicate any ethical issues or conflicts of interest that may arise because of sources of funding e.g. restrictions on publication of results

……none……...

……...

(b) Is any professional code of ethics to be followed **Y**
If yes, **name** AIS Code of Research Conduct……..……

(c) Is ethical approval required from any other body **N**
If yes, name and indicate when/if approval will be given

……...

5 DETAILS OF PROJECT

Briefly Outline:

(a) The objectives of the project

to identify factors that influence participant satisfaction with an open source software project……..……..…...

……...

(b) Method of data collection

face-to-face or email interviews ……..……...

……...

……...

……...

(c) The benefits and scientific value of the project

The results of this project will help developers of open source application software projects identify opportunities to increase user satisfaction, and it will also help users identify ways in which they might contribute to projects

(d) Characteristics of the participants

Developers and users of library and information management (LIM) open source application software projects.

(e) Method of recruitment

A purposive sample of developer and users will be drawn up from a range of sources, including articles and conference papers about LIM open source projects, and Web-based email archives for LIM open source project discussion lists. Prospective participants will be sent an email invitation (attached) inviting them to contribute to this stage of the project. The email invitation will be sent as plain text, to conform to th norms of the open source community. The contact information at the bottom of the message and the 'vuw.ac.nz' email address will provide my institutional affiliation.

(f) Payments that are to be made/expenses to be reimbursed to participants

None……...

(g) Other assistance (e.g. meals, transport) that is to be given to participants

None……...

(h) Any special hazards and/or inconvenience (including deception) that participants will encounter

None

(i) State whether consent is for: (Please indicate as many as it applies)

(i)	the collection of data	**Y**
(ii)	attribution of opinions or information	**N**
(iii)	release of data to others	**N**
(iv)	use for a conference report or a publication	**Y**
(v)	use for some particular purpose (specify)	**Y**

PhD thesis deposited in the VUW library or entered into an institutional repository…...

……...

Attach a copy of any questionnaire or interview schedule to the application

(j) How is informed consent to be obtained (see paragraphs 4.31(g), 5.2, 5.5 and 5.61 of the Guidelines)

(i) the research is strictly <u>anonymous</u>, an information sheet is supplied and informed consent is implied by voluntary participation in filling out a questionnaire for example (include a copy of the information sheet)

N

(ii) the research is <u>not anonymous</u> but is confidential and informed consent will be obtained through a signed consent form (include a copy of the consent

form and information sheet) **Y (face to face)**

(iii) the research is neither anonymous nor confidential and informed consent will be obtained through a signed consent form (include a copy of the consent form and information sheet) **N**

(iv) informed consent will be obtained by some other method (please specify and provide details) **Y (email)**

Email interview partipants will be advised that they are implying consent to participate by accepting the initial invitation and subsequently responding to email interview questions. If they do not wish to participate, they can ignore the email invitation.

With the exception of anonymous research as in (i), if it is proposed that written consent will not be obtained, please explain why

The email interviews will be carried out asynchronously using email as the main communication channel. Requiring the participants to complete and return a consent form via another channel (such as post or fax) will introduce a barrier to participation, and is expected to lower the response rate. To avoid this, participants will be advised that they are implying consent by accepting the initial invitation and by subsequently replying to the email interview questions.

(k) If the research will not be conducted on a strictly anonymous basis state how issues of confidentiality of participants are to be ensured if this is intended. (See paragraph 4.3.1(e) of the Guidelines). (e.g. who will listen to tapes, see questionnaires or have access to data). Please ensure that you distinguish clearly between anonymity and confidentiality. Indicate which of these are applicable.

(i) access to the research data will be restricted to the investigator **N**

(ii) access to the research data will be restricted to the investigator and their supervisor (student research) **Y**

(iii) all opinions and data will be reported in aggregated form in such a way that individual persons or organisations are not identifiable **Y**

(iv) Other (please specify)

……...

……...

……...

(l) Procedure for the storage of, access to and disposal of data, both during and at the conclusion of the research. (see section 7 of the guidelines). Indicate which are applicable:

(i) all written material (questionnaires, interview notes, etc) will be kept in a locked file and access is restricted to the investigator **Y**

(ii) all electronic information will be kept in a password-protected file and access will be restricted to the investigator **Y**

(iii) all questionnaires, interview notes and similar materials will be destroyed:
(a) at the conclusion of the research **N**

or (b) __2__ years after the conclusion of the research **Y**

(iv) any audio or video recordings will be returned to participants and/or electronically wiped **Y**

(v) other procedures (please specify):

……...

……...

If data and material are not to be destroyed please indicate why and the procedures envisaged for ongoing storage and security

……...

……...

……...

(m)Feedback procedures (See section 8 of the Guidelines). You should indicate whether feedback will be provided to participants and in what form. If feedback will not be given, indicate the reasons why.

I will be posting ongoing, informal reports on my research progress on a blog, and participants will be advised of its location and invited to comment. When the research is complete, all participants will be sent a summary of the main research findings.

(n)Reporting and publication of results. Please indicate which of the following are appropriate. The proposed form of publications should be indicated on the information sheet and/or consent form.

(i) publication in academic or professional journals **Y**

(ii) dissemination at academic or professional conferences **Y**

(iii) deposit of the research paper or thesis in the University Library (student research) **Y**

(iv) a case study used for teaching purposes **N**

(v) other (please specify)

PhD thesis deposited in the VUW library or entered into an institutional repository……

……...

Signature of investigators as listed on page 1 **(including supervisors) and Chair of Informatics HEC.**

NB: <u>All investigators and the Chair of Informatics HEC must sign the form, then send it to Perumal Pillai for filing in the University's Research Office once the electronic application has been approved.</u>

…… Date……...

…… Date……...

…… Date……...

Supervisors:

…… Date……...

…… Date……...

Chair of Informatics HEC:

…… Date ……..

APPLICATIONS FOR HUMAN ETHICS APPROVAL

CHECKLIST

☐ Have you read the Human Ethics Committee Policy?

☐ Have you read the Faculty of Commerce and Administration's HEC Guide?

☐ Is ethical approval required for your project?

☐ Have you established whether informed consent needs to be obtained for your project?

☐ In the case of student projects, have you consulted your supervisor about any human ethics implications of your research?

☐ Have you included an information sheet for participants which explains the nature and purpose of your research, the proposed use of the material collected, who will have access to it, whether the data will be kept confidential to you, how anonymity or confidentiality is to be guaranteed?

☐ Have you included a written consent form?

☐ If not, have you explained on the application form why you do not need to get written consent?

Are you asking participants to give consent to:

- ☐ collect data from them
- ☐ attribute information to them
- ☐ release that information to others
- ☐ use the data for particular purposes

☐ Have you indicated clearly to participants on the information sheet and/or consent form how they will be able to get feedback on the research from you (e.g. they may tick a box on the consent form indicating that they would like to be sent a summary), and how the data will be stored or disposed of at the conclusion of the research?

☐ Have you included a copy of any questionnaire or interview checklist you propose using?

POINTERS TO AVOID HAVING APPLICATIONS RETURNED BEFORE HEC REVIEW

- The approval process is speeded up by not requiring the hard copy of your application form with the signatures on it at the initial review process. The complete application (HEC application form, info sheet, consent form, covering letter, questionnaire etc.) is to be emailed as an attachment in one file to your supervisor who will email it to an INFORMATICS HEC member for a preliminary review.
- Do not insert a date into item 3 a.
- Delete the "Y" or "N" option that is not required. **DO NOT** remove any other text from the application form.
- **BOLD** your answers if you wish but do not alter the font anywhere else in the

B

STAGE 1 FACE-TO-FACE INVITATION

Face-to-face interview invitation (sent via email)

Participant satisfaction with open source software: invitation

Dear [ ]

I am a Ph.D. student in the School of Information Management at Victoria University of Wellington, and I would like to invite you to be a participant in the first phase of my project. I am looking at factors that influence participant satisfaction with library and information management open source software projects, and I have approached you because of your association with the [insert project name] project.

This phase of the project will involve an interview of up to one hour, at a time that is convenient for you. Please indicate your willingness to be interviewed by replying to this email message. Your name will be confidential to me and my supervisors, and the results will be presented in aggregated form so that individual people or projects cannot be identified.

The results of this research will be used to develop a Web-based questionnaire for the second stage of my project, due to be carried out early in 2007. They will be discussed in my Ph.D. thesis, and may also be published in articles in professional or academic journals or presented at conferences.

If you accept this invitation, I will send you a more detailed information sheet, a consent form, and a list of interview questions before the interview takes place. If you would like to read about my research progress, you may check my blog, which is available at http://www.of2minds.net/reflections/. When my thesis is completed, I will send you a summary of the findings.

If you have any questions about this project, you may contact me by email at brenda.chawner@vuw.ac.nz. My supervisors are Professor Gary Gorman (gary.gorman@vuw.ac.nc) and Professor Sid Huff (sid.huff@vuw.ac.nz).

I look forward to hearing from you.

Regards,

Brenda Chawner

School of Information Management
Victoria University of Wellington
P O Box 600
Wellington NEW ZEALAND
(04) 463 5780 fax (04) 463 5446

C

STAGE 1 FACE-TO-FACE INFORMATION SHEET

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

SCHOOL OF INFORMATION MANAGEMENT

Participant satisfaction with open source software: Stage 1

Information Sheet

I am a Ph.D. student in the School of Information Management at Victoria University of Wellington, studying factors that influence participant satisfaction with library and information management open source software projects. The results of this project will help developers of open source application software projects identify opportunities to increase user satisfaction, and it will also help users identify ways in which they might contribute to projects.

This phase of the project will involve an interview, which may take up to one hour. The interview will be recorded electronically, and I may also take brief notes during the interview. I will prepare a transcript of the interview, and you will have an opportunity to check it before I start to analyse the data.

Your name will be confidential to me and my supervisors, and the results will be presented in aggregated form so that individual people or projects cannot be identified. Your responses will be stored securely in a password-protected file on my personal computer, and any hard copies will be stored in a locked filing cabinet. The data will be kept for a period of 2 years after my project is completed, after which it will be deleted, and any hard copy shredded.

The results of this stage of my research will be used to develop a Web-based questionnaire for the second stage of my project, due to be carried out early in 2007. They will be discussed in my Ph.D. thesis, which will be deposited in the Victoria University of Wellington Library or in an institutional repository. They may also be published in articles in professional or academic journals or presented at conferences. If you would like to read about my research progress, you may check my blog, which is available at http://www.of2minds.net/reflections/. When my thesis is completed, I will send you a summary of the findings.

Victoria University of Wellington requires ethical approval for all research involving human participants, and this project has been approved by the School of Informatics Human Ethics Committee.

You may withdraw from the project without explanation at any time until the start of the data analysis on 1 January 2007. If you withdraw from the project, any data you have provided will be destroyed.

If you have any questions about this project, you may contact me by email at brenda.chawner@vuw.ac.nz. My supervisors are Professor Gary Gorman (gary.gorman@vuw.ac.nc) and Professor Sid Huff (sid.huff@vuw.ac.nz).

P.O .Box 600, Wellington, New Zealand Telephone +64-4-463 5384 Facsimile +64-4-463 5446

D

STAGE 1 FACE-TO-FACE INFORMATION SHEET

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

SCHOOL OF INFORMATION MANAGEMENT

4

Participant satisfaction with open source software: Stage 1

Consent form

☐ I have been provided with adequate information relating to the nature and objectives of this research project, I have understood that information and have been given the opportunity to seek further clarification or explanations.

☐ I understand that I will have an opportunity to check the interview transcripts before the data analysis begins on 1 January 2007.

☐ I understand that I may withdraw from this study at any time before the start of data analysis on 1 January 2007 data without providing reasons.

☐ I understand that if I withdraw from the project, any data I have provided will be destroyed.

☐ I understand that any information or opinions I provide will be kept confidential and reported only in an aggregated/non-attributable form.

☐ I understand that when this research is completed the information obtained will be retained for up to 2 years and then destroyed.

☐ I understand that the results of this research will be used in a PhD thesis that will be deposited in the Victoria University of Wellington Library or in an institutional repository, and that it may also be presented at conferences or published in academic or professional journals.

☐ I understand that I will receive a summary of the research findings when the project is complete.

☐ I agree to having my interview recorded.

Name: _____ Signature: _____

Date: _____

P.O .Box 600, Wellington, New Zealand Telephone +64-4-463 5384 Facsimile +64-4-463 5446

E

STAGE 1 FACE-TO-FACE INTERVIEW GUIDE

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

SCHOOL OF INFORMATION MANAGEMENT

Participant satisfaction with open source software projects

Interview Guide

Interview questions

1. Participant background (getting to know you)

Could you tell me a bit about yourself and your background? For example, what are your educational qualifications? What is your current position, and what types of positions have you held in the past? How would you describe your technical skills? Other skills?

2. Project background (getting to know the project)

If you are involved in more than one project, include up to three, discussing each project separately.

Could you tell me a bit about the software and what it does? What are its technical requirements? How did you get involved with the project? How do other people get involved with the project?

3. Participant contributions and roles (getting to know your relationship with the project)

I am interested in learning about the different ways participants contribute to F/OSS projects, not just contributing to the code, but also in other ways, such as answering questions, writing documentation, providing resources, or promoting the project.

How would you describe your main role in the project?

Could you tell me the sorts of contributions you have made to this project?

What are some of the contributions other people have made to this project?

How would you describe their roles in the project?

4. Participant satisfaction

Overall, how satisfied are you with different aspects of this project, such as its functionality, ease of use, documentation, reliability, or interaction between participants? If you could change anything about this project to increase your satisfaction, what would it be?

5. Other

Is there anything else you think I should be asking about?

P.O .Box 600, Wellington, New Zealand Telephone +64-4-463 5384 Facsimile +64-4-463 5446

F

STAGE 1 EMAIL INTERVIEW INVITATION

Email invitation/information sheet

Participant satisfaction with open source software: invitation

Dear [ ]

I am a Ph.D. student in the School of Information Management at Victoria University of Wellington, and I would like to invite you to be a participant in the first phase of my project. I am looking at factors that influence participant satisfaction with library and information management open source software projects, and I have approached you because of your association with the [insert project name] project. The results of this project will help developers of open source application software projects identify opportunities to increase user satisfaction, and it will also help users identify ways in which they might contribute to projects.

This phase of the project will involve an email interview. If you agree to participate, I will send you three sets of questions, spread out over a period of 2-3 weeks. The questions will be in the body of the email message, and you will be able to respond to them by replying to the message. I may have a few follow up questions for you, depending on your responses. I expect it to take you no more than 30 minutes to reply to each set of questions.

Your name will be confidential to me and my supervisors, and no individuals or projects will be identified in the final report or any publications resulting from this research. Your responses will be stored securely in a password-protected file on my personal computer, and any hard copies will be stored in a locked filing cabinet. The data will be kept for a period of 2 years after my project is completed, after which it will be deleted, and any hard copy shredded.

The results of this stage of my research will be used to develop a Web-based questionnaire for the second stage of my project, due to be carried out early in 2007. They will be discussed in my Ph.D. thesis, which will be deposited in the Victoria University of Wellington Library or in an institutional repository. They may also be published in articles in professional or academic journals or presented at conferences If you would like to read about my research progress, you may check my blog, which is available at http://www.of2minds.net/reflections/. When my thesis is completed, I will send you a summary of the findings.

Victoria University of Wellington requires ethical approval for all research involving human participants, and this project has been approved by the School of Informatics Human Ethics Committee.

You are implying consent to participate in this research project by accepting this invitation, and by responding to the subsequent sets of interview questions. You may withdraw from the project without explanation at any time until the start of the data analysis on 1 January 2007. If you withdraw from the project, any data you have provided will be destroyed.

If you have any questions about this project, you may contact me by email at brenda.chawner@vuw.ac.nz. My supervisors are Professor Gary Gorman (gary.gorman@vuw.ac.nc) and Professor Sid Huff (sid.huff@vuw.ac.nz).

I look forward to hearing from you.

Regards,

Brenda Chawner

School of Information Management
Victoria University of Wellington
P O Box 600
Wellington NEW ZEALAND
(04) 463 5780 fax (04) 463 5446

G

STAGE 1 EMAIL INTERVIEW INFORMATION SHEET

Participant satisfaction with open source software: Stage 1

Information Sheet

I am a Ph.D. student in the School of Information Management at Victoria University of Wellington, studying factors that influence participant satisfaction with library and information management open source software projects. The results of this project will help developers of open source application software projects identify opportunities to increase user satisfaction, and it will also help users identify ways in which they might contribute to projects.

This phase of the project will involve an email interview. If you agree to participate, I will send you three sets of questions, spread out over a period of 2-3 weeks. The questions will be in the body of the email message, and you will be able to respond to them by replying to the message. I may have a few follow up questions for you, depending on your responses. I expect it to take you no more than 30 minutes to reply to each set of questions.

Your name will be confidential to me and my supervisors, and the results will be presented in aggregated form so that individual people or projects cannot be identified. Your responses will be stored securely in a password-protected file on my personal computer, and any hard copies will be stored in a locked filing cabinet. The data will be kept for a period of 2 years after my project is completed, after which it will be deleted, and any hard copy shredded.

The results of this stage of my research will be used to develop a Web-based questionnaire for the second stage of my project, due to be carried out early in 2007. They will be discussed in my Ph.D. thesis, which will be deposited in the Victoria University of Wellington Library or in an institutional repository. They may also be published in articles in professional or academic journals or presented at conferences. If you would like to read about my research progress, you may check my blog, which is available at http://www.of2minds.net/reflections/. When my thesis is completed, I will send you a summary of the findings.

Victoria University of Wellington requires ethical approval for all research involving human participants, and this project has been approved by the School of Informatics Human Ethics Committee. You may withdraw from the project without explanation at any time until the start of the data analysis on 1 January 2007. If you withdraw from the project, any data you have provided will be destroyed.

If you have any questions about this project, you may contact me by email at **brenda.chawner@vuw.ac.nz**. My supervisors are Professor Gary Gorman (**gary.gorman@vuw.ac.nz**) and Professor Sid Huff (**sid.huff@vuw.ac.nz**).

Brenda Chawner

School of Information Management

Victoria University of Wellington
P O Box 600
Wellington NEW ZEALAND
(04) 463 5780 fax (04) 463 5446

This is written in **HTML 4.01 Strict** and **CSS**.
Best viewed in **any browser**.

Created: 20 October 2006
Last Modified: 20 October 2006

H

STAGE 1 EMAIL INTERVIEW GUIDE

Email interview questions

Participant satisfaction with open source software: questions part [1 | 2 | 3]

Thank you for agreeing to participate in my research project. Here is the [first | second | final] set of interview questions. Please respond by typing your answers below each question, and then 'replying' to me.

The information sheet for this research is available at

http://www.vuw.ac.nz/staff/brenda_chawner/infosheet.html.

1. Participant background (getting to know you)

Could you tell me a bit about yourself and your background? For example, what are your educational qualifications? What is your current position, and what types of positions have you held in the past? How would you describe your technical skills? Other skills?

2. Project background (getting to know the project)

If you are involved in more than one project, include up to three, discussing each project separately.

Could you tell me a bit about the software and what it does? What are its technical requirements? How did you get involved with the project? How do other people get involved with the project?

3. Participant contributions and roles (getting to know your relationship with the project)

I am interested in learning about the different ways participants contribute to F/OSS projects, not just contributing to the code, but also in other ways, such as answering questions, writing documentation, providing resources, or promoting the project.

How would you describe your main role in the project?

Could you tell me the sorts of contributions you have made to this project?

What are some of the contributions other people have made to this project?

How would you describe their roles in the project?

4. Participant satisfaction

Overall, how satisfied are you with different aspects of this project, such as its functionality, ease of use, documentation, reliability, or interaction between participants? If you could change anything about this project to increase your satisfaction, what would it be?

5. Other

Is there anything else you think I should be asking about?

I

STAGE 2 HUMAN ETHICS APPLICATION

Victoria
UNIVERSITY OF WELLINGTON
Te Whare Wānanga
o te Ūpoko o te Ika a Māui

SIM HUMAN ETHICS COMMITTEE
Application for Approval of Research Projects

Please email applications to your supervisor, who will then email it to a SIM HEC member for a preliminary review.

Note: The Human Ethics Committee attempts to have all applications approved within 6 working days, but a longer period may be necessary if applications require substantial revision.

1 NATURE OF PROPOSED RESEARCH:

(a) Student Research (delete one)

(b) If Student Research Degree Ph.D. Course Code INFO 690

(c) Project Title: Participant satisfacton with open source software: Stage 2

2 INVESTIGATORS:

(a) Principal Investigator

Name Brenda Chawner

e-mail address brenda.chawner@vuw.ac.nz

School/Dept/Group Information Management

(b) Other Researchers Name Position

…….. ……

…….. ……

(c) Supervisor (in the case of student research projects)

Gary Gorman Professsor

Sid Huff Professor

3 DURATION OF RESEARCH

(a) Proposed starting date for data collection – After HEC approval has been granted.
(Note: that NO part of the research requiring ethical approval may commence prior to approval being given)

(b) Proposed date of completion of project as a whole 30 June 2009

4 PROPOSED SOURCE/S OF FUNDING AND OTHER ETHICAL CONSIDERATIONS

(a) Sources of funding for the project
Please indicate any ethical issues or conflicts of interest that may arise because of sources of funding e.g. restrictions on publication of results

None……...

……...

(b) Is any professional code of ethics to be followed **Y**
If yes, **name** AIS Code of Research Conduct

(c) Is ethical approval required from any other body **N**
If yes, name and indicate when/if approval will be given

……...

5 DETAILS OF PROJECT

Briefly Outline:

(a) The objectives of the project

to identify factors that influence participant satisfaction with an open source software project

……...

……...

(b) Method of data collection

anonymous online survey……...

……...

……...

……...

(c) The benefits and scientific value of the project

The results of this project will help developers of open source application software projects identify opportunities to increase user satisfaction, and it will also help users identify ways in which they might contribute to projects

(d) Characteristics of the participants

Developers and users of library and information management open source application software projects.

(e) Method of recruitment

An email invitation to participate in the survey will be sent to a number of project and library technology email discussion lists. A copy of the message is included as part of this HEC application.

(f) Payments that are to be made/expenses to be reimbursed to participants

None……...

(g) Other assistance (e.g. meals, transport) that is to be given to participants

None……...

(h) Any special hazards and/or inconvenience (including deception) that participants will encounter

None……...

……...

(i) State whether consent is for: (Please indicate as many as it applies)

(i)	the collection of data	**Y**
(ii)	attribution of opinions or information	**N**
(iii)	release of data to others	**N**
(iv)	use for a conference report or a publication	**Y**
(v)	use for some particular purpose (specify)	**Y**

Ph.D. thesis deposited in the VUW library or entered into the institutional repository……...

……...

Attach a copy of any questionnaire or interview schedule to the application

(j) How is informed consent to be obtained (see paragraphs 4.31(g), 5.2, 5.5 and 5.61 of the Guidelines)

(i) the research is strictly anonymous, an information sheet is supplied and informed consent is implied by voluntary participation in filling out a questionnaire for example (include a copy of the information sheet)

Y

(ii) the research is not anonymous but is confidential and informed consent will be obtained through a signed consent form (include a copy of the consent form and information sheet) **N**

(iii) the research is neither anonymous nor confidential and informed consent will be obtained through a signed consent form (include a copy of the consent form and information sheet) **N**

(iv) informed consent will be obtained by some other method (please specify and provide details) **N**

……...

……...

With the exception of anonymous research as in (i), if it is proposed that written consent will not be obtained, please explain why

……...

……...

……...

(k) If the research will not be conducted on a strictly anonymous basis state how issues of confidentiality of participants are to be ensured if this is intended. (See paragraph 4.3.1(e) of the Guidelines). (e.g. who will listen to tapes, see questionnaires or have access to data). Please ensure that you distinguish clearly between anonymity and confidentiality. Indicate which of these are applicable.

(i) access to the research data will be restricted to the investigator **N**

(ii) access to the research data will be restricted to the investigator and their supervisor (student research) **Y**

(iii) all opinions and data will be reported in aggregated form in such a way that individual persons or organisations are not identifiable **Y**

(iv) Other (please specify)

……...

……...

……...

(l) Procedure for the storage of, access to and disposal of data, both during and at the conclusion of the research. (see section 7 of the guidelines). Indicate which are applicable:

(i) all written material (questionnaires, interview notes, etc) will be kept in a locked file and access is restricted to the investigator **Y**

(ii) all electronic information will be kept in a password-protected file and access will be restricted to the investigator **Y**
(iii) all questionnaires, interview notes and similar materials will be destroyed:
(a) at the conclusion of the research **N**
or (b) 2 years after the conclusion of the research **Y**
(iv) any audio or video recordings will be returned to participants and/or electronically wiped **n/a**
(v) other procedures (please specify):

……...

……...

If data and material are not to be destroyed please indicate why and the procedures envisaged for ongoing storage and security

……...

……...

……...

(m)Feedback procedures (See section 8 of the Guidelines). You should indicate whether feedback will be provided to participants and in what form. If feedback will not be given, indicate the reasons why.

A summary of the results will be sent to a range of project and library technology email discussion lists.

(n)Reporting and publication of results. Please indicate which of the following are appropriate. The proposed form of publications should be indicated on the information sheet and/or consent form.

(i) publication in academic or professional journals **Y**
(ii) dissemination at academic or professional conferences **Y**
(iii) deposit of the research paper or thesis in the University Library (student research) **Y**
(iv) a case study used for teaching purposes **N**
(v) other (please specify)

An electronic copy will be available on the Web via the VUW institutional repository (http://researcharchive.vuw.ac.nz/).

……...

Signature of investigators as listed on page 1 **(including supervisors) and Chair of SIM HEC.**

NB: <u>All investigators and the Chair of SIM HEC must sign the form, then send it to the SIM HEC administrator for filing once the electronic application has been approved.</u>

…… Date……...

…… Date……...

…… Date……...

Supervisors:

…… Date……...

…… Date……...

Chair of SIM HEC:

……. Date ……..

APPLICATIONS FOR HUMAN ETHICS APPROVAL

CHECKLIST

- ☐ Have you read the Human Ethics Committee Policy?
- ☐ Have you read the Faculty of Commerce and Administration's HEC Guide?
- ☐ Is ethical approval required for your project?
- ☐ Have you established whether informed consent needs to be obtained for your project?
- ☐ In the case of student projects, have you consulted your supervisor about any human ethics implications of your research?
- ☐ Have you included an information sheet for participants which explains the nature and purpose of your research, the proposed use of the material collected, who will have access to it, whether the data will be kept confidential to you, how anonymity or confidentiality is to be guaranteed?
- ☐ Have you included a written consent form?
- ☐ If not, have you explained on the application form why you do not need to get written consent?

Are you asking participants to give consent to:

- ☐ collect data from them
- ☐ attribute information to them
- ☐ release that information to others
- ☐ use the data for particular purposes

- ☐ Have you indicated clearly to participants on the information sheet and/or consent form how they will be able to get feedback on the research from you (e.g. they may tick a box on the consent form indicating that they would like to be sent a summary), and how the data will be stored or disposed of at the conclusion of the research?
- ☐ Have you included a copy of any questionnaire or interview checklist you propose using?

POINTERS TO AVOID HAVING APPLICATIONS RETURNED BEFORE HEC REVIEW

- The approval process is speeded up by not requiring the hard copy of your application form with the signatures on it at the initial review process. The complete application (HEC application form, info sheet, consent form, covering letter, questionnaire etc.) is to be emailed as an attachment in one file to your supervisor who will email it to an SIM HEC member for a preliminary review.
- Do not insert a date into item 3 a.
- Delete the "Y" or "N" option that is not required. **DO NOT** remove any other text from the application form.
- **BOLD** your answers if you wish but do not alter the font anywhere else in the form.

J

STAGE 2 SURVEY INVITATION

Wed, 15 Oct 2008 11:48 AM

Subject: Satisfaction with free/open source software: survey invitation
Date: Wednesday, 15 October 2008 11:47 AM
From: Brenda Chawner <Brenda.Chawner@vuw.ac.nz>
Conversation: Satisfaction with free/open source software: survey invitation

I am a Ph.D. candidate in the School of Information Management at Victoria University of Wellington in New Zealand, and my research topic is an investigation of factors that influence participant satisfaction with library or information management free/open source software projects. Some library-related examples are DSpace, EPrints, Koha, Evergreen, Greenstone, and MyLibrary. More general information management software includes web content management software such as Drupal, wiki software such as MediaWiki or PmWiki, or blogging software such as WordPress.

If you use or are involved with a relevant project, I would like to invite you to complete an online survey. I am especially interested in hearing from people in a range of roles, for example user, developer, release manager, or system administrator. I am keen to have responses from people who have had either positive or negative experiences with free/open source software, to ensure that I get a perspective on factors that diminish satisfaction, as well as those that contribute toward it.

The results of this project will help developers of free/open source application software projects identify opportunities to increase user satisfaction, and it will also help users identify ways in which they might contribute to projects.

I expect the survey to take between 15 and 20 minutes of your time. You may receive several copies of this message, since I am sending it to a number of project and library technology email discussion lists; however, I ask that you only complete the survey once. If you have colleagues who you think would be interested in completing it, please forward this invitation to them.

The survey is available at:

http://surveys.sim.vuw.ac.nz/survey.aspx?surveyid=205

It will be available until Friday, 14 November, 2008.

I will post a summary of the results to relevant project and library technology email discussion lists, once the thesis is finished in mid-2009. A copy of the thesis will be deposited in the Victoria University of Wellington Institutional Repository (http://researcharchive.vuw.ac.nz/).

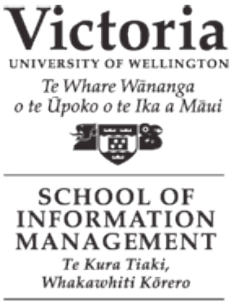
If you have any questions about this survey, please contact me at brenda.chawner@vuw.ac.nz or phone +64 4 463 5780. My supervisors are Professor Gary Gorman, email gary.gorman@vuw.ac.nz and Professor Sid Huff, email sid.huff@vuw.ac.nz.

Regards,
--
Brenda Chawner
School of Information Management
Victoria University of Wellington
P O Box 600
Wellington NEW ZEALAND
(04) 463 5780 fax (04) 463 5446 brenda.chawner@vuw.ac.nz

K

STAGE 2 WEB-BASED SURVEY

Free/Open Source Participant Satisfaction Survey



This survey has been designed to gather data to investigate factors influencing participants' satisfaction with free/open source software. It consists of 30 questions, asking about your background, your attitudes to free/open source software in general, and your experience and satisfaction with one library or information management free/open source software project. It should take you between 15 and 20 minutes to complete the survey, which has received ethical approval from the School of Information Management Human Ethics Committee.

Please answer all questions that apply to your current situation. If a question does not apply to you, please leave it blank, or choose 'n/a'.

In this research, free/open source software is defined as software that is issued under a license that guarantees access to source code, and ensures that users have:

a. the freedom to run the software, for any purpose;
b. the freedom to read the source code to see how it works, and to modify it to suit local conditions;
c. the freedom to redistribute copies; and
d. the freedom to improve it, and redistribute the improved version.

No data that identifies you individually is being collected, and the results of the research will not be related to specific projects, apart from indicating how many people responded for each project. The data will be used for a PhD thesis, which will be deposited in the university library and made available online in its institutional repository. The results may also be presented at conferences, or published as articles in academic or professional journals. Only aggregate data will be presented in the thesis and any publications resulting from this research, and any quotes taken from comments will not be attributed.

By completing and submitting the survey, you are implying consent to participate. I will post a summary of the results to relevant project and library technology email discussion lists, once the thesis is finished in mid-2009.

The software used for this survey was issued under a free/open source license, in keeping with the topic of the research. The data you provide will be stored securely in password-protected files for up to 2 years, and then it will be destroyed.

If you have any questions about this survey, please contact me at brenda.chawner@vuw.ac.nz or phone +64 4 463 5780. My supervisors are Professor Gary Gorman, email gary.gorman@vuw.ac.nz and Professor Sid Huff, email sid.huff@vuw.ac.nz.

Thank you for your time.

Brenda Chawner
Ph.D. Candidate
School of Information Management
Victoria University of Wellington
New Zealand

Section 1: Background and education

1. How old are you?

◯ 20 or younger ◯ 21-25 ◯ 26-30 ◯ 31-35 ◯ 36-40 ◯ 41-45 ◯ 46-50 ◯ 51-55 ◯ 56-60 ◯ 61 or older

2. What is your gender?

◯ Female ◯ Male

3. What is your highest educational qualification?

◯ None
◯ Secondary or high school graduate
◯ Postsecondary certificate or diploma
◯ Undergraduate degree
◯ Postgraduate certificate or diploma
◯ Master's degree
◯ PhD

4. What country do you live in?

[Select Country]

5. How long have you been using a computer, either at work or at home?

◯ Less than 5 years
◯ 5-10 years
◯ 11-15 years
◯ 16-20 years
◯ 21-25 years
◯ 26-30 years
◯ More than 30 years

6. Please rate your level of knowledge and skills in the following areas:

	minimal	some	moderate	much	extensive
Knowledge and use of hardware	◯	◯	◯	◯	◯
Knowledge and use of operating systems	◯	◯	◯	◯	◯
Knowledge and use of one or more programming languages	◯	◯	◯	◯	◯
Knowledge and use of library or information management application software	◯	◯	◯	◯	◯
Ability to provide system designers with information required to develop library or information management application software	◯	◯	◯	◯	◯
Ability to define library or information management application software requirements	◯	◯	◯	◯	◯
Ability to assess library or information management application software features	◯	◯	◯	◯	◯

Section 2: Attitude to free/open source software

7. Before starting this survey, how familiar were you with the idea of free/open source software?

- not at all familiar
- slightly familiar
- somewhat familiar
- quite familiar
- very familiar

8. To what extent do you use free/open source software:

	not at all	very little	sometimes	often	as much as possible	don't know
On computers provided by your employer	○	○	○	○	○	○
On computers you own	○	○	○	○	○	○

9. What operating system do you use on computers provided by your employer?

10. When choosing a new application software package for use at work, to what extent do you give preference to free/open source alternatives?

- It makes no difference to me
- Other people make the decision for me
- I prefer to use proprietary software with vendor support
- I will consider a free/open source option, and choose it if it meets my needs best
- I give preference to free/open source software whenever possible
- I only use free/open source software

Other (please specify)

11. What operating system do you use on computers you own?

12. When choosing a new application software package for use on computers you own, to what extent do you give preference to free/open source alternatives?

- It makes no difference to me
- Other people make the decision for me
- I prefer to use proprietary software with vendor support
- I will consider a free/open source option, and choose it if it meets my needs best

- ◯ I give preference to free/open source software whenever possible
- ◯ I only use free/open source software

Other (please specify)

Section 3: Experience and satisfaction with one library or information management free/open source project

In this section of this survey, please answer based on *one* library or information management free/open source source software project you use or are involved with in some other way. Some library-related examples are DSpace, EPrints, Koha, Evergreen, Greenstone, and MyLibrary. More general information management software includes web content management software such as Drupal, wiki software such as MediaWiki or PmWiki, or blogging software such as WordPress.

If you are involved with more than one project, please choose the one that you have used or contributed to most recently.

13. What is the name of the project? There are too many to list here, so please specify the one on which you will base your subsequent responses.

14. How long have you been using or contributing to this project?

- ◯ Less than 6 months
- ◯ Between 6 months and one year
- ◯ 1 to 2 years
- ◯ 2 to 4 years
- ◯ 4 to 6 years
- ◯ 6 to 8 years
- ◯ More than 8 years

15. How would you describe your current role in this project? Examples of roles include user, developer, maintainer, trainer, release manager, etc. If you have more than one role, please choose the one that takes up most of your time.

16. What other roles have you held in this project, if any?

17. This survey is concerned with two aspects of a free/open source software project: roles that relate to a specific implementation used in one or more institutions, and roles

that relate to the wider project/developer community or the version of the software available for anyone to download. You may be involved in one or both of these aspects. Please indicate:

	None	less than 5 hours	5-10 hours	11-20 hours	21-30 hours	more than 30 hours
In the last 6 months, how many hours per week have you spent in a role relating to a specific implementation, on average?	○	○	○	○	○	○
In the last 6 months, how many hours per week have you spent in a role relating to the wider project/developer community or the version of the software available for anyone to download, on average?	○	○	○	○	○	○

18. What proportion of your time working on this project, either locally or on the wider project, has been part of your paid employment?

- ○ None
- ○ Less than 20%
- ○ Between 20% and 50%
- ○ Between 50% and 80%
- ○ Between 80% and 100%

19. Which of the following activities have you carried out with this software/project?

	Please tick all that apply
Installed the software	☐
Upgraded the software to a more recent release	☐
Studied the source code to see how it works	☐
Used the software	☐
Distributed the software to others	☐
Joined the project's email discussion list/forum	☐
Asked a question on the project's email discussion list/forum	☐
Answered a question on the project's email discussion list/forum	☐
Promoted the project by talking about it to others, for example at a conference	☐
Promoted the project by writing about it for publication	☐
Provided resources to support the project, such as hosting an email discussion list, forum, or wiki	☐
Organised an event relating to the project, such as a meeting or conference	☐
Written documentation to help others use the software	☐
Customised the software to meet local needs, either yourself, or by having a developer do so	☐
Reported a bug to the system developers	☐
Requested an enhancement from the system developers	☐
Contributed local changes back to the project	☐

Fixed one or more bugs	☐
Evaluated existing software functionality	☐
Written software to add new features	☐

20. Have you used or contributed to this project in any other ways? Please specify.

21. Which of the following best describes how any training you have received affects your use of the software:

	n/a	not at all	very little	somewhat	considerably	extensively
Training provided by outside organisations	○	○	○	○	○	○
In-house training	○	○	○	○	○	○
Self-study using tutorials or online help	○	○	○	○	○	○
Self-study using manuals or other documents	○	○	○	○	○	○

22. Please briefly describe any other training you have received that affects your use of the software:

23. Please indicate your general level of satisfaction with the following characteristics of the software:

	n/a	not at all satisfied	slightly satisfied	somewhat satisfied	quite satisfied	completely satisfied
reliability (i.e doesn't freeze, crash, or lose data)	○	○	○	○	○	○
functionality	○	○	○	○	○	○
free from bugs	○	○	○	○	○	○
easy to use	○	○	○	○	○	○
easy to learn	○	○	○	○	○	○
documentation	○	○	○	○	○	○
easy to install	○	○	○	○	○	○
easy to configure to meet local needs	○	○	○	○	○	○
release frequency	○	○	○	○	○	○

easy to add new features	○	○	○	○	○	○
helpfulness of community	○	○	○	○	○	○
security and access control	○	○	○	○	○	○

24. Please rate your experience in the following categories, relative to your perception of other people involved in the project:

	Considerably less than most	Slightly less than most	About the same as most	Slightly more than most	Significantly more than most
Experience using this type of software	○	○	○	○	○
Experience using this particular software package	○	○	○	○	○
Experience using computers in general	○	○	○	○	○
Experience as a member of a software development project	○	○	○	○	○

25. Please indicate your agreement with each of the following statements about the project's developers:

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	n/a
The project's developers are sensitive to others' needs	○	○	○	○	○	○
The project's developers typically get right to the point when communicating with others	○	○	○	○	○	○
The project's developers pay attention to what other people say	○	○	○	○	○	○
The project's developers deal effectively with others	○	○	○	○	○	○
The project's developers are easy to understand	○	○	○	○	○	○
The project's developers generally say the right thing at the right time	○	○	○	○	○	○
The project's developers are easy to communicate with	○	○	○	○	○	○
The project's developers respond to messages quickly	○	○	○	○	○	○
The project's developers express ideas clearly	○	○	○	○	○	○

26. Please indicate your agreement with the following statements about the project's culture:

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
I feel encouraged to contribute to this project	○	○	○	○	○
Anyone is encouraged to contribute to this project	○	○	○	○	○
Only a few people are allowed to contribute to this project	○	○	○	○	○
I find other people's contributions to this project valuable.	○	○	○	○	○

Other people find my contributions to this project valuable.	○	○	○	○	○
Information about the future development plans for this project is easy to find	○	○	○	○	○
The future development plans for this project are clear	○	○	○	○	○
The project has infrequent, formal releases of new versions of the software	○	○	○	○	○
The project has frequent releases of incremental versions with bug fixes and small enhancements	○	○	○	○	○

27. How much influence have you had on the software features/functionality, in your institution's local version?

- ○ None
- ○ Very little
- ○ A moderate influence
- ○ Much influence
- ○ Very much influence

28. How much influence have you had on the software features/functionality, in the version that is available for downloading by others?

- ○ None
- ○ Very little
- ○ A moderate influence
- ○ Much influence
- ○ Very much influence

29. Please indicate your agreement with each of the following statements about the software:

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
In comparison with other software I work with, this software has complex requirements	○	○	○	○	○
This software has a complex design	○	○	○	○	○
When working with this software, I have clear, planned goals and objectives for the tasks I am carrying out	○	○	○	○	○
When working with this software, I know what I am responsible for	○	○	○	○	○
When working with this software, I know exactly what other people expect of me	○	○	○	○	○

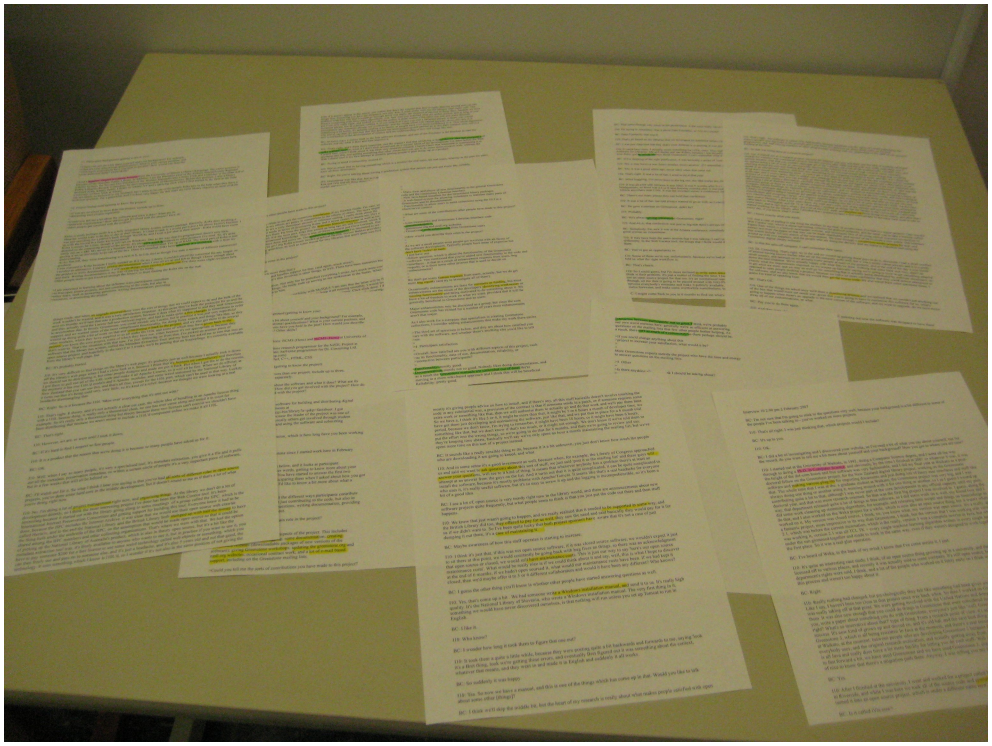
30. Are there any other comments you would like to make about your use of this software package, your involvement in the project, or reasons for your satisfaction or dissatisfaction? For example, how does it compare to other projects you are involved with?

Submit form

Page 1 / 1

L

HIGHLIGHTED TRANSCRIPTS



M

STAGE 1 DATA ANALYSIS: ACTIVITIES

Interviewee	Role	Activity	Category	Focus
1	Project manager	website redevelopment	outreach	
		commit code	code	
		host wiki	sponsor	
		ask questions	use	
		answer questions	community	
		organise conference	community	
		write documentation	community	
		customise code	code	
		article in journal	outreach	
		conference presentation	outreach	
		set up user group	community	
2	User	set up parameters	local use	internal
3, 4, 5	User	instigate project	local use	internal
		coordinate updates	outreach	internal
		localisation	coding	internal
		talk at conference	outreach	
6	Developer	commit code	code	
		write code	code	
		apply for grants	sponsor	
		issue press releases	outreach	
		conference presentation	outreach	
		media monitoring	governance	
7	Project manager	localisation	code	internal
		commit code	code	
8	Project sponsor	answer questions	community	
		write documentation	community	
		demonstrate system	outreach	
		training	community	
9	Core developer/project s	rewrite code	coding	
		write documentation	community	
		hire developers	manage	
		testing	coding	
		write manuals	community	
		set up mailing lists	sponsor	
		fix bugs	code	
		establish licensing policy	governance	
10	User	training	community	internal
		identifying changes	code	internal
11	User	write documentation	community	
		answer questions	community	
12	User			
13	Developer	fix bugs	code	
		write code	code	
		testing	code	
		write documentation	community	
		package release	code	
14	Developer	write code	code	
		write documentation	community	
		answer questions	community	
		organising staff tasks	manage	
		leading workshop	community	
		organise translations	community	
15	Release manager	write code	code	
		manage release	manage	
		plan future directions	governance	
16	Project manager	implement software	code	
		demonstrate system	outreach	
17	User	write documentation	community	
		provide financial resource	sponsorship	

STAGE 1 DATA ANALYSIS: ROLES

Role	Type	Category
Code integrator	Project	Code
Contributor	Project	Code
Developer	Project	Code
Developer	Project	Code
Documenter	Project	Code
Installation coordinator	Project	Code
Interface designer	Project	Code
Senior developer	Project	Code
Tester	Internal	Code
Windows packager	Project	Code
Coordinator	Project	Management
Documentation manager	Project	Management
IT project officer	Internal	Management
Operations manager	Internal	Management
Project leader	Internal	Management
Project manager	Project	Management
QA manager	Project	Management
Release manager	Project	Management
Team leader	Internal	Management
Training coordinator	Internal	Management
Translation manager	Project	Management
Project initiator	Project	Sponsor
Promoters	Project	Sponsor
Sponsors	Project	Sponsor
Sponsors	Project	Sponsor
Supporter	Project	Sponsor
Active user	Internal	User
End user	Internal	User
Team member	Internal	User
Team member	Internal	User
Technology analyst and coach	Internal	User
Website services assistant	Internal	User

O

STAGE 1 DATA ANALYSIS: ACTIVITY CATEGORIES

Activity	Category
fix bugs	code
fix bugs	code
identifying changes	code
implement software	code
implement software	code
package release	code
testing	code
write code	code
write code	code
write code	code
write code	code
commit code	code
commit code	code
commit code	code
customise code	code
fix bugs	code
localisation	code
localisation	code
rewrite code	code
testing	code
write code	code
answer questions	community
answer questions	community
answer questions	community
answer questions	community
answer questions	community
answer questions	community
leading workshop	community
organise conference	community
organise translations	community
run workshops	community
set up user group	community
training	community
training	community
training	community
translate documentation	community
write documentation	community
write documentation	community
write documentation	community
write documentation	community
write documentation	community
write documentation	community
write documentation	community
write documentation	community
write documentation	community
write manuals	community
establish licensing policy	governance
media monitoring	governance
plan future directions	governance
instigate project	local use
set up parameters	local use
hire developers	manage
invite other developers	manage
manage release	manage
organising staff tasks	manage
article in journal	outreach
conference presentation	outreach
conference presentation	outreach
coordinate updates	code

P

STAGE 1 DATA ANALYSIS: SATISFACTION CAUSES

Satisfaction

Satisfaction	Type
disappointment with documentation	Feature
strong community	Community
people answering questions	Community
flexibility of project	Feature
project openness	Openness
people talk together on IRC regularly	Community
meeting other developers face to face at a conference	Community
communication problems cause dissatisfaction	Community
value for money	Cost
dissatisfaction with documentation (a bit light)	Feature
reliability “it's been going the whole time”	Feature
good support from local vendor	Community
sharing customisations with other users	Participation
good functionality	Feature
easy to use	Ease
ability to customise for local context	Customisation
easy to add-on to proprietary system	Participation
global community; met people from all over the world	Community
opportunity to travel	Other
getting feedback on code	Other
fun challenging proprietary vendors	Attitude
dissatisfaction with loss of momentum	Community
seeing people get involved (answering questions)	Community
documentation is lacking	Feature
reliability	Feature
ability to control timing of upgrades	Control
ability to make local improvements	Customisation
growth of global community	Community
choice of support options	Community
winning prizes!	Other
seeing translated versions	Community
seeing other people pick up the software and use it	Community
having other people make the software better	Community
ability to make changes quickly	Customisation
dissatisfaction with implementation issues	Other
easy to use	Ease
lack of training	Training
communication problems with IT support staff	Community
easy to use	Ease
internal communication problems	Community
lack of reliability “things going wrong”	Feature
ease of use	Ease
documentation good, but can be out of date	Feature
lack of marketing material	Community
communication with developers could be better, especiall	Community
some functionality is limited	Feature
functionality good	Feature
documentation limited	Feature
limited interaction between participants	Community
software is reliable/stable	Feature
documentation good, but can be out of date	Feature
translated manuals out of date	Feature
wants the user community to contribute more	Community

Table 82: Survey question characteristics

QUESTION	DESCRIPTION	MEASUREMENT SCALE	VARIABLE TYPE	ANALYSIS TECHNIQUE
1	Age group	5 year age bands, starting from '20 or younger', '21-25', through '61 or older'	Ordinal	Frequency/ per cent
2	Gender	choice of 'Female' or 'Male'	Dichotomous	Frequency/ per cent
3	Education	Closed scale, starting from 'None' (1), 'Secondary or high school graduate' (2), through 'PhD' (7)	Ordinal	Frequency/ per cent
4	Country of residence	Drop down list	Nominal	Frequency/ per cent
5	Experience with computers	5-year bands, starting from 'Less than 5 years' (1)through 'More than 30 years' (7)	Ordinal	Frequency/ per cent
6	Perceived knowledge and skills with computers	Closed scale, ranging from 'minimal' (1) through 'extensive' (5) for 7 aspects of technology	Ordinal	Frequency/ per cent
7	Perceived familiarity with FLOSS	Closed scale, ranging from 'not at all familiar' (1) through 'very familiar' (5)	Ordinal	Frequency/ per cent

QUESTION	DESCRIPTION	MEASUREMENT SCALE	VARIABLE TYPE	ANALYSIS TECHNIQUE
8a	Use of FLOSS on work computers	Closed scale, ranging from 'not at all' (1) through 'as much as possible' (5). Also included a 'don't know' (6) option.	Ordinal	Frequency/ per cent
8b	Use of FLOSS on own computers	Closed scale, ranging from 'not at all' (1) to 'as much as possible' (5). Also included a 'don't know' (6) option.	Ordinal	Frequency/ per cent
9	Operating system(s) used on work computers	Free text	Nominal	Content analysis/ coding
10	Preference for using FLOSS on work computers	Closed scale, ranging from 'it makes no difference' to 'I only use free/open source software; also included an 'Other' option	Nominal	Frequency/ per cent
11	Operating system(s) used on own computers	Free text	Nominal	Content analysis/ coding
12	Preference for using FLOSS on own computers	Closed scale, ranging from 'it makes no difference' to 'I only use free/open source software; also included an 'Other' option	Nominal	Frequency/ per cent
13	FLOSS project name	Free text	Nominal	Frequency/ per cent

QUESTION	DESCRIPTION	MEASUREMENT SCALE	VARIABLE TYPE	ANALYSIS TECHNIQUE
14	Length of involvement	Closed scale, ranging from 'Less than 6 months' (1) to 'More than 8 years' (7)	Ordinal	Frequency/per cent
15	Current role	Free text	Nominal	Content analysis/coding
16	Other roles	Free text	Nominal	Content analysis/coding
17a	Average hours/week using in-house	Closed scale, ranging from 'None' (1) to 'more than 30 hours' (6)	Ordinal	Frequency/per cent
17b	Average hours/week contributing	Closed scale, ranging from 'None' (1) to 'more than 30 hours' (6)	Ordinal	Frequency/per cent
18	Paid proportion	Closed scale, ranging from 'None' (1) to 'Between 80% and 100%' (5)	Ordinal	Frequency/per cent
19a-t	Project activities	Tick boxes for 20 different activities	Dichotomous	Frequency/per cent
20	Other activities/-contributions	Free text		Content analysis/coding
21a-d	Training effectiveness	Closed scale, ranging from 'n/a' (0), 'not at all' (1), through 'extensively' (5)	Ordinal	Frequency/per cent
22	Other training	Free text		Content analysis/coding

QUESTION	DESCRIPTION	MEASUREMENT SCALE	VARIABLE TYPE	ANALYSIS TECHNIQUE
23a-l	Satisfaction	Closed scale for 12 aspects of the project, ranging from 'n/a' (0), 'not at all satisfied' (1), through 'completely satisfied' (5)	Ordinal	Frequency/per cent
24a-d	Perceived experience	Closed scale, ranging from 'Considerably less than most' (1) to 'Significantly more than most' (5)	Ordinal	Frequency/per cent
25a-j	Developer communication	Closed scale, ranging from 'Strongly disagree' (1) to 'Strongly agree' (5), plus 'n/a' (0)	Ordinal	Frequency/per cent
26a-i	Project culture	Closed scale, ranging from 'Strongly disagree' (1) to 'Strongly agree' (5); Question 26c was reverse coded to be consistent with the other measures of project culture	Ordinal	Frequency/ per cent
27	Perceived influence (in-house)	Closed scale, ranging from 'None' (1) to 'Very much influence' (5)	Ordinal	Frequency/ per cent
28	Perceived influence (shared project)	Closed scale, ranging from 'None' (1) to 'Very much influence' (5)	Ordinal	Frequency/ per cent
29a-e	Perceived system and task complexity	Closed scale, ranging from 'Strongly disagree' (1) to 'Strongly agree' (5)	Ordinal	Frequency/ per cent

QUESTION	DESCRIPTION	MEASUREMENT SCALE	VARIABLE TYPE	ANALYSIS TECHNIQUE
30	Other comments	Free text		Content analysis/ coding

BIBLIOGRAPHY

Abbott, E. A. (1953). *Flatland: A romance of many dimensions* (6th ed.). New York: Dover.

Abdinnour-Helm, S. F., Chaparro, B. S., & Farmer, S. M. (2005). Using the End-User Computer Satisfaction (EUCS) to measure satisfaction with a web site. *Decision Sciences*, *36*(2), 341-364.

Agarwal, R., & Prasad, J. (1997). The role of innovation characteristics and perceived voluntariness in the acceptance of information technologies. *Decision Sciences*, *28*(3), 557-582.

Aksulu, A., & Wade, M. (2010). A comprehensive review and synthesis of open source research. *Journal of the Association for Information Systems*, *11*(11/12), 576-656.

Aladwani, A. M. (2002). Organizational actions, computer attitudes, and end-user satisfaction in public organizations: An empirical study. *Journal of End User Computing*, *14*(1), 42-49.

Aladwani, A. M. (2003). A deeper look at the attitude-behavior consistency assumption in information systems satisfaction research. *Journal of Computer Information Systems*, *44*(1), 57-63.

Albrect, A. J., & Gaffney, J. E. (1983). Software function, source lines of code, and development effort prediction: A software science validation. *IEEE Transactions on Software Engineering*, *SE-9*(6), 639-648.

Al-Gahtani, S. S., & King, M. (1999). Attitudes, satisfaction and usage: Factors contributing to each in the acceptance of information technology. *Behaviour & Information Technology*, *18*(4), 277-297.

Allen, D. R. (2004). *Customer satisfaction research management: A comprehensive guide to integrating customer loyalty and satisfaction metrics*. Milwaukee, WI: ASQ Quality Press.

Allen, M. W., Armstrong, D. J., Reid, M. F., & Riemenschneider, C. K. (2008). Factors impacting the perceived organizational support of IT employees. *Information & Management*, *45*(8), 556-563.

Alreck, P. L., & Settle, R. B. (2004). *The survey research handbook* (3rd ed.). Boston: McGraw-Hill Irwin.

Amoako-Gyampah, K., & White, K. B. (1993). User achievement and user satisfaction. *Information & Management*, *25*(1), 1-10.

Applegate, R. (1997). Models of satisfaction. In A. Kent, H. Lancour, W. Nasri, & J. Daily (Eds.), *Encyclopedia of library and information science* (p. 199-227). New York: Marcel Dekker.

Association for Computing Machinery. (1998). *ACM software copyright and license agreement.* Retrieved 5/9/2010, from `http://www.acm.org/publications/policies/softwarecrnotice`

Au, N., Ngai, E., & Cheng, T. (2008). Extending the understanding of end user information systems satisfaction formation: An equitable needs fulfillment approach. *MIS Quarterly*, *32*(1), 43-66.

Au, N., Ngai, E. W., & Cheng, T. E. (2002). A critical review of end-user information system satisfaction research and a new research framework. *Omega*, *30*(6), 451-478.

Bailey, J. E., & Pearson, S. W. (1983). Development of a tool for measuring and analyzing computer user satisfaction. *Management Science*, *29*(5), 530-545.

Bargas-Avila, J., Lötscher, J., Orsini, S., & Opwis, K. (2009). Intranet satisfaction questionnaire: Development and validation of a questionnaire to measure user satisfaction with the Intranet. *Computers in Human Behavior*, *25*(6), 1241-1250.

Barki, H., & Hartwick, J. (1989). Rethinking the concept of user involvement. *MIS Quarterly*, *13*(1), 53-63.

Barki, H., & Hartwick, J. (1994). Measuring user participation, user involvement, and user attitude. *MIS Quarterly*, *18*(1), 59-82.

Barki, H., & Huff, S. L. (1985). Change, attitude to change, and decision support system success. *Information & Management*, *9*(5), 261-268.

Barki, H., & Huff, S. L. (1990). Implementing decision support systems: correlates of user satisfaction and system usage. *INFOR*, *28*(2), 89-101.

Baronas, A.-M. K., & Louis, M. R. (1988). Restoring the sense of control during implementation: How user involvement leads to system acceptance. *MIS Quarterly*, *12*(1), 111-125.

Baroudi, J. J., Olson, M. H., & Ives, B. (1986). An empirical study of the impact of user involvement on system usage and information satisfaction. *Communications of the ACM*, *29*(3), 232-238.

Baroudi, J. J., & Orlikowski, W. J. (1988). A short-form measure of user information satisfaction: A psychometric evaluation and notes on use. *Journal of Management Information Systems*, *4*(4), 44-59.

Basset, T. (2005). Coordination and social structures in an open source project: VideoLAN. In S. Koch (Ed.), *Free/open source software development* (p. 125-151). Hershey, PA: Idea Group.

Baudoin, P., & Branschofsky, M. (2004). Implementing an institutional repository: The DSpace experience at MIT. *Science & Technology Libraries*, *24*(1/2), 31–45.

Bentley, J. (1986). *Programming pearls*. Reading, MA: Addison-Wesley.

Bentley, J. (1989). *More programming pearls: Confessions of a coder*. Reading, MA: Addison-Wesley.

Biship, R. (2007). *The philosophy of the social sciences: An introduction*. London: Continuum.

Blackshaw, P. (2008). *Satisfied customers tell three friends, angry customers tell 3,000: Running a business in today's consumer-driven world*. New York: Doubleday.

Blili, S., Raymond, L., & Rivard, S. (1998). Impact of task uncertainty, end-user involvement, and competence on the success of end-user computing. *Information & Management*, *33*(3), 137-153.

Bloor, M. (1997). Techniques of validation in qualitative research: A critical commentary. In G. Miller & R. Dingwall (Eds.), *Context & method in qualitative research* (p. 37-50). London: Sage.

Boldyreff, C., Nutter, D., & Rank, S. (2004). Communication and conflict issues on collaborative software research projects. In J. Feller, B. Fitzgerald, S. Hissam, & K. R. Lakhani (Eds.), *Collaboration, conflict and control: Proceedings of the 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland* (p. 14-17).

Bonacccorsi, A., & Rossi, C. (2004). Contributing to OS projects: A comparison between individual and firms. In J. Feller, B. Fitzgerald, S. Hissam, & K. R. Lakhani (Eds.), *Collaboration, conflict and control: Proceedings of the 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland* (p. 18-22).

Bonacccorsi, A., & Rossi, C. (2006). Comparing motivations of individual programmers and firms to take part in the open source movements: From community to business. *Knowledge, Technology & Policy*, *18*(4), 40-64.

Braccini, A. M., Silvestri, C., & Za, S. (2010). Information systems: People, organizations, institutions, and technologies. In A. D'Atri & D. Sacchá (Eds.), (p. 549-556). Berlin: Physica-Verlag.

Bringhurst, R. (2002). *The elements of typographic style*. Point Roberts, WA: Hartley & Marks.

Brown, P. (2010). *Free software is a matter of liberty, not price.* Retrieved 5/9/2010, from `http://www.fsf.org/about/`

Bryman, A. (2008). *Social research methods* (3rd ed.). Oxford: Oxford University Press.

Bryman, A., & Bell, E. (2007). *Business research methods* (2nd ed.). Oxford: Oxford University Press.

Bryman, A., & Cramer, D. (2004). *Quantitative data analysis with SPSS 12 and 13*. London: Routledge.

Capiluppi, A., Lago, P., & Morisio, M. (2003). Evidences in the evolution of OS projects through changelog analysis. In J. Feller, B. Fitzgerald, S. Hissam, & K. R. Lakhani (Eds.), *Taking stock of the bazaar: Proceedings of the 3rd workshop on open source software engineering* (p. 23-28).

Carte, T. A., & Russell, C. J. (2003). In pursuit of moderation: Nine common errors and their solutions. *MIS Quarterly*, *27*(3), 479-501.

Casaló, L. V., Cisneros, J., Flavián, & Guinaliu, M. (2009). Determinants of success in open source software networks. *Industrial Management & Data Systems*, *109*(4), 532-549.

Cavaye, A. L. (1995). User participation in system development revisited. *Information & Management*, *28*(5), 311-323.

Chen, L.-D., Soliman, K. S., Mao, E., & Frolick, M. N. (2000). Measuring user satisfaction with data warehouses: An exploratory study. *Information & Management*, *37*(1), 103-110.

Chen, Q., Rodgers, S., & He, Y. (2008). A critical review of the e-satisfaction literature. *American Behavioral Scientist*, *52*(1), 38-59.

Cheung, C. M., & Lee, M. K. (2005). *The asymmetric effect of website attribute performance on satisfaction: An empirical study.* Paper read at 38th Hawaii International Conference on System Sciences.

Cheung, C. M., & Lee, M. K. (2008). The structure of Web-based information systems satisfaction: testing of competing models. *Journal of the American Society for Information Science and Technology*, *59*(10), 1617-1630.

Chin, P. O., & Cooke, D. (2004). *Satisfaction and coordination in virtual communities.* Paper read at 10th Americas Conference on Information Systems, 6-8 August, at New York, NY.

Cho, N., & Park, S. (2001). Development of electronic commerce user-computer satisfaction (EUSCI) for Internet shopping. *Industrial Management & Data Systems*, *101*(8), 400-405.

Clement, A., & Hurrell, C. (2007). Information/communications rights as a new environmentalism? Core environmental concepts for linking rights-oriented computerization movements. In K. L. Kramer & M. S. Elliott (Eds.), *Computerization movements and technology diffusion: From mainframes to ubiquitous computing* (p. 337-358). Medford, NJ: Information Today.

Cohen, J. (1992). A power primer. *Psychological Bulletin*, *112*(1), 155-159.

Cohen, J., Cohen, P., West, S. G., & Aiken, L. S. (2003). *Applied multiple regression/correlation analysis for the behavioural sciences* (3rd ed.). Mahwah, NH: Lawrence Erlbaum.

Coleman, E. G., & Hill, B. (2005). The social production of ethics in Debian and free software communities: Anthropological lessons for vocational ethics. In S. Koch (Ed.), *Free/open source software development* (p. 273-295). Hershey, PA: Idea Group.

Colford, S. (2009). Explaining free and open source software. *Bulletin of the American Society for Information Science and Technology*, *35*(2), 10-14.

Conklin, M., Howison, J., & Crow. (2005). Collaboration using OSSmole: a repository of FLOSS data and analyses. In A. E. Hassan, R. C. Holt, & S. Diehl (Eds.), *MSR 2005: Proceedings 2nd International Workshop on Mining Software Repositoriesd, St. Louis, Missouri* (p. 116-120). New York: ACM.

Conrath, D. W., & Mignen, O. P. (1990). What is being done to measure user satisfaction with EDP/MIS. *Information & Management*, *19*(1), 7-19.

Cooper, M. D. (1006). *Design of library automation systems: File structures, data structures, and tools.* New York: John Wiley.

Couper, M. P. (2005). Technology trends in survey data collection. *Social Science Computer Review*, *23*(4), 486-501.

Couper, M. P., Traugott, M. W., & Lamias, M. J. (2001). Web survey design and administration. *Public Opinion Quarterly*, *65*(2), 230-253.

Creswell, J. W. (2009). *Research design: Qualitative, quantitative, and mixed methods approaches* (3rd ed.). Thousand Oaks, CA: Sage.

Creswell, J. W., & Plano Clark, V. L. (2007). *Designing and conduction mixed methods research*. Thousand Oaks, CA: Sage Publications.

Creswell, J. W., & Plano Clark, V. L. (2011). *Designing and conducting mixed methods research* (2nd ed.). Los Angeles: Sage.

Crowston, K., Annabi, H., & Howison, J. (2003). *Defining open source software project success.* Paper read at Twenty-fourth International Conference on Information Systems, 14-17 December, at Seattle, WA.

Crowston, K., Annabi, H., Howison, J., & Masango, C. (2004). Towards a portfolio of FLOSS project success measures. In J. Feller, B. Fitzgerald, S. Hissam, & K. R. Lakhani (Eds.), *Collaboration, conflict and control: Proceedings of the 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland* (p. 29-33).

Crowston, K., Howison, J., & Annabi, H. (2006). Information systems success in free and open source software development: Theory and measures. *Software Process Improvement and Practice*, *11*(2), 123-148.

Crowston, K., Li, Q., Eseryel, U. Y., & Howison, J. (2007). Self-organization of teams for free/libre open source software development. *Information and Software Technology*, *49*(6), 564-575.

Crowston, K., Wei, K., & Howison, J. (2006). *Core and periphery in free/libre and open source software team communications.* Paper read at 39th Hawai'i International Conference on System System, Kaua'i, Hawai'i.

Crowston, K., Wei, K., Howison, J., & Wiggins, A. (2012). Free/libre open source software development: What we know and what we do not know. *ACM Computing Surveys, (in press)*.

Dahl, M., Banerjee, K., & Spalti, M. (2006). *Digital libraries : integrating content and systems*. Oxford: Chandos.

Damodaran, L. (1996). User involvement in the systems design process—a practical guide for users. *Behaviour & Information Technology*, *15*(6), 363-377.

Darby, A. (2006). Implementing an open source application in a college library – ECU's Pirate Source. *College & Undergraduate Libraries*, *13*(1), 41.

David, P. A., & Shapiro, J. S. (2008). Community-based production of open-source software: What do we know about the developers who participate? *Information Economics and Policy*, *20*(4), 364-398.

David, P. A., Waterman, A., & Arora, S. (2003). *FLOSS-US: The free/libre/open source software survey for 2003*. SIRPR-NSF Project on Open Source Software Working Paper.

Davis, C. H., & Lundeen, G. R. (1981). *Illustrative computer programming for libraries: Selected examples for informaiton specialists*. Westport, CT: Greenwood.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, *13*(3), 319-340.

de Vaus, D. (2002). *Analysing social science data: 50 key problems in data analysis*. London: Sage.

De Brabander, B., & Thiers, G. (1984). Successful information system development in relation to situational factores which affect effectve communication between MIS-users and EDP-specialists. *Management Science*, *30*(2), 137-155.

Dedrick, J., & West, J. (2007). Movement ideology vs. user pragmatism in the organizational adoption of open source software. In K. L. Kramer & M. S. Elliott (Eds.), *Computerization movements and technology diffusion: From mainframes to ubiquitous computing* (p. 427-452). Medford, NJ: Information Today.

Delone, W., & McLean, E. (1992). Information systems success: The quest for the dependent variable. *Information Systems Research*, *3*(1), 60-95.

Delone, W., & McLean, E. (2003). The DeLone and McLean model of information systems success: A ten-year update. *Journal of Management Information Systems*, *19*(4), 9-30.

Dempsey, B. J., Weiss, D., Jones, P., & Greenberg, J. (2002). Who is an open source software developer? Profiling a community of Linux developers. *Communications of the ACM*, *45*(2), 67-72.

Denove, C., & Power, J. D., IV. (2006). *Satisfaction: How every great company listens to the voice of the customer*. New York: Portfolio.

DiBona, C., Ockman, S., & Stone, M. (Eds.). (1999). *Open sources: Voices from the open source revolution*. Sebastopol, CA: O'Reilly.

Doll, W. J., Raghunathan, T. S., Lim, J. S., & Gupta, Y. P. (1995). A confirmatory factor analysis of the user information satisfaction instrument. *Information Systems Research*, *6*(2), 177-188.

Doll, W. J., & Torkzadeh, G. (1988). The measurement of end-user computing satisfaction. *MIS Quarterly*, *12*(2), 259-274.

Doll, W. J., & Torkzadeh, G. (1989a). A discrepancy model of end-user computing involvement. *Management Science*, *35*(10), 1151-1171.

Doll, W. J., & Torkzadeh, G. (1989b). The measurement of end-user software involvement. *Omega*, *18*(4), 399-406.

Doll, W. J., & Torkzadeh, G. (1991). The measurement of end-user computing satisfaction: Theoretical and methodological issues. *MIS Quarterly*, *15*(1), 5-10.

Doll, W. J., Xia, W., & Torkzadeh, G. (1994). A confirmatory factor analysis of the end-user computing satisfaction instrument. *MIS Quarterly*, *18*(4), 453-461.

Edström, A. (1977). User influence and the success of MIS projects: A contingency approach. *Human Relations*, *30*(7), 589-607.

Egan, S. (2005). *Open source messaging application development: Building and extending gaim*. Berkeley, CA: Apress.

Elliott, M. S. (2007). Examining the success of computerization movements in the ubiquitous computing era: Free and open source software movements. In K. L. Kramer & M. S. Elliott (Eds.), *Computerization movements and technology diffusion: From mainframes to ubiquitous computing* (p. 359-380). Medford, NJ: Information Today.

Elliott, M. S., & Kraemer, K. L. (2007). Comparative perspective on computerization movements: Implications for ubiquitous computing. In K. L. Kramer & M. S. Elliott (Eds.), *Computerization movements and technology diffusion: From mainframes to ubiquitous computing* (p. 521-545). Medford, NJ: Information Today.

Ellis, P. D. (2010). *The essential guide to effect sizes: Statistical power, meta-analysis, and the interpretation of research results*. Cambridge, UK: Cambridge University Press.

Etezadi-Amoli, J., & Farhoomand, A. F. (1996). A structural model of end user computing satisfaction and user performance. *Information & Management*, *30*(2), 65-73.

Feller, J., & Fitzgerald, B. (2002). *Understanding open source software development*. London: Addison-Wesley.

Field, A. (2009). *Discovering statistics using SPSS (and sex and drugs and rock 'n'roll)* (3rd ed.). Los Angeles: Sage.

Fink, M. (2003). *The business and economics of Linux and open source*. Upper Saddle River, NJ: Prentice Hall PTR.

Flynn, B. A., & Flyn, E. J. (1999). Information-processing alternatives for coping with manufacturing environment complexity. *Decision Sciences*, *30*(4), 1021-1052.

Foster, S. T., Jr. (1999). User involvement during information systems development: A comparison of analyst and user perceptions of system acceptance. *Journal of Engineering and Technology Management*, *16*(3-4), 329-348.

Franz, C. R., & Robey, D. (1986). Organizational context, user involvement, and the usefulness of information systems. *Decision Sciences*, *17*(3), 329-356.

Free Software Foundation GNU Project. (2010). *Various licenses and comments about them*. Retrieved 5/9/2010, from `http://www.gnu.org/licenses/license-list.html`

Gabriel, R. P., & Goldman, R. (2002, May). Open source: Beyond the fairy tales. *Perspectives on Business Innovation*, 59-65.

Gacek, C., & Arief, B. (2004). The many meanings of open source. *IEEE Software*, *21*(1), 34-40.

Galetta, D. F., & Lederer, A. L. (1989). Some cautions on the measurement of user information satisfaction. *Decision Sciences*, *20*(3), 419-318.

Gallivan, M. J. (2001). Striking a balance between trust and control in a virtual organization: a content analysis of open source software case studies. *Information Systems Journal*, *11*(4), 277-304.

Gallivan, M. J., & Keil, M. (2003). The user-developer communication process: A critical case study. *Information Systems Journal*, *13*(1), 37-68.

Gatian, A. W. (1994). Is user satisfaction a valid measure of system effectiveness? *Information & Management*, *26*(3), 119-131.

Gefen, D., & Keil, M. (1998). The impact of developer responsiveness on perceptions of usefulness and ease of use: An extension of the Technology Acceptance Model. *The Data Base for Advances in Information Systems*, *29*(2), 35-48.

Gelderman, M. (1998). The relation between user satisfaction, usage of information systems, and performance. *Information & Management*, *34*(1), 11-18.

German, D. M. (2003). The GNOME project: A case study of open source, global software development. *Software Process Improvement and Practice*, *8*(4), 201-215.

Ghosh, R., & Prakash, V. V. (2000). The Orbiten free software survey. *First Monday*, *5*(7).

Ghosh, R. A., Glott, R., Krieger, B., & Robles, G. (2002). *Free/libre and open source software: Survey and study. part iv: Survey of developers*. Maastrict: International Institute of Infonomics/Merit.

Golden, B. (2005). *Succeeding with open source*. Boston: Addison-Wesley.

Goldman, R., & Gabriel, R. P. (2005). *Innovation happens elsewhere: Open source as a business strategy*. Amsterdam: Morgan-Kaufman.

González-Baharona, J. M., Løpez, L., & Robles, G. (2004). Community structure of modules in the Apache project. In J. Feller, B. Fitzgerald, S. Hissam, & K. R. Lakhani (Eds.), *Collaboration, conflict and control: Proceedings of the 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland* (p. 43-47).

González-Baharona, J. M., Robles, G., Andradas-Izquierdo, R., & Ghosh, R. A. (2008). A geographic origin of libre software developers. *Information Economics and Policy*, *20*(4), 356-363.

Gorman, G. E., & Clayton, P. (1997). *Qualitative research for the information professional: A practical handbook*. London: Library Association.

Graham, S. L. (2001). From research software to open source. In R. Wilhelm (Ed.), *Informatics: 10 years back, 10 years ahead* (p. 195-208). Berlin: Springer-Verlag.

Green, S. B. (1991). How many subjects does it take to do a regression analysis? *Multivariate Behavioral Research*, *26*(3), 499-510.

Gregor, S. (2002). A theory of theories in information systems. In S. Gregor & D. Hart (Eds.), *Information systems foundations: Building the theoretical base* (p. 1-20). Canberra: Australian National University.

Gregor, S. (2006). The nature of theory in information systems. *MIS Quarterly*, *30*(3), 611-642.

Griffiths, J. R., Johnson, F., & Hartley, R. J. (2007). User satisfaction as a measure of system performance. *Journal of Librarianship and Information Science*, *39*(3), 142-152.

Guimaraes, T., & Gupta, Y. P. (1988). Measuring top management satisfaction with the MIS department. *Omega*, *16*(1), 17-24.

Guimaraes, T., & Igbaria, M. (1997). Client/server system success: Exploring the human side. *Decision Sciences*, *28*(4), 851-876.

Guimaraes, T., Igbaria, M., & Lu, M.-T. (1992). The determinants of DSS success: An integrated model. *Decision Sciences*, *23*(2), 409-430.

Guimaraes, T., Staples, D. S., & McKeen, J. D. (2003). Empirically testing some main user-related factors for systems development quality. *Quality Management Journal*, *10*(4), 39-54.

Guimaraes, T., Yoon, V. Y., & Clevenson, A. (2001). Exploring some determinants of ES quality. *Quality Management Journal*, *8*(1), 23-33.

Hair, J. F., Babin, B., Money, A. H., & Samouel, P. (2003). *Essentials of business methods research*. Hoboken, NJ: Wiley.

Hair, J. F., Jr., Black, W. C., Babin, B. J., Anderson, R. E., & Tatham, R. L. (2006). *Multivariate data analysis* (6th ed.). Upper Saddle River, NJ: Pearson Prentice Hall.

Hall, J. A., & Kiedtka, S. L. (2005). Financial performance, CEO compensation, and large-scale information technology outsourcing decisions. *Journal of Management Information Systems*, *22*(1), 193-221.

Hansen, P. B. (1970). The nucleus of a multiprogramming system. *Communications of the ACM*, *13*(4), 238-241, 250.

Harrison, A. W., & Rainer, R. K., Jr. (1996). A general measure of user computer satisfaction. *Computers in Human Behavior*, *12*(1), 79-92.

Hars, A., & Ou, S. (2002). Working for free? Motivations for participating in open-source projects. *International Journal of Electronic Commerce*, *6*(3), 25-39.

Hawk, S. R., & Dos Santos, B. L. (1991). Successful system development: The effect of situational factors on alternate user roles. *IEEE Transactions on Engineering Management*, *38*(4), 316-327.

Hawk, S. R., & Raju, N. S. (1991). Test-retest reliability of user information satisfaction: A comment on Galletta and Lederer's paper. *Decision Sciences*, *22*(4), 1165-1170.

Hayes, B. E. (2008). *Measuring customer satisfaction and loyalty: Survey design, use, and statistical analysis methods* (3rd ed.). Milwaukee, WI: ASQ Quality Press.

He, J., & King, W. R. (2008). The role of user participation in information systems development: Implications from a meta-analysis. *Journal of Management Information Systems*, *25*(1), 301-331.

Helling, J. (2010). Cutting the proprietary cord: A case study of one library's decision to migrate to an open source ILS. *Library Review*, *59*(9), 702-707.

Hendrickson, A. R., Glorfeld, K., & Cronan, T. P. (1994). On the repeated test-retest reliability of the end-user computing satisfaction instrument: A comment. *Decision Sciences*, *25*(4), 655-667.

Henn, M., Weinstein, M., & Foard, N. (2006). *A short introduction to social research*. London: Sage Publications.

Herbold, R. (1960). Quad i. *Communications of the ACM*, *3*(2), 74.

Hinkin, T. R. (1998). A brief tutorial on the development of measures for use in survey questionnaires. *Organizational Research Methods*, *1*(1), 104-121.

Holck, J., & Jorgensen, N. (2005). Do not check in on red: Control meets anarchy in two open source projects. In S. Koch (Ed.), *Free/open source software development* (p. 1-26). Hershey, PA: Idea Group.

Hopkins, T. (2002). Renovating the collected algorithms from ACM. *Communications of the ACM*, *28*(1), 59-74.

Howison, J., Conklin, M., & Crowston, K. (2006). FLOSSmole: A collaborative repository for FLOSS research data and analyses. *International Journal of Information Technology and Web Engineering*, *1*(3), 17-26.

Howison, J., & Crowston, K. (2004). *The problems and perils of mining SourceForge.* Paper read at Mining Software Repositories Workshop, International Conference on Software Engineering, 25 May, at Edinburgh.

Hulin, C., & Cudeck, R. (2001). Cronbach's alpha on two-item scales. *Journal of Consumer Psychology*, *10*(1&2), 55.

Hulin, C., Netemeyer, R., & Cudeck, R. (2001). Can a reliability coefficient be too high? *Journal of Consumer Psychology*, *10*(1&2), 55-57.

Hunt, F., & Johnson, P. (2002). *On the Pareto distribution of SourceForge projects.* Paper read at Open Source Software Development Workshop, 25-26 February 2002, at Newcastle upon Tyne.

Hunton, J. E. (1997). Effects of user participation on systems development: A longitudinal field experiment. *MIS Quarterly*, *21*(4), 359-388.

Hwang, M. I., & Thorn, R. G. (1999). The effect of user engagement on system success: A meta-analytical integration of research findings. *Information & Management*, *35*(4), 229-236.

Igbaria, M., Guimaraes, T., & Davis, G. B. (1995). Testing the determinants of microcomputer usage via a structural equation model. *Journal of Management Information Systems*, *11*(4), 87-114.

Igbaria, M., & Iivari, J. (1995). The effects of self-efficacy on computer usage. *Omega*, *23*(6), 587-605.

Igbaria, M., & Nachman, S. A. (1990). Correlates of user satisfaction with end user computing: An exploratory study. *Information & Management*, *19*(2), 73-82.

Iivari, J. (1997). User information satisfaction: A critical review. In A. Kent, H. Lancour, W. Nasri, & J. Daily (Eds.), *Encyclopedia of library and information science* (Vol. 60, p. 341-34). New York: Marcel Dekker.

Iivari, J., & Ervasti, I. (1994). User information satisfaction: IS implementation and effectiveness. *Information & Management*, *27*(4), 205-220.

Iivari, J., & Karjalainen, M. (1989). Impact of prototyping on user information satisfaction during the IS specification phase. *Information & Management*, *17*(1), 31-45.

IIvari, N. (2004). *Exploring the rhetoric on representing the user: Discourses on user involvement in system development.* Paper read at 25th International Conference on Information Systems, 12-15 December, at Washington, DC.

Ilgen, D. R., Hollenbeck, J. R., Johnson, M., & Jundt, D. (2005). Teams in organisations: From Input-Process-Output models to IMOI models. *Annual Review of Psychology*, *56*, 517-543.

Ives, B., Olson, M. H., & Baroudi, J. J. (1983). The measurement of user information satisfaction. *Communications of the ACM*, *26*(10), 785-793.

Jang, C.-L. (2009). The moderating effect of self-efficacy on the antecedents of online satisfaction with an electronic document exchange system. *Social Behavior and Personality*, *37*(2), 239-254.

Jensen, C., & Scacchi, W. (2007). *Role migration and advancement processes in OSSD projects: A comparative case study.* Paper read at 29th International Conference on Software Engineering.

Jiang, J., Chen, E., & Klein, G. (2002). The importance of building a foundation for user involvement in information systems projects. *Project Management Journal*, *33*(1), 20-26.

Jiang, J. J., Klein, G., Roan, J., & Lin, J. T. (2001). IS service performance: Self-perceptions and user perceptions. *Information & Management*, *38*(8), 499-506.

Jin, L., Robey, D., & Boudreau, M.-C. (2007). Beyond development: A research agenda for investigating open source software user communities. *Information Resources Management Journal*, *20*(1), 68-80.

Joshi, K. (1990). An investigation of equity as a determinant of user information satisfaction. *Decision Sciences*, *21*(4), 786-807.

Kappelman, L. A. (1995). Measuring user involvement: A diffusion of innovation perspective. *The Data Base for Advances in Information Systems*, *26*(2 & 3), 65-86.

Kekre, S., Krishnan, M. S., & Srinivasan, K. (1995). Drivers of customer satisfaction for software products: Implications for design and service. *Management Science*, *41*(5), 1456-1470.

Kettinger, W. J., & Lee, C. C. (1994). Perceived service quality and user satisfaction with the information services function. *Decision Sciences*, *25*(5/6), 737-766.

Khalifa, M., & Liu, V. (2003). Determinants of satisfaction at different adoption stages of Internet-based sevices. *Journal of the Association for Information Systems*, *4*(5), 206-231.

Kim, K. K. (1990). Task characteristics, decentralization, and the success of hospital information systems. *Information & Management*, *19*(2), 83-93.

Kim, S., & McHaney, R. (2000). Validation of the end-user computing satisfaction instrument in case tool environments. *Journal of Computer Information Systems*, *41*(1), 49-55.

Kleinbaum, D. G., Kupper, L. L., Nizam, A., & Muller, K. E. (2008). *Applied regression analysis and other multivariable methods* (4th ed.). Belmont, CA: Thomson Brooks/Cole.

Knuth, D. (1992a). The error log of TEX (1978-1991). In *Literate programming* (p. 293-339). Stanford, CA: Center for the Study of Language and Information.

Knuth, D. (1992b). The errors of TEX (1989). In *Literate programming* (p. 243-291). Stanford, CA: Center for the Study of Language and Information.

Krishnamurthy, S. (2002). Cave or community? An empirical examination of 100 mature open source projects. *First Monday*, *7*(6).

Kroah-Hartman, G., Corbet, J., & McPherson, A. (2009). *Linux kernel development: How fast it is going, who is doing it, what they are doing, and who is sponsoring it: An August 2009 update*. San Francisco, CA: The Linux Foundation.

Lakhani, K. R., & Hippel, E. von. (2003). How open source software works: "Free" user-to-user assistance. *Research Policy*, *32*(7), 923-943.

Lamont, M. (2009). Gender, technology, and libraries. *Information Technology and Libraries*, *28*(3), 137-142.

Larsen, T. J. (2009). A multilevel explanation of end-user computing satisfaction with an enterprise resource planning system within an international manufacturing organization. *Computers in Industry*, *60*(9), 657-668.

Lascarides, M. (2009). Infomaki: An open source, lightweight usability testing tool. *The Code4Lib Journal*(8).

Lawrence, M., & Low, G. (1993). Exploring individual user satisfaction within user-led development. *MIS Quarterly*, *17*(2), 195-208.

Lee, J.-F., & Chan, T.-Y. (2004). Organisational structure of "user collaboration community": Insights from the case of an open source software project. In J. Feller, B. Fitzgerald, S. Hissam, & K. R. Lakhani (Eds.), *Collaboration, conflict and control: Proceedings of the 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland* (p. 58-63).

Lee, S. M., Kim, Y. R., & Lee, J. (1995). An empirical study of the relationship among end-user information acceptance, training, and effectiveness. *Journal of Management Information Systems*, *12*(2), 189-202.

Lee, S.-Y. T., Kim, H.-W., & Gupta, S. (2009). Measuring open source software success. *Omega*, *37*(2), 426-438.

Leedy, P. D., & Ormrod, J. E. (2001). *Practical research: Planning and design* (7th ed.). Upper Saddle River, NJ: Merrill Prentice-Hall.

Leonard, A. (2000). *Chapter 1: Boot time.* Salon Magazine.

Lincoln, Y., & Guba, E. (1985). *Naturalistic inquiry*. Los Angeles: Sage.

Linux Devices. (2008). *Linux still top embedded OS.* Retrieved 5/9/2010, from `http://www.linuxfordevices.com/c/a/News/Linux-still-top-embedded-OS/`

Lu, H.-P., & Chiou, M.-J. (2010). The impact of individual differences on e-learning system satisfaction: A contingency approach. *British Journal of Educational Technology*, *41*(2), 307-323.

Maass, W. (2004). Inside an open source software community: Empirical analysis on individual and group level. In J. Feller, B. Fitzgerald, S. Hissam, & K. R. Lakhani (Eds.), *Collaboration, conflict and control: Proceedings of the 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland* (p. 65-70).

Mahmood, M. A., & Becker, J. D. (1986). Effect of organizational maturity on end-users' satisfaction with information systems. *Journal of Management Information Systems*, *2*(3), 37-64.

Mahmood, M. A., Burn, J. M., Gemoets, L. A., & Jacquez, C. (2000). Variables affecting information technology and user-satisfaction: A meta-analysis of the empirical literature. *International Journal of Human-Computer Studies*, *52*(4), 751-771.

Mann, C., & Stewart, F. (2003). Internet interviewing. In J. Gubrium & J. Holstein (Eds.), *Postmodern interviewing* (p. 81-105). Thousand Oaks, CA: Sage.

Marks, M. A., Mathieu, J. E., & Zaccaro, S. J. (2001). A temporally based framework and taxonomy of team processes. *Academy of Management Review*, *26*(3), 356-376.

Markus, M. L. (1983). Power, politics and MIS implementation. *Communications of the ACM*, *26*(6), 430-444.

Markus, M. L. (2007). The governance of free/open source software projects: monolithic, multidimensional, or configurational? *Journal of Management and Governance*, *11*(2), 151-163.

Markus, M. L., & Mao, J.-Y. (2004). Participation in development and implementation–updating an old, tired concept for today's IS contexts. *Journal of the Association for Information Systems*, *5*(11-12), 514-544.

McGill, T., & Dixon, M. (2001). Spreadsheet knowledge: An exploratory study. In *Managing information in a global economy* (p. 621-625). Vancouver, B.C.: Idea Group.

McGill, T., Hobbs, V., & Klobas, J. (2003). User-developed applications and information systems success: A test of DeLone and McLean's model. *Information Resources Management Journal*, *16*(1), 24-45.

McGill, T., & Klobas, J. (2008). User developed application success: sources and effects of involvement. *Behaviour & Information Technology*, *27*(5), 407-422.

McHaney, R., & Cronan, T. P. (1998). Computer simulation success: On the use of the End-User Computing Satisfaction instrument: A comment. *Decision Sciences*, *29*(2), 525-536.

McHaney, R., Hightower, R., & Pearson, J. (2002). A validation of the end-user computing satisfaction instrument in Taiwan. *Information Resources Management Journal*, *39*(6), 503-511.

McHaney, R., Hightower, R., & White, D. (1999). EUCS test-retest reliability in representational model decision support systems. *Information & Management*, *36*(2), 109-119.

McKeen, J. D., Guimaraes, T., & Wetherbe, J. C. (1994). The relationship between user participation and user satisfaction: An investigation of four contingency factors. *MIS Quarterly*, *18*(4), 427-451.

McKinney, V., Yoon, K., & Zahedi, F. M. (2002). The measurement of Web-customer satisfaction: An expectation and disconfimation approach. *Information Systems Research*, *13*(3), 396-315.

McNamara, N., & Kirakowski, J. (2011). Measuring user-satisfaction with electronic consumer products: The Consumer Products Questionnaire. *International Journal of Human-Computer Studies*, *69*(6), 375-386.

Melone, N. P. (1990). A theoretical assessment of the user-satisfaction construct in information systems research. *Management Science*, *36*(1), 76-91.

Miller, J., & Doyle, B. A. (1987). Measuring the effectiveness of computer-based information systems in the financial sector. *MIS Quarterly*, *11*(1), 107-124.

Millington, P., & Nixon, W. J. (2007). EPrints 3 pre-launch briefing. *Ariadne*(50).

Mirani, R., & King, W. R. (1994). The development of a measure for end-user computing support. *Decision Sciences*, *25*(4), 481-498.

Molyneux, R. E. (2009). Evergreen in context. *Bulletin of the American Society for Information Science and Technology*, *35*(2), 26-30.

Monge, P., Bachman, S., Dillard, J. P., & Eisenberg, E. (1983). Communicator competence in the workplace: Model testing and scale

development. In M. Burgoon (Ed.), *Communication yearbook 5* (p. 89-105). New Brunswick, NJ: International Communication Association.

Montezami, A. R. (1988). Factors affecting information satisfaction in the context of small business environment. *MIS Quarterly*, *12*(2), 239-256.

Moody, G. (2002). *Rebel code: Linux and the open source revolution*. London: Penguin.

Morgan, E. L. (2008). MyLibrary: A digital library framework and toolkit. *Information Technology and Libraries*, *27*(3), 12-24.

Mullany, M. J., Tan, F. B., & Gallupe, R. B. (2006). *The S-Statistic: A measure of user satisfaction baseed on Herzberg's theory of motivation.* Paper read at ACIS 2006.

Mullany, M. J., Tan, F. B., & Gallupe, R. B. (2007). *The impact of analyst-user cognitive style differences on user satisfaction.* Paper read at 11th Pacific-Asia Conference on Information Systems.

Munro, M. C., Huff, S. L., Marcolin, B. L., & Compeau, D. R. (1997). Understanding and measuring user competence. *Information & Management*, *33*(1), 45-57.

Muylle, S., Moenaert, R., & Despontin, M. (2004). The conceptualization and empirical validation of web site user satisfaction. *Information & Management*, *41*(5), 543-560.

Neiderman, F., Davis, A., Greiner, M. E., Wynn, D., & York, P. T. (2006a). A research agenda for studying open source I: A multi-level framework. *Communications of the Association for Information Systems*, *18*(7), 129-149.

Neiderman, F., Davis, A., Greiner, M. E., Wynn, D., & York, P. T. (2006b). A research agenda for studying open source II: View through the lens of referent discipline theories. *Communications of the Association for Information Systems*, *18*(8), 150-175.

Nelson, M., Sen, R., & Subramaniam, C. (2006). Understanding open source software: A research classification framework. *Communications of the Association for Information Systems*, *17*(12), 266-287.

Nelson, R. R., & Cheney, P. H. (1987). Training end users: an exploratory study. *MIS Quarterly*, *11*(4), 547-549.

Netcraft. (2010). *Web server survey.* Retrieved 13/7/2010, from `http://news.netcraft.com/archives/category/web-server-survey/`

Ngamkajornwiwat, K., Zhang, D., Koru, A. G., Zhou, L., & Nolker, R. (2008). *An exploratory study on the evolution of OSS developer communities.* Paper read at 41st Hawaii International Conference on System Sciences.

Nielsen, J. (2006, October). Participation inequality: Encouraging more users to contribute. *Alertbox*.

Ohira, M., Ohsugi, N., Ohoka, T., & Matsumoto, K.-I. (2005). Accelerating cross-project knowledge collaboration using collaborative filtering and social networks. In A. E. Hassan, R. C. Holt, & S. Diehl

(Eds.), *MSR 2005: Proceedings 2nd International Workshop on Mining Software Repositories, St. Louis, Missouri* (p. 111-115). New York: ACM.

Oliver, R. L. (1996). *Satisfaction: A behavioral perspective on the consumer*. Boston: Irwin McGraw-Hill.

Oliver, R. L. (2010). *Satisfaction: A behavioral perspective on the consumer* (2nd ed.). Armonk, NY: M.E. Sharpe.

Olson, M. H., & Ives, B. (1980). *Measuring user involvement in information system development.* Paper read at 25th International Conference on Information Systems, 12-15 December, at Washington, DC.

Ong, C.-S., & Lai, J.-Y. (2007). Measuring user satisfaction with knowledge management systems: Scale development, purification, and initial test. *Computers in Human Behavior*, *23*(3), 1329-1346.

Open Source Initiative. (2010a). *History of the OSI.* Retrieved 5/9/2010, from `http://www.opensource.org/history`

Open Source Initiative. (2010b). *Licenses by name.* Retrieved 5/9/2010, from `http://opensource.org/licenses/alphabetical`

Orlikowski, W. J., & Baroudi, J. J. (1991). Studying information technology in organizations: Research approaches and assumptions. *Information Systems Research*, *2*(1), 1-18.

Orwell, G. (1968). Politics and the English language. In S. Orwell & I. Angus (Eds.), *Collected essays, journalism, and letters* (Vol. 4, p. 127-140). Secker & Warburg.

Palvia, P. C. (1996). A model and instrument for measuring small business user satisfaction with information technology. *Information & Management*, *31*, 151-163.

Palvia, P. C., & Palvia, S. C. (1999). An examination of the IT satisfaction of small business users. *Information & Management*, *35*(3), 127-137.

Palvia, S. C. (2000/2001). Effectiveness of asynchronous and synchronous modes for learning computer software for end users: An experimental investigation. *Journal of Computer Information Systems*, *41*(2), 99-109.

Peytchev, A., Couper, M. P., McCabe, S. E., & Crawford, S. D. (2006). Web survey design: Paging vs scrolling. *Public Opinion Quarterly*, *70*(4), 596-607.

Pickard, A. J. (2007). *Research methods in information.* London: Facet.

Porter, S. R. (2004, Spring). Raising response rates: What works? *New Directions for Institutional Research*, 5-21.

Porter, S. R., Whitcomb, M. E., & Weitzer, W. H. (2004, Spring). Multiple surveys of students and survey fatigue. *New Directions for Institutional Research*, 63-73.

Powers, R. F., & Dickson, G. W. (1973). MIS project management: Myths, opinions, and reality. *California Management Review*, *15*(3), 147-156.

Presser, S., Couper, M. P., Lessler, J. T., Martin, E., Martin, J., Rothgeb, J. M., et al. (2004). Methods for testing and evaluating survey questions. *Public Opinion Quarterly*, *68*(1), 109-130.

Rai, A., Lang, S. S., & Welker, R. B. (2002). Assessing the validity of IS success models: An empirical test and theoretical models. *Information Systems Research*, *13*(1), 50-69.

Ramasubbu, N., Mithas, S., & Krishnan, M. (2008). High tech, high touch: The effect of employee skills and customer heterogeneity on customer satisfaction with enterprise system support services. *Decision Support Systems*, *44*(2), 509-525.

Ranganathan, S. R. (1963). *The five laws of library science* (2nd ed.). Bombay: Asia Publishing House.

Ransom, J., Cormack, C., & Blake, R. (2009). How hard can it be? : Developing in open source. *The Code4Lib Journal*(7).

Raymond, E. S. (2001). The cathedral and the bazaar. In *The cathedral and the bazaar: Musings on Linux and open source by an accidental revolutionary* (Rev. ed., p. 19-63). Sebastopol, CA: O'Reilly.

Raymond, L. (1985). Organizational characteristics and MIS success in the context of small business. *MIS Quarterly*, *9*(1), 37-52.

Raymond, L. (1987). Validating and applying user satisfaction as a measure of MIS success in small organizations. *Information & Management*, *12*(4), 173-179.

Rea, L. M., & Parker, R. A. (2005). *Designing and conducting survey research: A comprehensive guide* (3rd ed.). San Francisco, CA: Jossey-Bass.

Rizzo, J. R., House, R. J., & Lirtzman, S. I. (1993). Role conflict and ambiguity. In W. O. Bearden, R. G. Netemeyer, & M. F. Mobley (Eds.), *Handbook of marketing scales: Multi-item measures for marketing and consumer behavior research* (p. 283-285). Newbury Park, CA: Sage.

Roberts, J. A., Hann, I.-H., & Slaughter, S. A. (2006). Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects. *Management Science*, *52*(7), 984-999.

Robey, D., & Farrow, D. (1982). User involvement in information system development: A conflict model and empirical test. *Management Science*, *28*(1), 73-85.

Robey, D., Farrow, D., & Franz, C. R. (1989). Group process and conflict in system development. *Management Science*, *35*(10), 1172-1191.

Rogers, E. M. (2003). *Diffusion of innovations* (5th ed.). New York: Free Press.

Rosenkerantz, S. A., Luthans, F., & Hennessey, H. W. (1983). Role conflict and ambiguity scales: An evaluation of psychometric properties and the roles of social desirability response bias. *Educational and Psychological Measurement*, *43*(4), 957-970.

Rothfuss, G. J. (2002). *A framework for open source projects*. Unpublished master's thesis, Department of Information Technology, University

of Zurich, Zurich.

Ruane, J. M. (2005). *Essentials of research methods: A guide to social science research*. Malden, MA: Blackwell.

Saarinen, T. (1996). An expanded instrument for evaluating information system success. *Information & Management*, *31*(2), 103-118.

Sabherwal, R., Jeyaraj, A., & Chowa, C. (2006). Information system success: Individual and organizational determinants. *Management Science*, *52*(12), 1849-1864.

Sagers, G. W. (2004). The influence of network governance factors on success in open source software development practices. In *ICIS 2004 Proceedings* (p. 427-438). Atlanta, GA: AIS.

Saleem, N. (1996). An empirical test of the contingency approach to user participation in information systems development. *Journal of Management Information Systems*, *13*(1), 145-166.

Sanders, G. L., & Courtney, J. F. (1985). A field study of organisational factors influencing DSS success. *MIS Quarterly*, *9*(1), 77-93.

Sandvig, J. C., Tyran, C. K., & Ross, S. C. (2005). Determinants of graduating MIS student starting salary in boom and bust job markets. *Communications of the Association for Information Systems*, *16*(1), 604-624.

Santhanam, R., Guimaraes, T., & George, J. F. (2000). An empirical investigation of ODSS impact on individuals and organizations. *Decision Support Systems*, *30*(1), 51-72.

Scacchi, W. (2002). Understanding the requirements for developing open source software systems. *IEE Proceedings Software*, *149*(2), 24-39.

Scacchi, W. (2004). Free and open source development practices in the game community. *IEEE Software*, *21*(1), 59-66.

Scacchi, W. (2007a). Emerging patterns of intersection and segmentation when computerization movements interact. In K. L. Kramer & M. S. Elliott (Eds.), *Computerization movements and technology diffusion: From mainframes to ubiquitous computing* (p. 381-404). Medford, NJ: Information Today.

Scacchi, W. (2007b). Free/open source software development: Recent research results and methods. *Advances in Computers*, *69*, 243-295.

Scharf, E. D. (2002). *Open source: A conceptual framework for collaborative artifact and knowledge construction*. Unpublished doctoral dissertation, University of Colorado, Department of Computer Science.

Schrauf, R. W., & Navarro, E. (2005). Using existing tests and scales in the field. *Field Methods*, *17*(4), 373-393.

Schweik, C., & Semenov, A. (2003). The institutional design of open souce programming: Implications for addressing complex public policy and management problems. *First Monday*, *1*.

Scozzi, B., Crowston, K., Eseryel, U. Y., & Li, Q. (2008). *Shared mental models among open source software developers.* Paper read at 41st

Hawaii International Conference on System Sciences.

Seddon, P., & Yip, S.-K. (1992). An empirical evaluation of user information satisfaction (UIS) measures for use with general ledger accounting software. *Journal of Information Systems*, *6*(1), 75-92.

Seddon, P., & Yip, S.-K. (1994). *A partial test and development of the DeLone and McLean model of IS success.* Paper read at International Conference on Information Systems, 14-17 December, at Vancouver, B.C.

Seddon, P. B., Staples, S., Patnayakuni, R., & Bowtell, M. (1999). Dimensions of information systems success. *Communications of the Association for Information Systems*, *2*(20).

Sedera, D., & Tan, F. T. C. (2005). *User satisfaction: An overarching measure of enterprise system success.* Paper read at Pacific Asia Conference on Information Systems, 7-10 July, Bangkok, Thailand.

Sengupta, K., & Zviran, M. (1997). Measuring user satisfaction in an outsourcing environment. *IEEE Transactions on Engineering Management*, *44*(4), 414-421.

Sessoms, P., & Sessoms, E. (2008). LibraryH3lp: A new flexible chat reference system. *The Code4Lib Journal*(4).

Shah, S. K. (2006). Motivation, governance, and the viability of hybrid forms in open source software development. *Management Science*, *52*(7), 1000-1014.

Sharma, R., & Yetton, P. (2007). The contingent effects of training, technical complexity, and task interdependence on successful information systems implementation. *MIS Quarterly*, *31*(2), 219-238.

Sharma, S., Durand, R. M., & Gur-Arie, O. (1981). Identification and analysis of moderator variables. *Journal of Marketing Research*, *18*(291-300).

Shih, C.-C., & Huang, S.-J. (2010). Exploring the relationship between organizational culture and software process improvement deployment. *Information & Management*, *47*(5-6), 271-281.

Shih, H.-P. (2004). An empirical study on predicting user acceptance of e-shopping on the Web. *Information & Management*, *41*(3), 351-368.

Shirani, A., Aiken, M., & Reithel, B. (1994). A model of user information satisfaction. *Data Base*, *25*(4), 17-23.

Simon, S. J., Grover, F., Teng, J. T., & Whticomb, K. (1996). The relationship of information training methods and cognitive ability to end-user satisfaction, comprehension, and skill transfer: A longitudinal field study. *Information Systems Research*, *7*(4), 466-490.

Singh, V., Twidale, M. B., & Nichols, D. M. (2009). *Users of open source software: How do they get help?* Paper read at 42nd Hawaii International Conference on System Sciences.

Smith, B. (n.d.). *A quick guide to GPLv3.* Retrieved 5/9/2010, from `http://www.gnu.org/licenses/quick-guide-gplv3.html`

Smith, L. W. (1961). What is proprietary in mathematical programming? impressions of a panel discussion. *Communications of the ACM*, *4*(12), 542, 594.

Somers, T. M., Nelson, K., & Karimi, J. (2003). Confirmatory factor analysis of the end-user computing satisfaction instrument : replication within an ERP doman. *Decision Sciences*, *34*(3), 595-621.

Spaeth, S., Stuermer, M., & von Krogh, G. (2007). *Sampling in open source software development: The case for using the Debian GNU/Linux distribution.* Paper read at 40th Hawai'i International Conference on System System, Kaua'i, Hawai'i.

Stallman, R. (1999). The GNU operating system and the free software movement. In C. DiBona, S. Ockman, & M. Stone (Eds.), *Open sources: Voices from the open source revolution* (p. 53-70). Sebastopol, CA: O'Reilly.

Stallman, R. (2002). Free software reality v. perception: Letter to the editor. *Communications of the ACM*, *45*(7), 11-12.

Stallman, R. (2010). *Did you say "intellectual property"? It's a seductive mirage.* Available from `http://www.gnu.org/philosophy/not-ipr.html`

Stark, J. (2006). Peer reviews as a quality management technique in open-source software development projects. In G. Goos, J. Hartmanis, & J. van Leeuwen (Eds.), *Software quality — ECSQ 2002* (p. 340-350). Berlin: Springer.

Stewart, K. J., & Gosain, S. (2006). The impact of ideology in effectiveness in open source software development teams. *MIS Quarterly*, *30*(2), 291-314.

Stoutenborough, J. W. (2008). Semantic differential technique. In *Encyclopedia of survey research methods.* Thousand Oaks, CA: Sage Online.

Stranack, K. (2007). The reSearcher software suite: A case study of library collaboration and open source software development. *Serials Librarian*, *55*(1/2), 117-139.

Studer, M. (2007). Community structure, individual participation and the social construction of merit. In J. Feller, B. Fitzgerald, W. Scacchi, & A. Sillitti (Eds.), *Open source development, adoption and innovation: IFIP Working Group 2.13 on Open Source Software, June 11–14, 2007, Limerick, Ireland* (p. 161-172). Boston: Springer.

Stürmer, M. (2005). *Open source community building.* Licentiate, Faculty of Economics and Social Science, University of Bern, Bern.

Subramanyan, R., Weisstein, F. L., & Krishnan, M. (2010). User participation in software development projects. *Communications of the ACM*, *53*(3), 137-141.

Sun, H., & Zhang, P. (2006). The role of moderating factors in user technology acceptance. *International Journal of Human-Computer Studies*, *64*(2), 53-78.

Sun, J., & Poole, M. S. (2010). Capturing user readiness to interact with information systems: An activity perspecive. *The Data Base for Advances in Information Systems*, *41*(2), 89-109.

Tait, P., & Vessey, I. (1988). The effect of user involvement on system success: A contingency approach. *MIS Quarterly*, *12*(1), 91-108.

Tan, B. W., & Lo, T. W. (1990). Validation of a user satisfaction instrument for office automation success. *Information & Management*, *18*(4), 203-208.

Thong, J. Y., & Yap, C.-S. (1996). Information systems effectiveness: A user satisfaction approach. *Information Processing & Management*, *32*(5), 601-610.

Tojib, D. R., Sugianto, L.-F., & Sendjaya, S. (2008). User satisfaction with business-to-employee portals: conceptualisation and scale development. *European Journal of Information Systems*, *17*(6), 649-667.

Torkzadeh, G., & Lee, J. (2003). Measures of perceived end-user computing skills. *Information & Management*, *40*(7), 607-615.

Torvalds, L. (1991). *Free minix-like kernel sources for 386-at [comp.os.minix newsgroup].*

Torvalds, L., & DIamond, D. (2002). *Just for fun: The story of an accidental revolutionary*. New York: HarperBusiness.

Townsend, A. M., Demaire, S. M., & Hendrickson, A. R. (2001). Desktop video conferencing in virtual workgroups: Anticipation, system evaluation and performance. *Information Systems Journal*, *11*(3), 213-227.

Umbach, P. D. (2004, Spring). Web surveys: Best practices. *New Directions for Institutional Research*, 23-38.

University of Minnesota. Department of Psychology. (1967). *Minnesota satisfaction questionnaire.* Retrieved 5/0/2010, from `http://www.psych.umn.edu/psylabs/vpr/msqinf.htm`

Vankatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User acceptance of information technology: toward a unified view. *MIS Quarterly*, *27*(3), 425-478.

Ven, K., & Verelst, J. (2008). The impact of ideology on the organizational adoption of open source software. *Journal of Database Management*, *19*(2), 58-72.

von Krogh, G., Spaeth, S., & Lakhani, K. R. (2003). Community, joining, and specialization in open source software innovation: a case study. *Research Policy*, *32*(7), 1149-1157.

Walejko, G. (2009). Online survey: Instant publication, instant mistake, all of the above. In E. Hargittai (Ed.), *Research confidential: Solutions to problems most social scientists pretend they never have* (p. 101-121). Ann Arbor: University of Michigan Press.

Walker, J. (2007). *DSpace: A case study in sustainability.* Retrieved 5/9/2010, from `http://www.oss-watch.ac.uk/resources/cs-dspace.xml`

Want, Y.-S., & Liao, Y.-W. (2007). The conceptualization and measurement of m-commerce user satisfaction. *Computers in Human Behavior*, *23*(1), 381-398.

Watson, R. T., Pitt, L. F., Cunningham, C. J., & Nel, D. (1993). User satisfaction and service quality of the is department: Closing the gaps. *Journal of Information Technology*, *8*(4), 257-265.

West, J., & O'Mahony, S. (2008). The role of participation architecture in growing sponsored open source communities. *Industry and Innovation*, *15*(2), 145-168.

Whitten, D. (2004-2005). User information satisfaction scale reduction: Application in an outsourcing environment. *Journal of Computer Information Systems*, *45*(2), 17-26.

Williams, D., & Xiong, L. (2009). Herding cats online: Real studies of virtual communities. In E. Hargittai (Ed.), *Research confidential: Solutions to problems most social scientists pretend they never have* (p. 122-140). Ann Arbor: University of Michigan Press.

Williams, S. (2002). *Free as in freedom: Richard Stallman's crusade for free software*. Sebastopol, CA: O'Reilly.

Willinsky, J. (2006). *The access principle*. Cambridge, MA: MIT Press.

Witten, I. H., & Bainbridge, D. (2007). A retrospective look at Greenstone: Lessons from the first decade. In *Cdl '07: Proceedings of the 7th acm/ieee-cs joint conference on digital libraries* (p. 147-156). New York: ACM.

Witten, I. H., Boddie, S. J., Bainbridge, D., & McNab, R. J. (2000). Greenstone: A comprehensive open-source digital library software system. In *DL '00: Proceedings of the fifth ACM conference on Digital Libraries* (p. 113-121). New York: ACM.

Wixom, B. H., & Todd, P. A. (2005). A theoretical integration of user satisfaction and technology acceptance. *Information Systems Research*, *16*(1), 85-102.

Wu, C.-G., Gerlach, J. H., & Young, C. E. (2007). An empirical analysis of open source software developers' motivations and continuance intentions. *Information & Management*, *44*(3), 253-262.

Xiao, L., & Dasgupta, S. (2002). *Measurement of user satisfaction with Web-based information systems: An empirical study.* Paper read at 8th Americas Conference on Information Systems.

Xu, J., Gao, Y., Christley, S., & Madey, G. (2005). *A topological analysis of the open source software development community.* Paper read at 38th Hawaii International Conference on System Sciences.

Ye, Y., & Kishida, K. (2003). Toward an understanding of the motivation of open source software developers. In *Proceedings of the 25th international conference on software engineering* (p. 419-429). Washington, DC: IEEE Computer Society. Paper read at 25th International Conference on Software Engineering.

Ye, Y., Nakakoji, K., Yamamoto, H., & Kishida, K. (2005). The co-evolution of systems and communities in free and open source

software development. In S. Koch (Ed.), *Free/open source software development* (p. 59-82). Hershey, PA: Idea Group.

Yoon, Y., & Guimaraes, T. (1995). Assessing expert systems impact on users' jobs. *Journal of Management Information Systems*, *12*(2), 225-249.

Yoon, Y., Guimaraes, T., & O'Neal, Q. (1995). Exploring factors associated with expert systems sucess. *MIS Quarterly*, *19*(1), 83-106.

Yun, G. W., & Trumbo, C. W. (2000). Comparative response to a survey executed by post, e-mail, and web form. *Journal of Computer-Mediated Communication*, *6*(1).

Zhang, P., & Dran, G. M. von. (2000). Satisfiers and dissatisfiers: A two-factor model for website design and evaluation. *Journal of the American Society for Information Science*, *51*(14), 1253-1268.

Zhao, L., & Deek, F. P. (2004). User collaboration in open source software development. *Electronic Markets*, *14*(2), 89-103.

Zviran, M., & Erlich, Z. (2003). Measuring IS user satisfaction: Review and implications. *Comunications of the Association for Information Systems*, *12*, 81-103.

Zviran, M., Glezer, C., & Avni, I. (2006). User satisfaction from commercial web sites: The effect of design and use. *Information & Management*, *43*(2), 157-178.

COLOPHON

This thesis was typeset with LaTeX 2ε using the free/libre Kp-Fonts package by Christophe Caignaert. This type face is available for LaTeX via CTAN as "kpfonts".

The typographic style was inspired by Robert Bringhurst's genius as presented in *The Elements of Typographic Style* (2002). It is available for LaTeX via CTAN as "`classicthesis`".

NOTE: The custom size of the textblock was calculated using the directions given by Mr. Bringhurst (pages 26–29 and 175/176).

To make your own calculations, use the following commands and look up the corresponding lengths in Bringhurst's book:

```
\settowidth{\abcd}{abcdefghijklmnopqrstuvwxyz}
\the\abcd % prints the value of the length
```

Please see the file `classicthesis.sty` for some precalculated values for Palatino and Minion.

Final Version as of July 21, 2011 at 18:09.