Participant Satisfaction with Open Source Software

by
Brenda Lynne Chawner

B.Sc. (University of Alberta) 1976
M.L.S. (University of Alberta) 1978

A research proposal for the degree of

Doctor of Philosophy

in

Information Systems

in the

School of Information Management

of

Victoria University of Wellington

Principal Supervisor:

Professor Gary Gorman

Co-supervisor:

Professor Sid Huff

July 2005

Participant Satisfaction with Open Source Software

Abstract

Open source software is increasingly seen as a viable alternative to conventional closed-source software. Developer/user satisfaction and involvement are common suggestions for measures of an open source project's success; however, as yet no substantive research has been done on this topic.The purpose of this two-phase, sequential mixed methods research project will be to develop and test theory about factors that influence participant satisfaction with open source software.

The central question of the proposed study is:

What factors influence participant satisfaction with an open source application software project?

The conceptual model for the proposed research will be based on McKeen, Guimaraes, and Wetherbe's (1994) model of factors that influence the relationship between participation in systems development and user satisfaction, modified and extended for an open source context. Previous research into the structure of open source projects has largely focused on identifying activities and roles that relate to code generation, rather than community building or user support. The proposed research is intended to extend the description of project participation to include community building and user support activities. Therefore, the first sub-question for the proposed research is:

What types of contributions do participants make to open source application projects?

In a typical open source project, community members have varying levels of participation, ranging from passive users to core developers. This research also proposes to see if there are differences in satisfaction between users who take on different roles within projects, and the second sub-question is therefore:

Do the factors that influence satisfaction with an open source application software project differ for different roles? If they do, in what way?

The first qualitative phase of the proposed research will use key informant interviews combined with observation of a purposive sample of projects to gain a better understanding of the type and frequency of different types of participation in the projects. This phase will also confirm the preliminary model of overall factors that influence participant satis-

faction with open source application software projects. The results of this phase will be used to develop a web-based questionnaire to gather quantitative data to test the model and associated propositions/hypotheses as phase 2.

The results of this research will have both practical and theoretical implications. They will help developers of open source application software identify opportunities to increase user satisfaction, and they will help users identify ways in which they might contribute to projects. They will also contribute to the growing body of theory about open source software development and communities, as well as the literature about user satisfaction with information systems.

Table of Contents

List of Figures

List of Tables

1 Introduction

Open source software is increasingly seen as a viable alternative to conventional closed-source software. A typical open source project starts when an individual (or small group) has a need for new software and writes some. In order to share this with others who have similar needs, the software is released under a licence that allows new users to see the source code and modify it to meet local needs or fix bugs. These new users of the software may then contribute their changes back to the community. This results in software that is said to evolve more quickly and meet user needs better than commercial closed-source packages. In addition, new users may be able to communicate directly with developers to solve problems or request enhancements. An unstated implication of these characteristics is that user satisfaction with open source software will be increased by this ability to be more involved with the software development process.

Crowston and others (2003) posted a question about measuring project success on Slashdot (a Web-based portal that has a high profile with the open source developer community). Developer/user satisfaction (43%) and involvement (41%) were the most common suggestions for measures of a project's success. Despite this indication that involvement and satisfaction are suitable measures of open source project success, as yet no substantive research has been done on this topic.

The purpose of the proposed two-phase, sequential mixed methods research project will be to develop and test theory about factors that influence participant satisfaction with open source application software.

The central question of the proposed study is:

What factors influence participant satisfaction with an open source application software project?

Prior research has established a number of models of user satisfaction with information systems. Most of these have been based on an unstated assumption that system development takes place in-house. The conceptual model for the proposed research will be based on McKeen, Guimaraes, and Wetherbe's (1994) conceptual model of factors that influence the relationship between participation in system development and user satisfaction, modified and extended to suit an open source context. The McKeen, Guimaraes, and Wetherbe model has 6 components:

- the dependent variable is *user satisfaction*;
- the independent variables are: *user influence*, *user–developer communication*, and *user participation;*
- moderating variables are *task complexity* and *system complexity.*

Previous research into the structure of open source projects has largely focused on identifying activities and roles that relate to code generation, rather than community building or user support. This research is intended to extend this picture of project participation to include community building and user support activities. Therefore, the first sub-question for the proposed research is:

What types of contributions do participants make to open source application software projects?

In a typical open source project, community members have varying levels of participation, ranging from passive users to core developers. This research also proposes to see if there are differences in satisfaction between users who take on different roles within projects, and the second sub-question is therefore:

Do the factors that influence satisfaction with an open source application software project differ for different roles? If they do, in what way?

The first qualitative phase of the proposed research will use key informant interviews combined with observation of selected projects to gain a better understanding of the type and frequency of different types of participation in the projects. This phase will also confirm the preliminary model of overall factors that influence participant satisfaction with open source application software projects. The results of this phase will be used to develop a web-based questionnaire to gather quantitative data to test the model and resulting propositions/hypotheses. A non-library open source community will be used for a pilot test of the questionnaire.

The results of this research will have both practical and theoretical implications. It will help developers of library and information management open source software identify opportunities to increase user satisfaction, and it will help users identify ways in which they might contribute to projects. It will also contribute to the growing body of theory about open source software development and communities, as well as the literature about satisfaction with information systems.

1.1 A note on terminology

The term ‘open source’, now widely used, is relatively new; it 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 established in 1985 by Richard Stallman (DiBona, Ockman and Stone 1999: 3; Feller and Fitzgerald 2002: 38). They established the Open Source Initiative to promote the Open Source Definition (OSD) (Open Source Initiative 2002) 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). 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 paper will treat them as synonyms, and will use a hybrid term, Free/Open Source Software (F/OSS), when appropriate. See Section 2.2 *Meaning of ‘free’ and ‘open source’* on page 6 for a more extensive discussion of these terms.

1.2 Definitions

The following terms have been defined for this proposal, and have been written for an F/OSS context.

1.2.1 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.

1.2.2 Developer

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

1.2.3 Free/open source software

Software released under a free/open source licence, making source code available to users (and potential users) free of charge, and allowing them to modify it to suit their needs, as well as redistribute to others.

1.2.4 Initiator

The person(s) or organisation(s) who start a project, as opposed to other participants who join when the project is under way.

1.2.5 Participation

Any type of contribution made to an F/OSS project, whether as source code, documentation, community support, or other resources.

1.2.6 Satisfaction

The *OED online* gives seven definitions for this term, in a range of contexts. For the context of this research, the most appropriate is “The satisfying of a need or desire as it affects or motivates behaviour”. In this context, ‘satisfying’ implies “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*, accessed 15 July 2005).

1.2.7 Source code

Statements in a programming language that comprise a computer program or software package.

1.2.8 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.

2 Literature review

2.1 Introduction

This section begins by clarifying the meaning of ‘free’ and ‘open source’ in the context of software licensing and distribution, followed by an overview of software licensing models. It then discusses F/OSS life cycles, followed by a brief history of F/OSS distribution. The next section reviews literature about F/OSS practices and communities, beginning with a discussion of issues associated with gathering data about F/OSS projects. The primary focus is on participant roles and activities. A section on user satisfaction models follows, and the literature review concludes with a preliminary model of factors that influence participant satisfaction with F/OSS.

Material for this literature review was collected using a range of techniques. The first step involved searching a range of databases including *ABI/Inform*, *Library Literature and Information Science Fulltext*, and the ACM and IEEE digital libraries for material on F/OSS projects and user satisfaction. Citations in recent review articles were used to locate older, classic articles, and *Web of Knowledge* was used to locate newer material that cited older, relevant papers. Significant conference web sites, such as the International Conference on Information Systems, were searched for papers on F/OSS projects and user satisfaction. F/OSS portals such as the MIT Free/Open Source Research Community portal (http://opensource.mit.edu/) and the opensource.ucc.ie: resources for researchers (http://opensource.ucc.ie/) were checked regularly. Proceedings from F/OSS conferences and workshops, particularly those associated with the annual International Conference on Software Engineering conference, were checked for relevant papers.

The conceptual model of an open source project used in this proposal is shown in Figure 2-1 below. It shows an F/OSS project as the intersection of three components: software, a development process, and a community of developers and users.

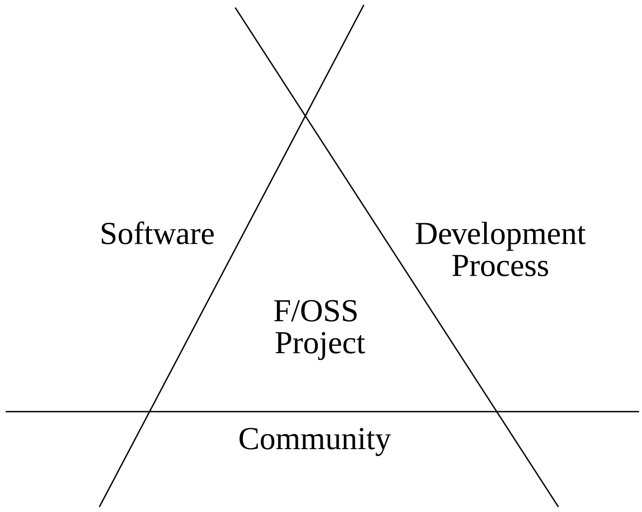


Figure 2-1. Conceptual model of an F/OSS project

2.2 Meaning of 'free' and 'open source'

If asked, most computer-literate people would probably say that free/open source software includes the original code for the program, whatever language it is written in (C, C++, Perl, Php, etc.) and they might also say that the software may be modified for local use and then subsequently redistributed. 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.

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 identifies four aspects of freedom:

1. the freedom to run the program, for any purpose (freedom 0)
2. the freedom to study how the program works, and adapt it to local needs (freedom 1)
3. the freedom to redistribute copies so others can benefit from the software (freedom 2)
4. 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 are from the user's perspective, and only freedom 4 mentions a software community. In order for software to qualify as 'free' under freedoms 1 and 3, a user must have access to the source code.

The FSF maintains a list of licences that comply with the Free Software Definition at http://www.fsf.org/licenses/licenses.html.

2.2.2 The Open Source Definition

The Open Source Initiative maintains the Open Source definition (OSD), version 1.9 at 1 October 2002, which has 9 clauses:

1. Software must be able to be freely distributed, without requiring a royalty or fee for sale.
2. The source code for the program must be available, and, if not included in a distribution, must be easily available (for example, downloadable from a web page) in a form which allows a programmer to modify the program.
3. Modifications and derived works must be allowed, and these must be able to be redistributed under the same terms as the original software.
4. The integrity of the original source code must be able to maintained, either by requiring modifications to be distributed as ‘patch files’, or by requiring modified versions to have a different name or version number.
5. There must be no discrimination against persons or groups.
6. There must be no discrimination against any field of endeavour. This clause is particularly important because it allows commercial use of the software.
7. The license must apply to anyone receiving a copy of the program, without requiring them to agree to another license.
8. The license must not be specific to a particular product or distribution.
9. The license must not apply to other software distributed along with the licensed program(s).

2.2.3 Discussion

Though these two definitions represent very different philosophical positions, with the OSI being pragmatic and the FSF uncompromising, in practice they overlap.

As Feller and Fitzgerald (2002: 12-13) note, while the OSI certification mark applies to software not licences, in practice the OSI certifies licences and not individual products. In January 2005, the OSI listed 57 licences that comply with the OSD (OSI 2005), while the FSF listed 60 ‘free’ licences (FSF 2005). Many of these of these, like the GNU General Public License (GPL) and the GNU Library or Lesser General Public License (LGPL) appear on both lists. The GPL is by far the most popular F/OSS licence, used by 40,850 projects listed on SourceForge (a significant F/OSS project repository) on 18 January

2005. The next most popular licences for SourceForge are the LGPL, with 6,563 projects, and the BSD License, used by 4,242 projects.

One particularly important point about both definitions is that they are about licensing and distributing software, and say nothing about the methods and processes used to develop it. There is considerable variation in the development processes used by different F/OSS projects, in particular the degree of ‘openness’ to new developers.

2.3 Types of software licences

The F/OSS model is one of many models for licensing and distributing software. The following table lists characteristics of a range of common software licensing options, to emphasise the unique features of the F/OSS model. The term ‘viral’ is used to describe an F/OSS licence that requires any software incorporating F/OSS source code to be released under a similar licence. ‘Crippleware’ is a term for software usually provided for trials, but with restrictions on number of users, number of uses, or size of implementation, or with a limited set of features. These licensing models have been selected as examples, and other licensing models, such as abandonware, adware, donationware, and postcardware are listed in Wikipedia (http://en.wikipedia.org/). Semi-open licences allow the software and source code to be distributed to certain types of organisations (for example, educational and non-profit institutions). ‘Shared source’ is a term used by some commercial software vendors (it originated with Microsoft), to describe their practice of allowing specific groups to have access to the source code for their products.

Table 2-1. Characteristics of software licences

Type	Source code available	Redistribution	Cost	Copyright	Restricted	Viral
Commercial	No	No	Yes	Yes	Yes	No
Shared source	Yes	No	?	Yes	Yes	No
Shareware	No	No	No	Yes	Yes	No
Crippleware	No	No	No	Yes	No	No
Freeware	No	Yes	No	Yes	No	No
Open source (non-viral)	Yes	Yes	No	Yes	No	No
Open source (viral)	Yes	Yes	No	Yes	No	Yes

Table 2-1. Characteristics of software licences (Continued)

Type	Source code available	Redistribution	Cost	Copyright	Restricted	Viral
Semi-open	Yes	Yes	No	Yes	Yes	No
Public domain	Yes	Yes	No	No	No	No

2.4 F/OSS project lifecycle

Schweik and Semenov (2003) identify 3 major stages in F/OSS projects:

1. Project initiation
2. Going 'open'
3. Project growth, stability, or decline

This linear approach to the process of releasing F/OSS software is useful because it identifies key points at which the structure of F/OSS project communities may change. In the proposed research project, the individual or group involved in stage 1 will be termed the project's 'initiator'.

The reasons for project initiation have been described in a number of papers. Raymond (1999CatB) says that a project must "scratch a developer's personal itch", meaning that it must be something that the developer is interested in. Bonaccorsi and Rossi say that a project "starts because an individual (or an organization) is facing a problem that needs a particular kind of software in order to be solved" (2003: 1246). Both of these imply a personal or organisational need for new software.

Schweik and Semenov (2003) extrapolated the following list of reasons for project initiation from research into F/OSS participant motivations:

Table 2-2. Reasons for F/OSS project initiation

Category	Motivation
Technological	to meet a personal need
	to work with the leading edge of technology
	to address a software crisis
	to have intellectual stimulation
Socio-Political	to satisfy an intrinsic motivation to do the work
	to take on a threat, rival, or monopoly
Economic	to improve programming skills
	to take advantage of low cost and the opportunity to participate with nothing to lose

This range of underlying reasons for project initiation suggests that there is likely to be a range of approaches to ‘going open’ for F/OSS project structures, depending on the underlying reason for the project to be started. The next section of this proposal gives a brief history of F/OSS software development, and with a particular focus on identifying reasons for ‘going open’.

2.5 A brief history of F/OSS software

Sharing source code for software is not a new concept, and developers make their code available for a variety of reasons. Fifty years ago, when computers were programmed in assembler, 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 2003). Leonard quotes 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) reports 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 from similar practices.

Since 1960, the Association for Computing Machinery (ACM) has published algorithms (known as CALGO, Collected Algorithms of ACM), in a variety of formats and publications (Hopkins 2002). Hopkins notes 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) suggests 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 these algorithms were published with a statement saying that “the reproduction of algorithms is explicitly permitted …without any charge” (Herbold 1960) and, since this meets both the FSF and OSI’s conditions of freedom and openness, distribution of these algorithms represents an early example of F/OSS. 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” (ACM, 1998), making CALGO currently semi-, rather than fully, open.

With the rise of commercial computing in the 1960s and 1970s, most software became proprietary, and sharing source code was largely restricted to computer science academics and students. Some software, such as Donald Knuth’s $\mathrm{T_EX}$, continued to be released along with source code. Knuth has provided extensive documentation of his experiences in publishing the $\mathrm{T_EX}$ source code (Knuth 1992a; 1992b); though he does not make an explicit statement about his motives in releasing the source code, he does say that “the real test begins as people with many different viewpoints undertake their own experiments” (Knuth 1992a: 262). This suggests his goal was quality—both in design and freedom from bugs.

Throughout the 1970s, publishing source code for educational purposes continued. Bentley’s *Programming Pearls* (1986, revised1989) and *More Programming Pearls* (1988), compiled from his columns in *Communications of the ACM* are examples of this, and 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 specialists* (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 F/OSS. 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 2003). 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. Unlike earlier F/OSS projects, the motivation for this release of source code was legal compliance with the government decree. 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 for 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 latest Macintosh operating system, Mac OS X.

In 1985, Richard Stallman 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 projects, including GCC (Gnu C Compiler,) and the multipurpose Emacs editor. Stallman's background and motivation for establishing the FSF are well documented (Bretthauer, 2002; Moody, 2002); to him, free software is a political and ethical statement. The most popular F/OSS licence, the GPL, was developed by the FSF, and it has been in use since 1999.

In 1991 Linus Torvalds, then an undergraduate student at the University of Finland, was the initiator of what is commonly regarded as the F/OSS 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' initial motivation for the project was personal—he wanted to run a Unix-like operating system on his new 386 computer, and started writing an operating system kernel. He gives two reasons for releasing it as free software: first, he disliked shareware, feeling guilty for not paying the fees, and second, he disliked the nominal fees charged for Minix, a Unix compatible operating system (Moody 2002: 44-45). Torvalds felt that making it freely available through FTP was a better option. Neither of these comments explains why Torvalds made the source code available, though his postings announcing the early versions give an indication of his thinking. He says: "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 FSF's GPL.

A further reason for releasing code is given by Fink (2003, 171)—the developer of Jump, a Java translator for the Palm platform, decided to discontinue the project, and released the source code so that other people could continue to work on it. After four years, another developer picked up the code and restarted the project. Commercial software vendors sometimes do this as well when they discontinue a product. In 2002, OCLC discontinued work on SiteSearch, a proprietary database searching package, and made the source code available under a semi-open licence allowing non-commerical use (Napier-Tyree 2002). OCLC hosted a user group meeting to coordinate development in May 2002. In response to a request from the developers, OCLC released the final version of the software under a fully open licence (the OCLC Research Public License 2.0 2002, available from http://www.oclc.org/research/software/license/default.htm) in early 2004, and the project (renamed OpenSiteSearch) is now being continued by an independent group of developers using SourceForge as the main project repository.

There is at least one more avenue to F/OSS distribution; some vendors release a limited version of their software under an F/OSS licence, but offer an extended version under a commercial licence. OpenOffice.org and StarOffice are the best known of these. Sun Microsystems founded and sponsored the OpenOffice.org project, and contributes to its development. The base OpenOffice.org package is available under the LGPL, while Sun Microsystems sells StarOffice, which incorporates proprietary code. Although the primary reasons for releasing the software under an F/OSS licence aren't stated explicitly, one possibility is that Sun wanted to gain from the ongoing development by the wider community (Wikipedia,"Openoffice.org", 2005 January 25, http://en.wikipedia.org/wiki/OpenOffice.org).

In summary, F/OSS projects are likely to have a range of motivations for going open, in addition to the philosophical and political reasons given the Free Software Foundation:

- to save coding and machine time;
- to return the benefit of previous access to source code;
- for educational use;
- to validate research results;
- to improve quality—both in design and freedom from bugs;
- to comply with legal requirements;
- to give others enjoyment and the opportunity to modify the software;
- to give others the opportunity to continue a project; and
- to benefit from a wider development community.

This range of reasons for releasing source code suggests that the F/OSS landscape is complex, and that there is more than one type of F/OSS project. The reason for a project's initiation and its reason for ‘going open’ are both factors that may influence the nature of community members’ participation and their satisfaction.

Several of the preceding examples show that the decision to go open can be made at a relatively late stage (OCLC’s SiteSearch and Sun Microsystem’s OpenOffice in particular). This supports the separation of a ‘going open’ stage from project initiation in the F/OSS project life cycle, though it is important to recognise that some projects may be ‘born open’, i.e. intended to be open before any code has been written.

2.6 Research into F/OSS practices and communities

F/OSS software development and the communities associated with F/OSS projects have attracted considerable attention from researchers in a range of fields in the last 5 years. Economists are interested in the ‘free’ aspect and the economic models underlying F/OSS (see Lerner and Tirole 2000, for one example), while the legal aspects of giving code away but retaining copyright is of interest to intellectual property specialists (O’Mahony 2003 is one example). From a management perspective, F/OSS communities are seen as examples of effective virtual organisations; the ways in which they coordinate work (Crowston and resolve conflict, and how these practices can be applied to other contexts have been studied. Computer scientists and information systems researchers have investigated F/OSS development methods (see Scacchi 2003 and Scacchi 2004 for recent examples).

Despite this extensive interest, the research literature on F/OSS development, projects, and processes is in what Kuhn called the developmental phase of a new paradigm (Kuhn 1970: 76). There is as yet no widely accepted conceptual or theoretical framework to describe, interpret, and compare F/OSS project structures and processes. A majority of the literature is in the form of conference papers, with most of the refereed journal articles published in special issues of journals (e.g. *Research Policy* 32 (6) 2003, which contained 7 research-based articles and *IEEE Software* 21(1) 2004, which contained 10 articles, some research-based, others opinion-based). Books about F/OSS are primarily ‘how-to’ guides for people interested in setting up an F/OSS project or using the resulting software (for example Pavlicek 2000; Sandred 2001; Golden 2005), or accounts of specific projects or people (Moody 2002; Torvalds and Diamond 2002; Williams 2002). There are also a number of self-published pre-prints on F/OSS portals, in particular the MIT Free/Open Source Research Community (http://opensource.mit.edu/online_papers.php). In 2002, the

FLOSS project, funded by the European Commission, released a comprehensive report on its study of people, companies, and public organisations developing and using F/OSS. It had a focus on Europe. The most common methodology used in studying F/OSS practices and communities is the descriptive case study, and the second most common is a web-based survey.

2.6.1 Issues associated with studying F/OSS projects and communities

Most F/OSS projects provide a wealth of freely available project information for interested users (and researchers). These resources usually include the source code itself, often with a credits file that shows who contributed what code and when it was contributed, plus mailing list archives, bug databases, and other project-related documentation. This means that it is relatively easy to gather data for 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 a recent example.

Comparing results across different projects, however, is more difficult. There is no single authoritative source of comparative information on F/OSS projects. SourceForge (http://sourceforge.net/index.php) is the largest repository of F/OSS projects, with over 100,000 projects listed on 15 July 2005. While this number is impressive, studies of project activity have shown that only a very small number of projects are active and a much larger number relatively static, resulting in a Pareto or power law distribution (Hunt and Johnson 2002). Although considerable raw data about F/OSS projects are available from SourceForge and FreshMeat (http://freshmeat.net), the quality of this data is uneven. For example, SourceForge provides a count of the number of times a package has been downloaded, but the interpretation of the resulting number is difficult. It is unlikely that everyone who has downloaded a package has installed it, as the phenomenon of downloading software on a trial basis to see what it can do is common in the F/OSS world. Howison and Crowston (2004: 11) suggest that packages undergoing frequent change (whether to add new functionality or to fix bugs) would be downloaded frequently, as users keep their version of the software up to date. This means that the meaning of a high number of downloads can be difficult to interpret.

Other issues with SourceForge data include developers classifying their projects by a range of attributes. Classification options include:

- development status (options are 1 - Planning; 2 - Pre-Alpha; 3 - Alpha; 4 - Beta; 5 - Production/Stable; 6 - Mature; and 7 - Inactive);
- intended audience (top level industry sectors are limited to Aerospace; Customer Service; Education; Financial and Insurance Industry; Government; Healthcare Industry; Information Technology; Legal Industry; Manufacturing; Non-Profit Organizations; Religion; Science/Research; and Telecommunications Industry); and
- Topic (top level categories are Communications; Database; Desktop Environment; Education; Formats and Protocols; Games/Entertainment; Internet; Multimedia; Office/Business; Other/Nonlisted Topic; Printing; Religion and Philosophy; Scientific/Engineering; Security; Sociology; Software Development; System; Terminals; and Text Editors).

This self-classification means that the classification data are not necessarily reliable, as different developers might interpret the classifications differently. To illustrate the variability of this data, SourceForge entries for 5 different wiki engines were checked on 31 January 2005; they were categorised as:

1. Dynamic content, Site management;
2. Front ends, Message boards, Site management, Documentation;
3. Dynamic content, Office/Business; and
4. Communication, WWW/HTTP.

One, PmWiki, was not yet classified. All of these engines do more or less the same thing: let an authorised group of users create and maintain a web site using HTML forms and text-based markup, but the developers who created the different entries clearly had different opinions about which topics were most appropriate.

Some of the data from SourceForge are useful as a measure of project-related activity, such as mailing list archives. However, many F/OSS projects use external resources to supplement or complement SourceForge project hosting, or are hosted on a site provided by the main developer or other interested party. The PmWiki project uses SourceForge as a back up software repository, 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/. Stürmer (2005) studied a total or 8 F/OSS projects, 5 of which were popular web content management systems, with 2 web application frameworks and 1 browser editor. While most of these projects had begun on SourceForge, a majority had migrated to other collaboration platforms as they became more popular (Stürmer 2005: 72). One project, Magnolia, used its own collaboration platform for all development activities, and used

SourceForge as a download site only. This means that research that treats SourceForge as a reliable repository of comprehensive project-related data is misleading.

Few frameworks exist to support meaningful comparisons of F/OSSprojects. Capiluppi, Lago, and Morisio (2003) used 14 project attributes to compare a random sample of 406 projects selected from the FreshMeat project repository:

- age;
- application domain (using developer-assigned values);
- programming language;
- code size;
- number of developers;
- number of users (based on a surrogate, the number of the subscribers to the project in the repository);
- modularity level (based on a surrogate, the number of directories for the source code, with 3 possible values);
- modularity (number of directories, plus average size of module, calculated as code size/number of modules);
- documentation level (3 possible values: comments in source code, README, or user manual);
- popularity (as calculated by the repository, based on number of visits to project home page, number of visits to project pages, and number of subscribers);
- status (the developer-assigned value);
- vitality (as calculated by the repository, based on number of announcements in a specified time period, age of project in days, and total number of announcements;
- relative version number; and
- version release date.

The major findings were that size, number of developers, and number of authors (this is not defined in the paper, but appears to be a combination of developers and other contributors) grow over time, and that the 97% of the projects sampled were relatively static. The authors' reliance on developer-assigned values and surrogate values for many of these attributes suggest that their results have limited reliability.

German and Mockus (2003) describe a software tool (itself an open source project), that gathers data from mailing list archives, CVS logs, source code versions, and bug tracking systems to trace the evolution of F/OSS code, and allow different projects to be compared.

Gacek and Arief (2004) have proposed the use of 9 project characteristics to compare projects: project starting point (similar to the preceding reasons for opening source code); motivation and underlying business model; community structure and culture; software development support; licensing; and size (community and code base).

Crowston and others (2004) have proposed a 'portfolio' of 4 measures of a project's success: development team size, bug-fixing hazard ratios, activity level, and number of downloads (the latter two taken directly from SourceForge). The use of the word 'success' is a bit misleading, as these measures are more concerned with activity, and success implies an interpretation from a particular perspective. In applying these 4 measures to 140 projects, Crowston and others (2004) found that there were strong correlations between community size and number of bug reports; activity level and number of downloads were also strongly correlated, but this could be explained in part because the formula used to calculate the activity level included the number of downloads. As noted earlier, using the number of downloads as a surrogate for the number of users is potentially misleading, because the proportion of users that result from the downloads is not known, and is likely to vary from project to project.

2.6.2 Community structure

This section discusses and compares three different perspectives on F/OSS community structures, based on case studies of different projects.

***Lakhani, Wolf, and Bates (2002)*.** The high-level stakeholder diagram below is from a presentation version of a report on The Boston Consulting Group Hacker Survey released on 11 January 2002. It shows different stakeholder groups who are involved in open source development, and identifies typical interactions between the groups.

)

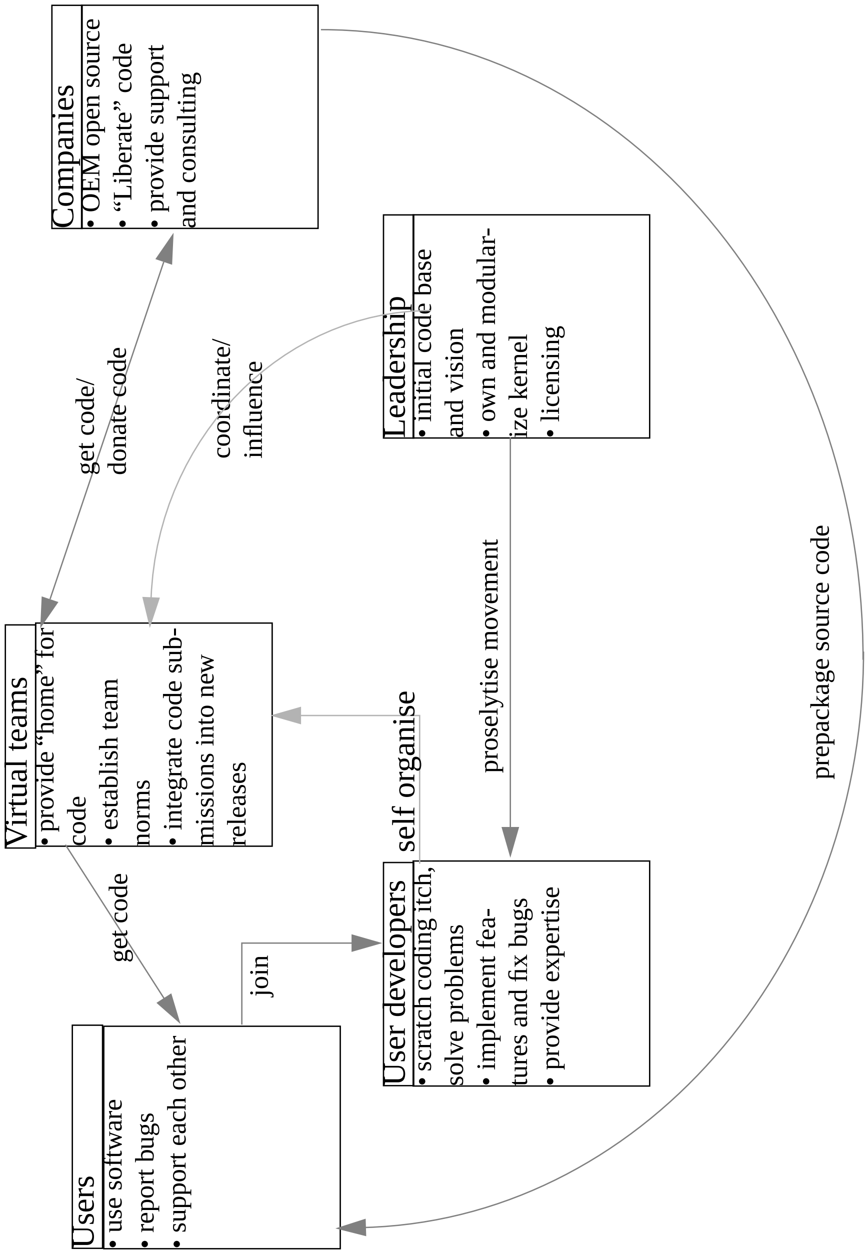


Figure 2-2. Lakhani, Wolf, and Bates stakeholder diagram

The diagram also represents different roles in the distributed team undertaking an F/OSS project, such as governance (indicated by a 'leadership' category), and code generation (the 'user developers'). One interesting feature of this model is that it shows support for users coming from either other users or from companies—implying that developers do not support non-developer users directly. This model of user support is described in the Lakhani and von Hippel study of the way in which users of the Apache web server software are supported (2003). In it they say that support for Apache users is “operated by and for users themselves”, with the developers providing no official support (2003: 925). It is unlikely that this can be generalised to other F/OSS projects, since projects in other contexts might have adopted a different model of user support. Core developers from Greenstone Digital Library project at the University of Waikato often respond to user requests on the mailing list, for example.

Gacek and Arief (2004). This model was originally developed in 2002 as part of a UK-based research project into the dependability of open source software (Gacek and others 2002). It shows how control and power are managed—both increase from top to bottom and from left to right. This model includes ‘passive’ users, unlike the previous example, and it is also the only model to show the types of changes that may occur over time, with transitions shown from passive user to active one, and from active user to developer, etc.

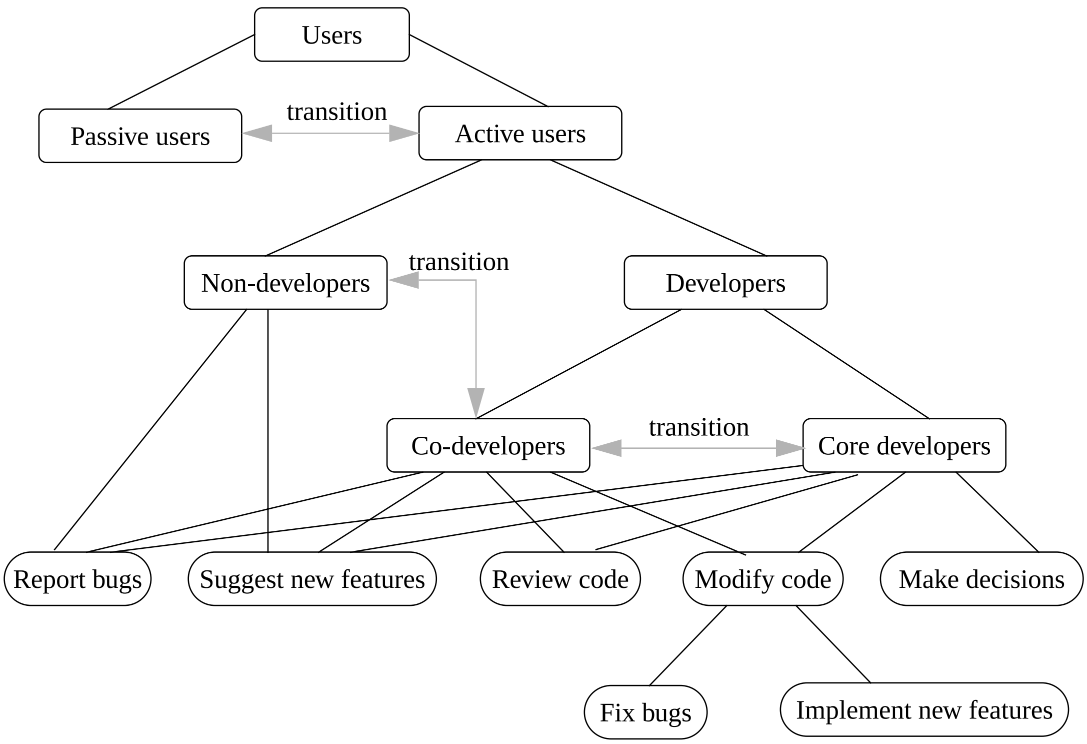


Figure 2-3. Gacek and others role/activity model

One assumption Gacek and Arief built into this model is that *all* developers are also users of the software. This generalisation may not always apply to domain-specific application software, which may involve users paying someone to develop specific functionality. One example of this is the Koha project to develop an F/OSS library management system. The Horowhenua Library Trust (HLT) commissioned developers at Katipo Communications Limited to write the first version of Koha (Chawner 2002), and Katipo staff, who are mainly web developers, do not use Koha in their day to day work.

Ye and others (2004). This model was initially presented at the Workshop on Principles of Software Evolution in May 2002 (Nakakoji and others 2002), and subsequently revised for publication in 2004. It is based on a case study of four F/OSS projects within Software Research Associates, a Japanese software development company. The projects were: GNU software, Linux Support, PostgresSQL, and Jun (a Smalltalk and Java library for 3D objects and multimedia data handling). Ye and others (2004) identify eight possible roles for F/OSS community members, but notes that some communities do not demonstrate all eight roles. They say that most community members start as passive users, and progress through the layers to become core members if they have the ability and interest.

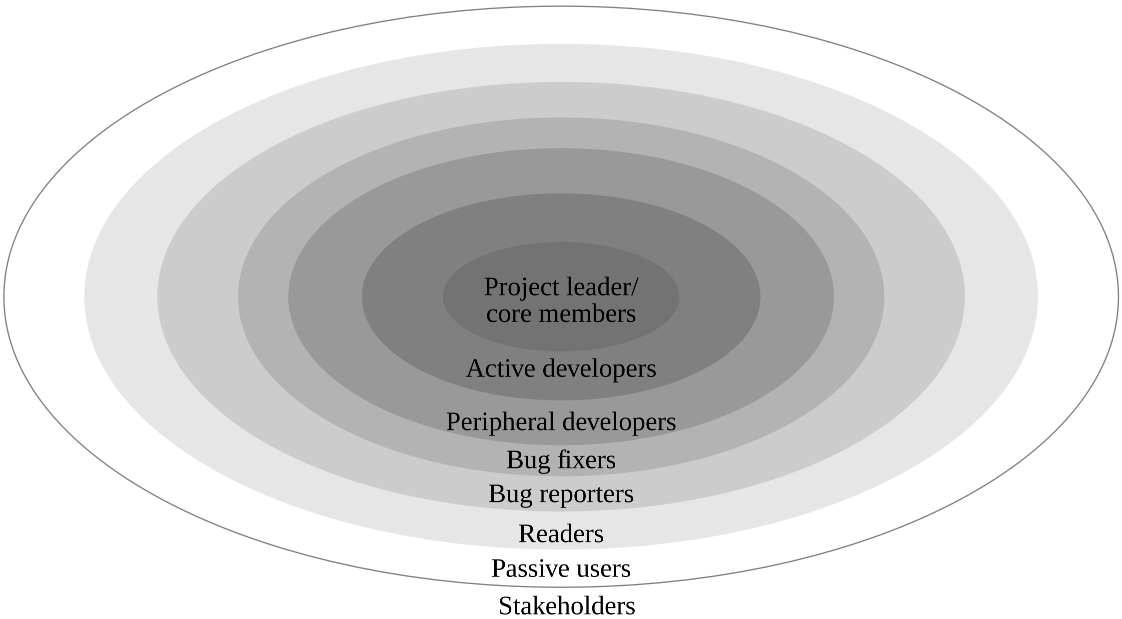


Figure 2-4. Ye and others onion model (2004)

This onion-style perspective has been used or implied by other F/OSS researchers. Crowtson and others (2005) include only 4 layers in their version: core developers, co-developers, active users, and passive users. They show initiator and release coordinator roles inside the core developer layer. Mockus, Fielding, and Herbsleb (2002) studied the Apache and Mozilla projects, finding that the core team size varied from 10-15 people.

Other writers suggest a range of different roles, usually refining or expanding the developer or core member roles. From their study of the Mozilla and FreeBSD projects, Holck and Jørgensen (2004) identify reviewer (responsible for manually checking code before it is committed to the main repository) and committer (authority to add or change code in the repository) roles, as well as the role of module owners. They note that these roles are earned, and that people must first show their competence before being granted commit privileges. Scacchi suggests that F/OSS roles include "core developer, module owner, code contributor, code repository administrator, reviewer, or end user" (2004: 60), while von Krogh and von Hippel (2003: 1152) identify the role of 'owner' as "those who have the exclusive right, recognized by the community at large, to redistribute modified versions". The Koha project has a 'kaitiaki', or project manager role (Chawner 2003). Unlike the Mozilla and FreeBSD projects, this does not necessarily require experience as a developer.

Behlendorf (1999) is one of the few people to take a broader perspective on roles needed to support a growing community. He suggests that a project will need someone to provide infrastructure support, a 'code captain' (similar to the reviewer/committer roles described above), someone responsible for bug database maintenance, a documentation/web site maintainer, and an 'evangelist' whose focus is primarily external and seeks to build momentum for the project.

From this we can conclude that there is an extensive range of types of contributions to an F/OSS project, going far beyond the 'code contribution' implied in the simpler models. Additional activity categories include project management and coordination tasks, promotion of the software, and providing resources to support the project community.

2.6.3 Participant motivation

The reasons people participate in F/OSS development has received a great deal of attention from researchers, perhaps because of the perceived paradox between people apparently working on something 'for free' and then giving away the results of their work. This perception is not well supported by survey results, which show that between 30% and 54% of F/OSS project contributors are paid at least in part for their work (International Institute of Infonomics and Berlecon Research GmbH 2002; Lakhani, Wolf, and Bates 2002; Hertel, Niedner, and Hermann 2003).

Few studies of F/OSS participant motivation have used an underlying theoretical model of motivation. They have generally relied on surveys giving a list of possible reasons for participating, and reported the frequency of different motivations. The main reasons people work on F/OSS projects are intellectual gratification (Bonaccorsi and Rossi 2003), learning and enjoyment (Lakhani, Wolf, and Bates 2002a, 2002b), and to gain reputation (Lerner and Tirole 2000). Marcus, Manville and Agres (2000) suggest that there are economic motivations as well, primarily the desire for better software. The FLOSS survey of 2784 developers found that the most common reasons for joining an F/OSS community were learning and developing new skills and sharing knowledge and skills (International Institute of Infonomics and Berlecon Research GmbH 2002: Part 4, 45)

2.6.4 Community governance

Governance is a multi-faceted aspect of F/OSS communities. Marcus, Manville, and Agres say that governance of virtual communities includes:

- managing membership;
- establishing rules and institutions
- monitoring members' behaviour;
- control mechanisms; and
- shared culture, values, and norms for behaviour (2000: 14).

F/OSS projects approach governance in different ways, but as a rule there is a small group of people who take on a leadership role (Bonaccorsi and Rossi 2003: 1247). Formal roles are not necessarily established, but larger projects may have a management group that is analogous to a company board of directors (Holck and Jørgensen 2004: 12). Procedures for joining such a group vary. Mozilla.org staff members are invited to join the management group, while FreeBSD's Core Team (9 members) are elected every second year by active developers (Holck and Jørgensen 2004: 12).

The Koha project has a kaitiaki (the Maori word for 'guardian' or 'trustee') who is responsible for keeping the project on track—unlike the Mozilla and FreeBSD projects, this person is not necessarily an active developer, but is viewed more as a project manager (Chawner 2003). This role is usually taken on by a volunteer, with the support of Koha community members.

2.6.5 Community practices

There is a growing body of literature about F/OSS community practices, with a particular focus on the way the distributed teams work effectively. Gallivan (2001) found that social control was an important mechanism. von Krogh, Spaeth, and Lakhani (2003) undertook a detailed case study of the Freenet project using grounded theory techniques to determine how people join the developer group for the project and how they initially contribute code. The Freenet project is relatively new (begun in 2000), and originated in 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 an 30 people with commit access to the project's CVS repository. Their results showed that the Freenet community had an implicit 'joining script' newcomers were expected to follow before being granted 'developer' access to the code repository. The most important aspects of the joining script were participating in technical discussions and offering new code, usually to fix a bug in the current version of the software. The results of this study are limited by its focus on a single F/OSS project.

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 (n=1288, 214 usable responses) or answers (n=421, 122 usable responses) between 1 October 1999 and 15 February 2000. The overall response rate to the survey was 19.6%. They found that people who answer questions learn from other people's questions and answers, which is their main reason for reading postings to the newsgroup. Their most significant finding was that people tended to provide 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. One point Lakhani and von Hippel made, but did not emphasise, is that there are many other sources for help with Apache, in addition to the newsgroup, and so the picture they present of support for Apache users is incomplete.

Researchers investigating F/OSS are beginning to recognise that the degree of 'openness' varies between projects. Ye and others identify two dimensions of openness: product and process (2004: 76-77). They define product openness in terms of the release practices of the development community; 'open release' means that only formal releases are available to the user community, while 'open development' means that interim releases are available as well. Process openness relates to the extent to which members of the broader community are able to participate in decisions about the software's development path. Ye and others identify three possible values for this: closed, transparent, and open. With a closed process, only the 'inner circle' of the core developers determines the development priorities, while with a transparent one, the discussions are accessible to all community members, but decisions are made by the core group. An open process allows any community member to participate fully in decisions about future development. Coleman and Hill (2004) describe that 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.

2.7 User satisfaction

User satisfaction has been used as a surrogate measure of information system success for many years. In a review of 59 articles about user satisfaction with information systems, Zviran and Erlich (2003) say that satisfaction is "the most prevalent measure of IS success due to its applicability and ease of use" (p. 83). Ives, Olson and Baroudi (1983) defined satisfaction as "the extent to which users believe the information system available to them meets their information requirements". Melone (1990) noted that satisfaction is an attitude involving evaluation, meaning that it can be evaluated as favourable/unfavourable, positive/negative, or pro/con. Igbaria and Tan (1997) said that satisfaction is based on "attitudes toward a variety of factors that are related to IS services and products". In a review of library-related research on user satisfaction with library services and products, Applegate comments that satisfaction is both "intellectual and emotional" (1997: 201), implying that it involves both affect (emotion) and behaviour.

Galetts and Lederer (1989) presented some cautions about studying user satisfaction. It cannot be measured directly, but instead is usually inferred from survey data. Typical questions about user satisfaction use Likert-scales to gather ordinal data about user perceptions. Zviran and Erlich (2003) identified 6 instruments that have been developed to measure user satisfaction, starting with Bailey and Pearson (1983) and ending with Etezadi-Amoli and Farhoomand (1996). The Ives, Olson, and Baroudi (1983) short-form questionnaire with 13 items was the most widely used instrument in the literature they reviewed.

Applegate (1997) identifies two approaches to studying satisfaction, terming them performance-only and disconfirmation. A performance-only approach generally identifies a range of factors that contribute to overall satisfaction, while a disconfirmation approach looks at the gap between expectation and experience/performance.

The next sections present 7 approaches to studying factors that influence user satisfaction, in chronological order. Five are performance only models, 1 is a framework to categorise studies of satisfaction, and 1 uses a disconfirmation approach.

2.7.1 Ives, Olson, and Baroudi model (1986)

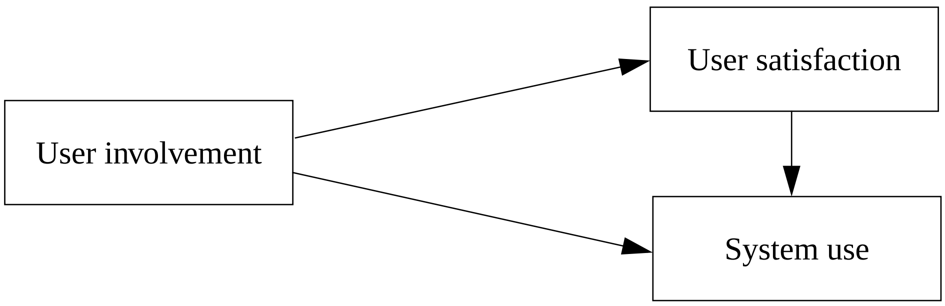


Figure 2-5. Ives, Olson, and Baroudi model (1986)

Ives, Olson, and Baroudi surveyed 800 production managers of U.S. manufacturing organisations, with a response rate of 25%. They used Bailey and Pearson's (1983) 39-item information satisfaction instrument, and developed a 30-item list of activities reflecting user involvement in various stages of information systems development. A similar list of system usage activities was developed for the research. Both lists were reviewed by experts before being finalised for the survey. The results showed that user involvement had a direct effect on user satisfaction, and had both direct and indirect effects on system use.

2.7.2 McKeen, Guimaraes, and Wetherbe (1994)

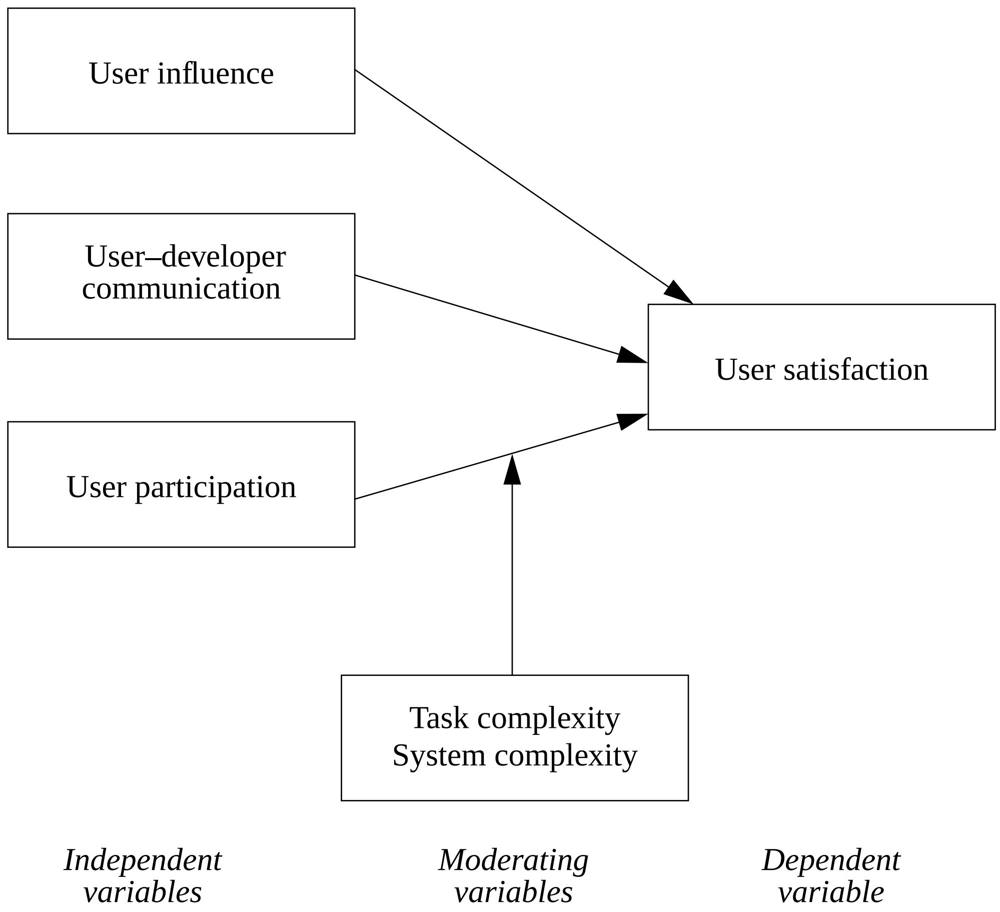


Figure 2-6. McKeen, Guimaraes, and Wetherbe (1994)

McKeen, Guimaraes, and Wetherbe (1994) undertook a survey of 151 mainframe-based systems development projects in 8 organisations to investigate factors that affect the relationship between user participation and user satisfaction. Their intention was to develop a parsimonious model, rather than to reflect all of the possible variables. Task complexity reflects uncertainty about user requirements, and system complexity reflects uncertainty about the underlying technology and development methodology. User influence is the extent to which user participation affects the final design of the system. User–developer communication is a measure of the quality of the relationship between the users and the developers, and effective communication directly affects user satisfaction. McKeen, Guimaraes, and Wetherbe used or modified existing instruments for all of the factors: Ives, Olson, and Baroudi's (1983) measure of satisfaction; Olson and Ives (1980) 44-item user participation measure; Tait and Vessey's (1988) 3-item measure of system complexity;

Rizzo, House, and Lirtzman's (1970) measure of task ambiguity as a surrogate for task complexity; Franz and Robey's (1986) measure of user involvement, which had 6 items relating to the design stage and 6 to the implementation stage; and Guinan's (1988) version of Monge and others' (1983) 12-item scale to measure communication quality.

2.7.3 Seddon and others analytical framework (1999)

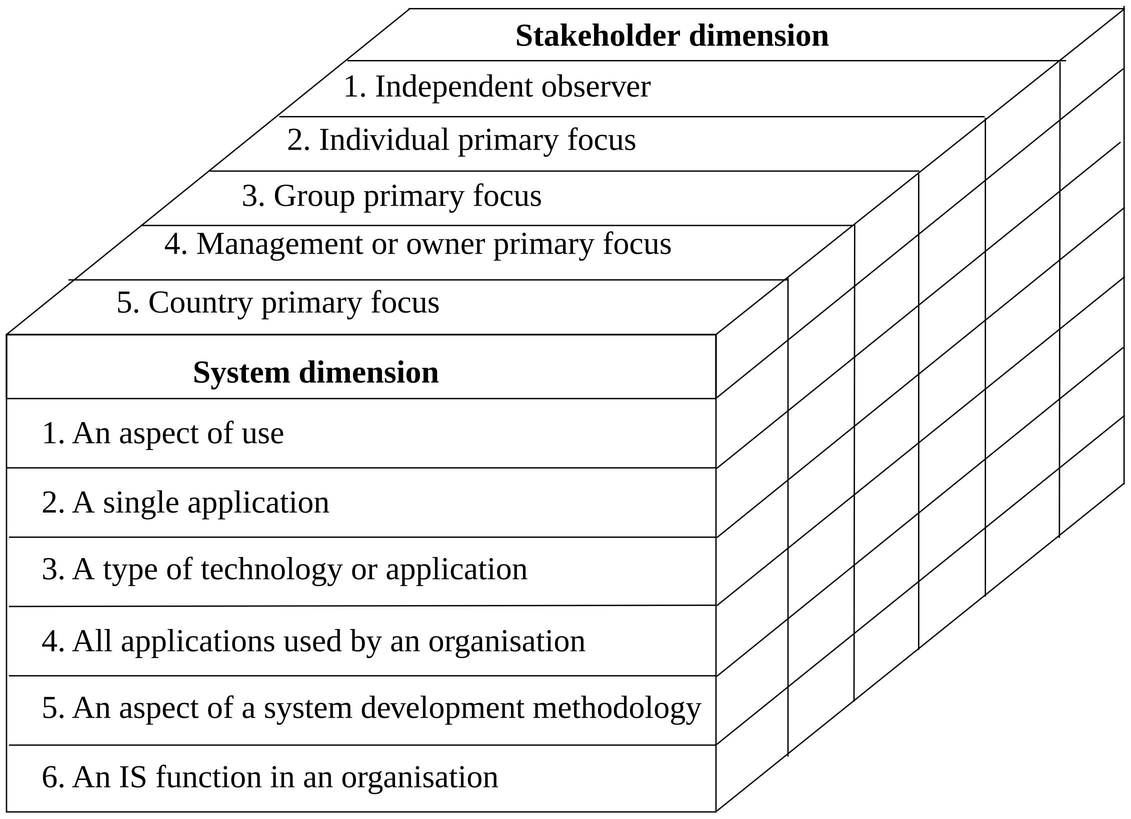


Figure 2-7. Seddon and others (1999)

Seddon and others (1999) recommended a two-dimensional approach to measuring information system effectiveness (i.e. satisfaction). It had 5 system-related values, and 6 stakeholder ones, giving a total of 30 perspectives. They suggest that different measures might be needed for each different perspective, since an individual's perspective is likely to be different from a manager's. Seddon and others' analysis of 186 empirical papers showed that studies of individual perspectives were most common (70 of 200), and that the most common information system perspective was a type of technology or application (80 of

200). The proposed study will examine individual perspectives, and will focus on application software released under an F/OSS licence.

2.7.4 Mahmood and others model (2000)

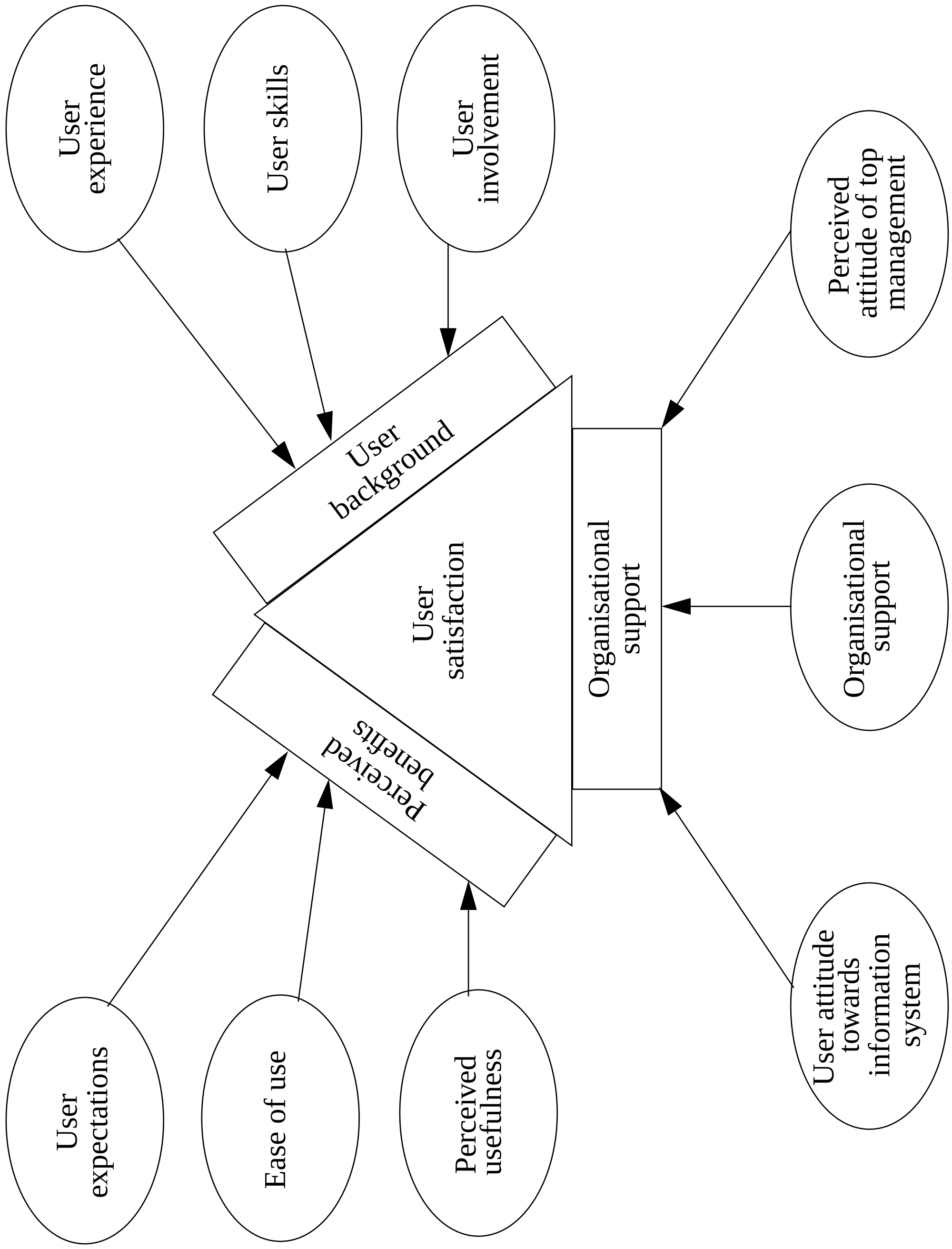


Figure 2-8. Mahmood and others (2000)

Mahmood and others developed this model by analysing 45 end-user satisfaction studies and then conducting a meta-analysis to validate the model. All 9 variables shown had a positive influence on user satisfaction, with the most significant being user involvement, perceived usefulness, user experience, organisational support, and user attitude. No new data were gathered to test the model. User involvement in system development had the largest combined effect size (0.661), followed by perceived usefulness (0.580), user experience (0.565), and organisational support (0.525).

2.7.5 Au, Ngai, and Cheng (2002) disconfirmation model

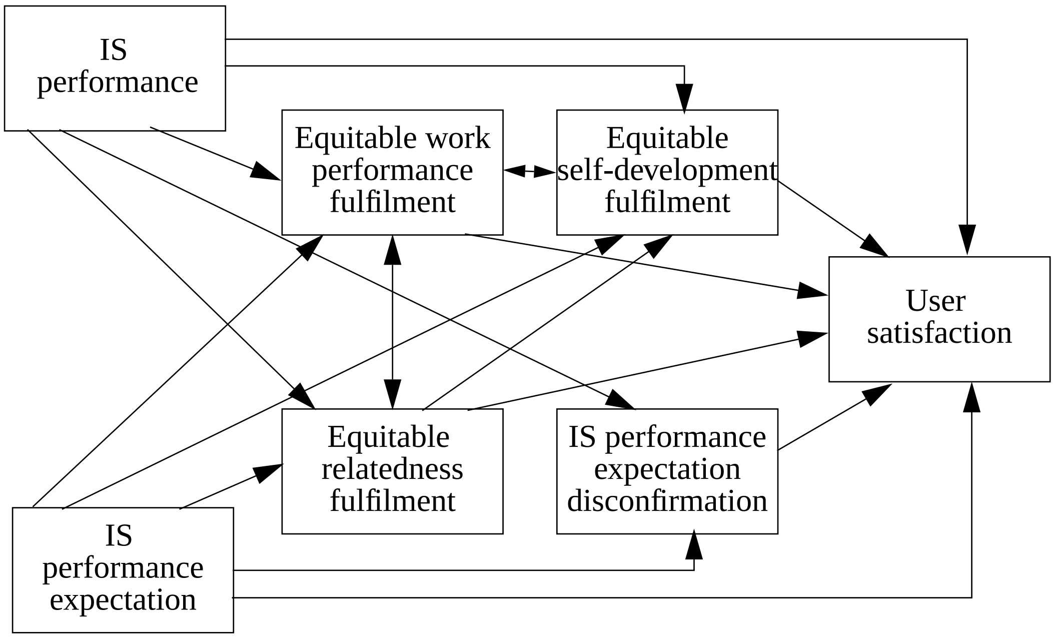


Figure 2-9. Au, Ngai, and Cheng (2002) disconfirmation model

Au, Ngai, and Cheng developed this model to provide a user satisfaction model that had a stronger theoretical grounding than other models. They took the discrepancy approach found in studies of consumer satisfaction, and combined it with equity and needs theory. The model has yet to be validated. The large number of relationships indicates a degree of complexity that makes this approach difficult to interpret.

2.7.6 Reformulated DeLone and McLean IS success model (2003)

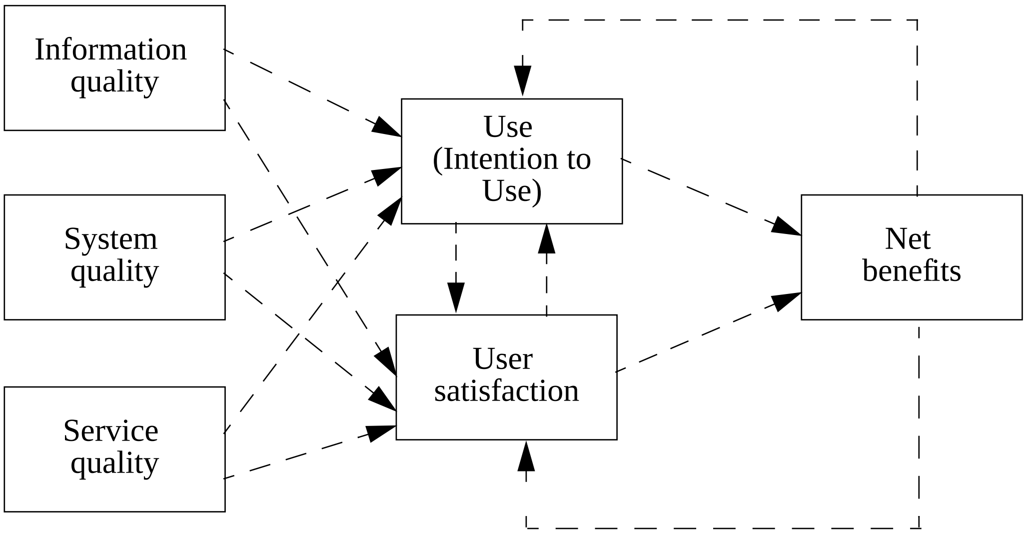


Figure 2-10. Reformulated DeLone and McLean IS success model (2003)

In 1992 DeLone and McLean published the first version of their success model, which included 'user satisfaction' as one of its components. The original model has been widely cited and tested (see Seddon and Kiew 1994 and McGill, Hobbs, and Klobas 2003 for examples). DeLone and McLean based this revision on an analysis of 150 papers that used the original model in full or in part. The main changes were the addition of 'service quality' as a factor, and the combination of 'individual impact' and 'organisational impact into the single factor 'net benefits'. This model emphasises the interdependent nature of information systems success. One point they make is that this model is intended to be 'parsimonious', and they have intentionally kept it simple. They say that it is up to studies using the model to define the context for the net benefits, suggesting that these will be measured differently for different contexts. The 'service quality' dimension was added to allow the model to be used to measure the quality of support provided by an organisation's IS department. In an F/OSS context, this dimension might apply to the community as a whole, or to members of a project's core developer group.

2.7.7 Guimaraes, Staples and McKeen user factors model (2003)

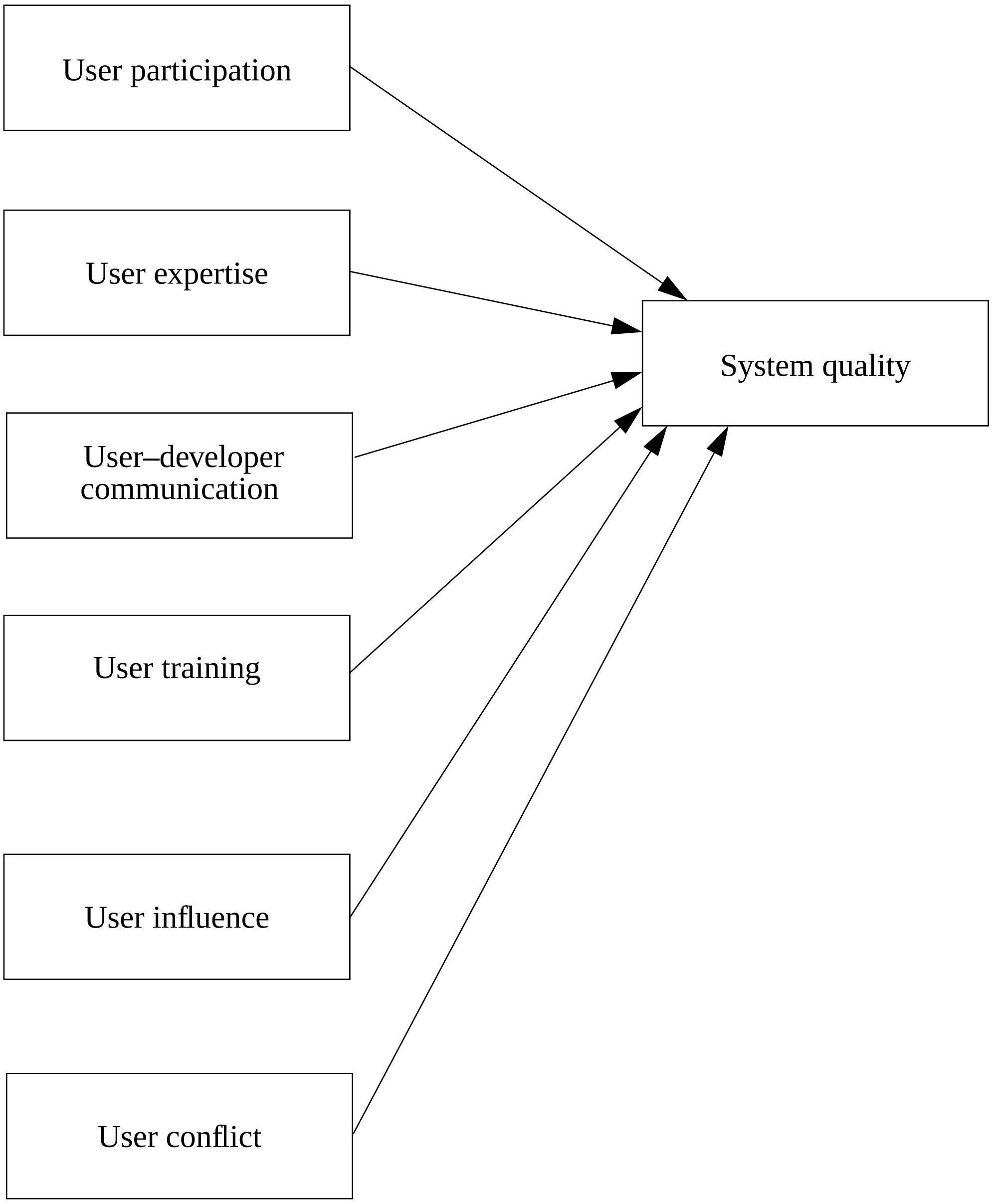


Figure 2-11. Guimaraes, Staples, and McKeen (2003) user factors model

In a survey of 228 primary users of recently developed application software, Guimaraes, Staples and McKeen used user satisfaction with information output, accuracy, usefulness and timeliness, system response/turnaround time, ease of use, and documentation usefulness as a surrogate for system quality. They found that user participation, user expertise, and user training were directly related to satisfaction, while the relationships between user–developer communication, user influence, and user conflict were not supported by the data.

2.7.8 Chin and Cooke satisfaction and coordination model (2004)

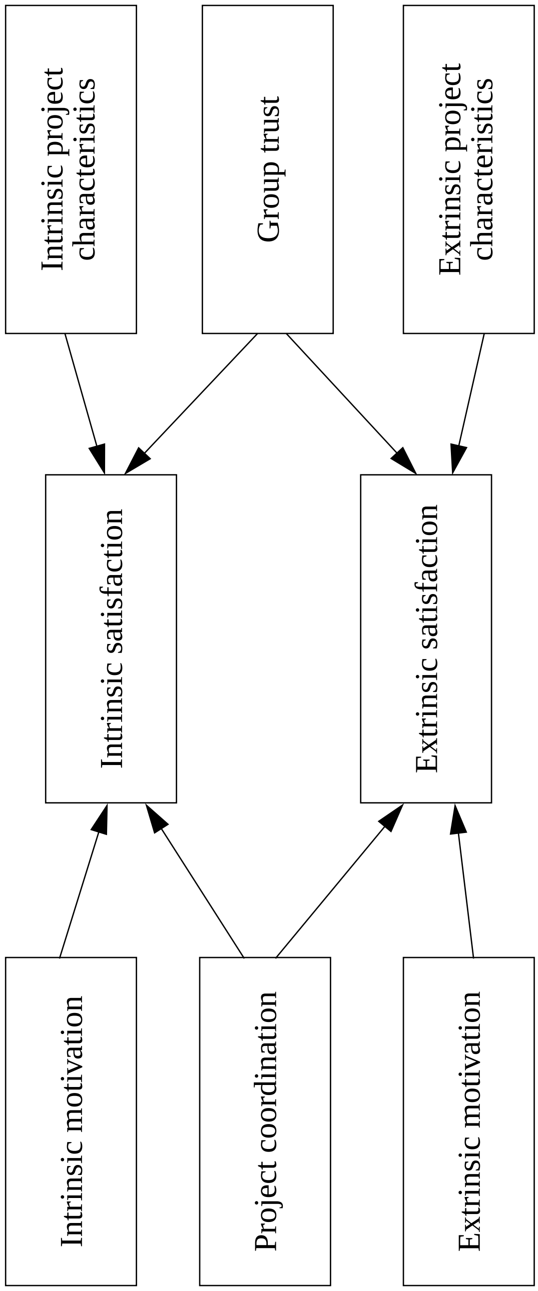


Figure 2-12. Chin and Cooke Satisfaction and coordination model (2004)

Chin and Cooke (2004) took a different approach to measuring satisfaction, developing a model specifically for F/OSS communities, and taking an approach based on 'job satisfaction', rather than the 'user' satisfaction implied in the previous models. This was based on an assumption that people who use F/OSS software take on a job-related role (often on a voluntary basis), and therefore an approach based on job satisfaction would be appropriate. Chin and Cooke measured intrinsic and extrinsic motivation using the 25-item Work Preference Inventory developed by Amabile and others (1994). Group trust was adapted from Jarvenpaa and Leidner (1999), with the project characteristics based on the FLOSS study. Job-related satisfaction was measured using 9 items from the short form Minnesota Satisfaction Questionnaire (Weiss and others 1967). In an exploratory study, they gathered data from 50 members of two open source support groups. Only 2 of the hypothesised relationships were supported: intrinsic motivation and group trust were positively related to intrinsic satisfaction. Their model did not measure the level of participation in an F/OSS project, which is a possible limitation with this approach.

2.7.9 Related research

The literature on information technology adoption/acceptance has many similarities to the literature on user satisfaction with information systems. There are four models commonly used to explain differences in adoption/acceptance:

1. Diffusion of innovation theory (Rogers 2004)
2. Theory of reasoned action (TRA) (Fishbein and Ajzen 1975)
3. Technology acceptance model (TAM) (Davis 1989)
4. Theory of planned behaviour (TPB) (Ajzen 1991)

More recently, Venkatesh and others (2003) formulated a unified theory of user acceptance and use of technology (UTAUT), which incorporated constructs from all four models, as well as four other less well known ones. This unified theory accounted for more variance in usage intention (70%) than any of the original ones when tested with empirical data (Venkatesh and others 2003: 467).

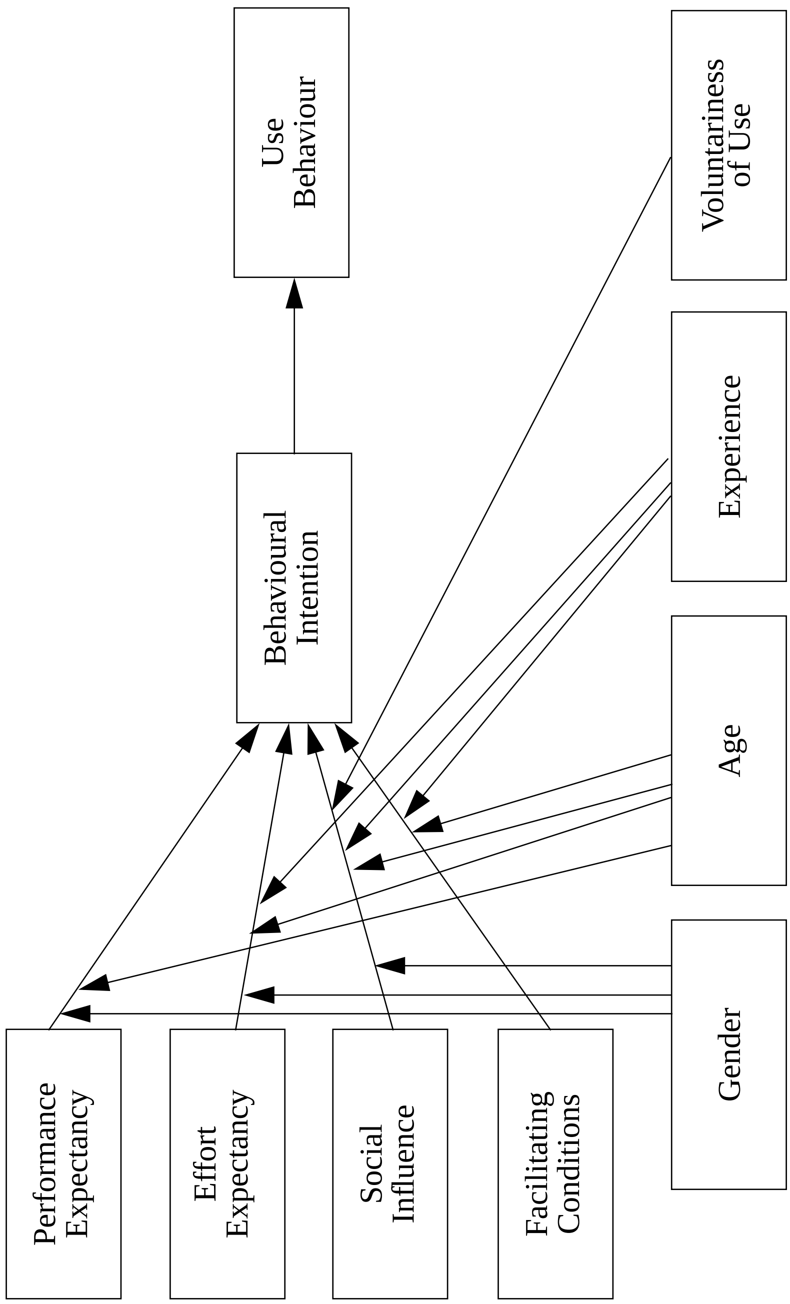


Figure 2-13. Venkatesh and others (2003) Unified Theory of Acceptance and Use of Technology (

Although the focus of UTAUT is use rather than satisfaction, several of the constructs can be related to satisfaction. Venkatesh and others define performance expectancy as the “degree to which an individual believes that using the system will help him or her to attain gains in job performance” (2003: 449). It includes perceived usefulness, relative advantage, and outcome expectations. ‘Effort expectancy’ is the “degree of ease associated with use of the system” (Venkatesh and others 2003: 450), and the constructs include perceived ease of use and complexity. 'Facilitating conditions' are defined as "the degree to which an individual believes that an organizational and technical infrastructure exists to support use of the system" ((Venkatesh and others 2003: 453). UTAUT includes the personal characteristics of gender, age, and experience as moderators of the main relationships in the model, and Venkatesh and others found that facilitating conditions were only significant when moderated by age and experience (Venkatesh and others 2003: 467).

2.7.10 Discussion

The preceding conceptual models show that there is a variety of approaches to measuring (user) satisfaction with information systems. The models vary in complexity and perspective. DeLone and McLean's revised model is intended to be generic, and they quote from their original article "the choice of success variables is often a function of the objective of the study, the organizational context, …" (2003). This suggests that they expect people to tailor their use of the model to the conditions being studied. There are similarities between the models: for example, Doll and Torzakdeh's model (1998) includes aspects of DeLone and McLean's 'information quality' (their accuracy and timeliness) and 'system quality' (ease of use and format). The Mahmood and others (1999) model is the most complex, with factors relating to the individual, the system/software, and the organisational context. The McKeen, Guimaraes, and Wetherbe (1994) model is the one that can most easily be translated into an F/OSS context, because it includes user participation and user–developer communication factors. However, it does not include user skills, experience, and expectations, which Mahmood and others' (2000) meta-analysis showed were important factors. The importance of experience as a determinant or moderator of use or satisfaction is supported by its inclusion in Venkatesh and others' UTAUT model of technology acceptance/adoption.

The most common method used in these studies is the self-administered survey, usually paper-based. This is because testing models and hypotheses requires the type of quantitative data obtained using such surveys.

2.8 Preliminary conceptual model for this research

The proposed research will use McKeen, Guimaraes, and Wetherbe's model (1994) as its basis, modifying it for an F/OSS context. The ease of use, usefulness, user skills and user experience constructs have been added to the model, given their importance in Mahmood and others' (2000) meta-analysis. Ye and others' (2004) concept of product and process openness have also been incorporated in the model.

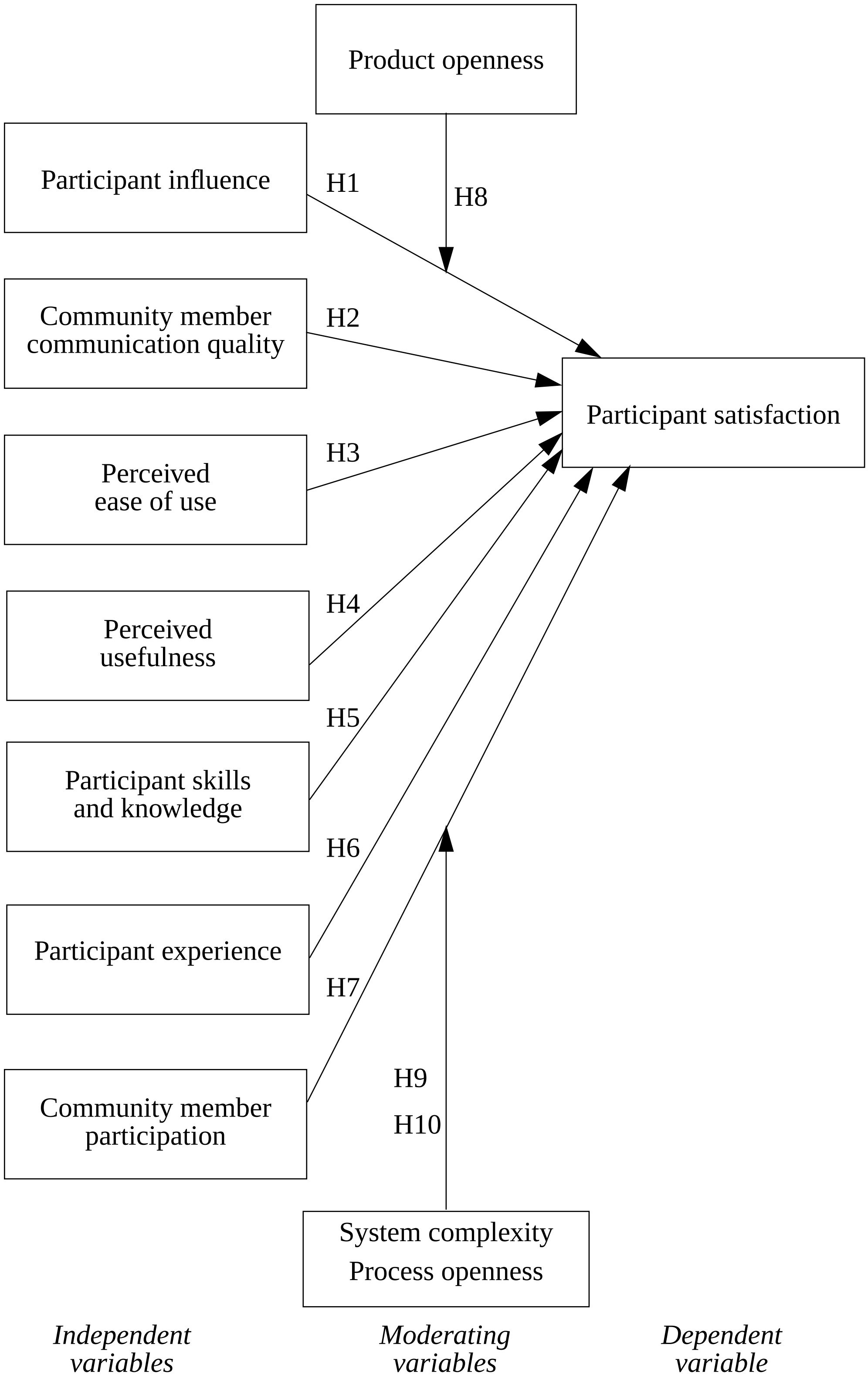


Figure 2-14. Preliminary model for this research

The constructs can be grouped into three broad categories, similar to the model of an F/OSS project shown in Figure 2-1 on page 6. Development process-related constructs are participant influence, community member communication quality, and product and process openness. Software-related constructs are ease of use, perceived usefulness, product openness, and system complexity. The third group relates to the individual community member, and consists of participant skills, participant experience, and community member participation. Each of the constructs is discussed briefly in the following sections.

2.8.1 Participant influence

Robey, Farrow, and Franz (1989) define ‘user influence’ as “the extent to which members of an organization affect decisions related to the final design of an information system”. McKeen, Guimaraes, and Wetherbe note that “without participation, there can be no influence” (1994: 434). Hartwick and Barki (1994) developed a standard 3-item measure of user influence. It asks participants to assess how much influence they had on decisions made during system development, to what extent their opinions were actually considered by others, and overall how much influence they had on the system. It was used by Guimaraes, Staples, and McKeen (2003) with a scale ranging from 1 ‘not at all’ to 5 ‘very much’.

2.8.2 Community member communication quality

This construct is been traditionally called ‘user–developer communication’ (see Guimaraes, Staples, and McKeen 2003 for a recent example); however, many F/OSS developers are also users of the software, which makes such a clear distinction between the two roles inappropriate. McKeen, Guimaraes, and Wetherbe (1994: 434) say that “[w]hat facilitates productive, collaborative effort is effective communication”, and this is the case for any software development project, whether undertaken in-house or across organisational boundaries, as in most F/OSS projects. User–developer communication is measured using questions that ask users and developers to rate the quality of communication. McKeen, Guimaraes and Wetherbe (1994) and Guimaraes, Staples, and McKeen (2003) used a modified version of an instrument originally developed by Monge et al. (1983). It includes 12 statements that respondents rate on a scale of 1 ‘very strong disagreement’ to 4 ‘neutral or don’t know’ to 7 ‘very strong agreement.

2.8.3 Perceived ease of use

This is a well-established construct that appears in conceptual models of both satisfaction and technology acceptance/adoption. It is usually measured by asking participants to assess how easy for them to use a software application package or information system, using a Likert scale.

2.8.4 Perceived usefulness

This construct is usually measured as the effect a software package or information system has on a respondent's ability to carry out his or her job (Mahmood 2001: 754).

2.8.5 Participant skills and knowledge

Torkzadeh and Lee (2003) developed a 26-item instrument to measure end-user computing skills, and tested it with a sample of 282 end-users from a range of industries and positions. Respondents were asked to rate themselves on a scale from 1 'not at all' to 5 'a great deal' against each statement. The full list of questions is included in "Appendix 1: Torkzadeh and Lee (2003) measures of end-user computing skills" on page 58

2.8.6 Participant experience

Guimaraes, Staples, and McKeen (2003) measured this by asking respondents to rate themselves relative to their peers in 5 dimensions: experience using a specific system, experience using systems of this type, experience using computers in general, being a member of a system development team, and as a member of the development team for the specific system. The proposed research will use this approach, modifying the questions to suit an F/OSS context.

2.8.7 Community member participation

Information systems development has a long tradition of involving users, starting in the 1960s (Terry and Standing 2004); this has traditionally been called 'user participation'. The reasons given for this include:

- increasing the likelihood of project success (Foster and Franz 1999);
- increasing user satisfaction (McKeen and Guimaraes 1997); and
- increasing the system's perceived usefulness (Foster and Franz 1999).

Recent literature on user involvement distinguishes between *participation*, usually described in terms of the activities the users carries out during the systems development process, and *involvement*, defined as “the subjective psychological state reflecting the importance and personal relevance that a user attaches to a given system” (Barki and Hartwick 1994: 60).

Cavaye (1995: 312) suggested that the term ‘user’ also needs clarification, noting that there are many types of user, ranging from senior management to middle management to end-users. Different types of users may undertake different forms of participation.

Participation is relatively easy to measure; typically researchers develop or adapt a list of specific behaviours or tasks users undertake and survey users to determine which behaviours or tasks they have carried out. With a traditional approach to systems development, behaviours and tasks are often linked to specific phases of development, such as feasibility, design, implementation, etc. McKeen and Guimaraes (1997) is one example of this approach, with 31 behaviours grouped by project phase. Cavaye (1995: 313) noted that the outcome of user participation is not guaranteed; user input may be ignored or it may be acted on. The model for this research includes the ‘participant influence’ construct to account for variations in the effect of participation.

The number of behaviours and tasks included in surveys of participation varies considerably. Franz and Robey (1986) identified 6 activities related to design and 7 related to development; Baroudi, Olson, and Ives (1986) had 47 development-related activities, 20 of which were general, and 27 activities associated with a specific system development phase.

Since F/OSS projects generally have a less clear boundary between users and developers, the construct has been renamed ‘community member participation’ to better reflect the F/OSS environment. The first sub-question for the proposed research, *What types of contributions do participants make to open source application software projects?* will extend the participation construct to an F/OSS context. Zhao and Deek (2004) studied user collaboration in F/OSS 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 response rate was 15.7%. Their participation measure identified 7 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. documentation.

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 cross-tabulation between questions. Finding bugs was the most common type of participation, followed by suggesting new features and finding usability problems.

2.8.8 Participant satisfaction

Section 2.7 *User satisfaction* on page 25 described a range of models of factors that affect user satisfaction with information systems. Zviran and Erlich (2003: 87) identify 6 instruments that are often used to measure user satisfaction:

1. Bailey and Pearson (1983), 39 items measuring satisfaction with an organisation's IS function;
2. Ives, Olson, and Baroudi (1983), 13 items measuring satisfaction with an organisation's IS function;
3. Miller and Doyle (1987), 38 items measuring satisfaction with an organisation's IS function;
4. Guimaraes and Gupta (1988), with 19 items measuring satisfaction with an organisation's department;
5. Doll and Torkzadeh (1988), 12 items measuring satisfaction with an IS application; and
6. Etezadi-Amoli and Farhoomand (1996), 31 items measuring satisfaction with an IS application.

The Doll and Torkzadeh (DT) (1988) and Etezadi-Amoli and Farhoomand (EAF) (1996), approaches are most suited for the proposed research, which will focus on individual F/OSS applications. The DT approach has been validated in a number of contexts, including case tools (Kim and McHaney 2000) and enterprise resource planning software (Somers, Nelson, and Karimi 2003). Doll and Torkzadeh developed their scale to measure end-user satisfaction with individual applications they used directly, in contrast to batch-oriented mainframe- or minicomputer-based systems which required system operators or analyst/ programmer intervention. Their main objective was to develop a reliable instrument that could be used to measure user satisfaction. DT has 5 first order factors and 1 second order one, shown in the following diagram.

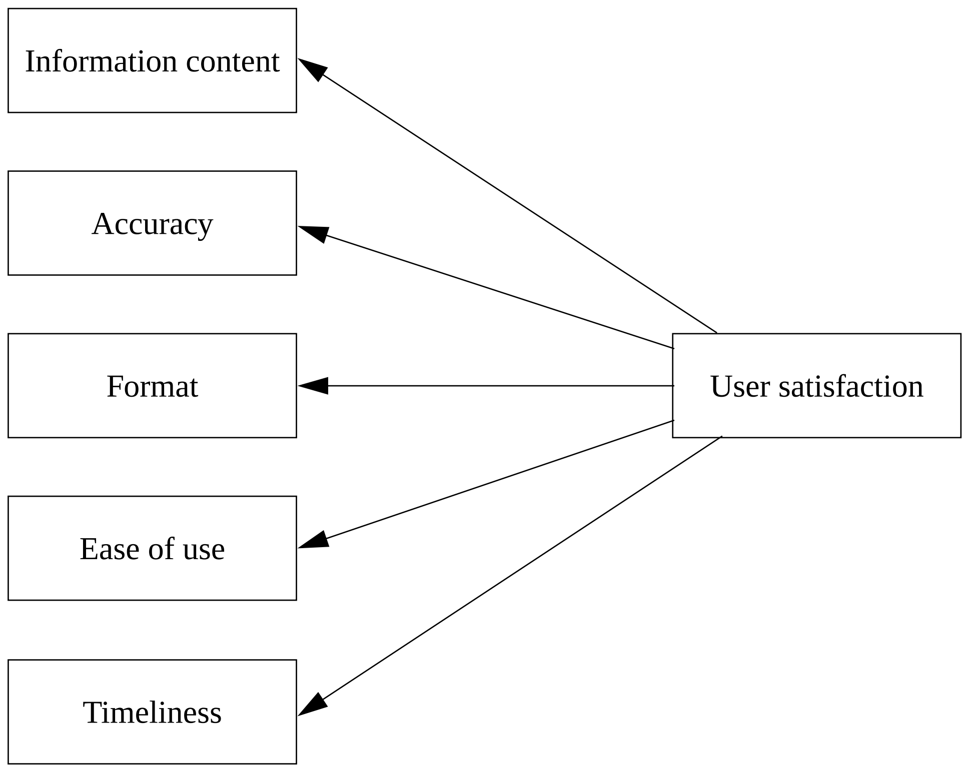


Figure 2-15. Doll and Torkzadeh (1988)

Doll and Torkzadeh initially developed a 40-item instrument with 5-point Likert scale responses to measure satisfaction with the 5 factors shown above; two summary questions asked users to rate their satisfaction with the system and its success. 618 users from 44 firms responded to the survey, representing 250 different applications. The analysis showed that 12 questions were sufficient: 4 related to content, and 2 to each of the other factors.

Etezadi-Amoli and Farhoomand (1996) also studied end user satisfaction, using the factors Documentation, Ease of Use, Functionality, Quality of output, Support, and Security, plus outcome-oriented questions to determine overall user satisfaction (which they termed ‘user performance’). The first stage of the proposed research will determine whether either of these approaches is better suited to an F/OSS context, or if a combination of factors from them would be more appropriate.

2.8.9 Product openness and Process openness

These constructs will be based on Ye and others' (2004) definitions, as described on Section 2.6.5 *Community practices* on page 24.

2.8.10 System complexity

McKeen, Guimaraes, and Wetherbe define *system complexity*, which relates to the complexity of the system under development (1994: 433). Tait and Vessey defined system complexity as "the perceived complexity associated with the analysis and design of a system" (1988: 98). The proposed research will measure task complexity using McKeen, Guimaraes, and Wetherbe's approach (1994), with 6 items related to the respondent's perception of the difficulty of the tasks they carry out with the software package. Tait and Vessey's 3-item measure of system complexity will be used, which asks respondents to assess the difficulty of determining the information requirements of the system, the complexity of the processing, and the overall complexity of the system design.

2.8.11 Hypotheses

These relate to the diagram in Figure 2-14 on page 37.

H1: The higher the participant influence, the higher the participant satisfaction with the software.

H2: The higher the quality of community member communication, the higher the participant satisfaction with the software

H3: There is a positive relationship between ease of use and participant satisfaction with the software.

H4: There is a positive relationship between perceived usefulness and participant satisfaction with the software.

H5: There is a positive relationship between participant skills and participant satisfaction with the software.

H6: There is a positive relationship between participant experience and participant satisfaction with the software.

H7: There is a positive relationship between community member participation in an F/OSS project and participant satisfaction with the software.

H8: The higher the product openness, the greater the relationship between participant influence and participant satisfaction.

H9: The higher the process openness, the greater the relationship between community member participation and participant satisfaction.

H10: The greater the system complexity, the greater the relationship between community member participation and participant satisfaction.

3 Methodology

This section begins with a discussion of the epistemological paradigm that will be taken for the proposed research, followed by a description of the population and sample, and the specific methods to be used. The proposed research will use a combination of qualitative and quantitative methods to answer the research questions. Stage 1 will be a two-part qualitative investigation of a purposive sample of F/OSS projects, and Stage 2 will be a quantitative survey of participants in a range of F/OSS projects. The results from the first Stage will be used to develop the participation construct and refine the preliminary model; the results from the second Stage will be used to test the model and resulting hypotheses.

3.1 Epistemological paradigm

The proposed research will take a positivist perspective, as it is concerned with factors that determine or cause participant satisfaction with F/OSS software. Bryman (2001: 11-12) says that positivist research generally involves the following steps:

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

This deductive approach of positivism can be contrasted with ‘interpretivism’, which is concerned more with social constructs (Gorman and Clayton 1997: 23), and is generally inductive, used for theory building rather than theory testing. Positivism is usually associated with a search for generalisable models that can be proven or disproven, while the results of interpretivist research must be placed in a specific social context (Bryman 2001: 12–16).

The positivist approach has been described as artificial and limiting, particularly when used to study complex social phenomena (Gorman and Clayton 1997: 29). The positivist focus on outcomes rather than process means that it does not result in a deep understanding of individual perspectives—positivism seeks objective answers, where interpretivism is inherently subjective. These differences between positivism and interpretivism suggest that the nature of the research question is important in determining the epistemological paradigm to be used in any research project.

The central question of the proposed research, *What factors influence participant satisfaction with an open source application software project?*, is best approached from a positivist perspective, because it assumes that specific factors can be identified, and that their influence on satisfaction can be measured. The models of factors influencing satisfaction discussed in Section 2.7 *User satisfaction* on page 25 have come from positivist studies. Because of this, the proposed research will take a positivist approach.

3.2 Population/sample

Section 2.6.1 *Issues associated with studying F/OSS projects and communities* on page 15 discussed some of the difficulties in identifying and selecting F/OSS projects for a quantitative study. The power law distribution of projects in the SourceForge repository means that selecting a random sample of projects is likely to have a large number of inactive projects, and simply selecting active projects is likely to bias the results. Hertel, Niedner, and Hermann (2003) restricted their survey to mailing lists used by members of the Linux kernel community. The proposed research will take a similar approach, using F/OSS projects from a specific application domain, library and information management (LIM). F/OSS concepts and projects have been promoted in LIM since 1998/1999, and a wide range of F/OSS projects are available in the field. The OSS4LIB web site (www.oss4lib.org) listed over 80 projects in September 2004. 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 (Avanti, Koha, OpenBiblio, and PhpMyLibrary) to software for creating digital libraries (Greenstone), suggesting that this sub-population will provide a suitably diverse range of projects for the proposed research.

3.3 Stage 1: Qualitative investigation

This stage will use qualitative methods to develop the participation construct and validate the preliminary model of user satisfaction in Section 2.8 *Preliminary conceptual model* *for this research* on page 36. A qualitative approach has been 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: 284–285).

3.3.1 Stage 1a: Development of participation construct

This phase will involve observation of purposive sample of 10-15 LIM F/OSS projects. Projects will be chosen to include a range of community sizes (measured as number of participants), project ages, software complexity, and activity levels, in order to develop a preliminary list of activities carried out by participants in the projects. A purposive sample has been chosen for this phase in order to gain an in-depth understanding of the first research sub-question, *What types of contributions do participants make to open source application software projects?*, as suggested by Cresswell (2003: 185). He says that a purposive sample is used to choose the *best* participants to help the researcher understand the problem (my emphasis), and this is the most suitable approach for this phase. Documentation, email discussion archives, source code, and web sites will be examined to identify the types of activities carried out by project community members.

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

The next phase of the qualitative stage will involve individual interviews, in order to validate and improve the preliminary research model shown in Figure 2-14 on page 37. Interviewees will be purposively selected from participants involved in a range of activities in LIM F/OSS projects. As with Stage 1a, a purposive sample is appropriate to ensure that interviewees have a range of viewpoints and experience. This population will include the authors of articles and conference papers about LIM F/OSS projects, as well as people who participate in relevant email discussion lists and contribute to the oss4lib portal. The purposive sample will seek to include people with a range of skills and experience, in addition to representing different types of involvement in a project. The number of interviews is likely to be between 15 and 20, but the final number will be based on reaching theoretical saturation—in other words, when no significant new insights are likely to be gained by further interviews (Bryman 2001: 303).

Since the LIM F/OSS community is global, with projects and participants based in North America, Europe, and Australasia, email will be used as the main interview communication medium, with face-to-face interviews being done when possible. The advantage of using an asynchronous email technique is that respondents do not need to be in a compatible time zone, unlike synchronous techniques such as face-to-face, phone, or chat. It will also allow respondents to reply at their convenience (Mann and Stewart 2003: 82). For the email interviews, the interview questions will be broken into three groups of questions, with each group sent in a separate message over a period of 2–3 weeks. The questions will be in the body of the message and interviewees will answer them using the 'reply' command. This approach has been selected to increase people's willingness to participate, as it is will involve a smaller amount of time for each reply than if all of the questions were sent at once in a single message. Using this technique will also allow the researcher to establish a 'key informant'-style relationship with remote respondents (Gorman and Clayton 1997: 87), because the interaction will take place over a longer time than a single interview would.

The interviews will use open-ended questions to gather rich qualitative data focusing on satisfaction with F/OSS software and the concepts included in the preliminary research model. In addition, participants will be asked to comment on the results of stage 1a, to validate the results of the observation and give them what Bryman terms 'credibility' (2001: 272). Credibility is a measure of the acceptability of research results to others. Olson and Ives used practitioners to comment on system development activities in establishing their framework to measure user involvement in system development (1980), showing that it is a suitable approach for the proposed research.

The main advantage of using semi-structured interviews for this stage is that the interviewer can probe for further elaboration, if the original answer was brief. This can be done in both modes. Possible issues associated with this approach are that the interviewer will have an influence on the replies, and that this will not be consistent between the two modes of interviewing.

3.4 Stage 2: Quantitative survey

The second stage of the proposed research will involve developing and administering a quantitative, web-based survey to test the model and hypotheses. Quantitative methods are best suited for collecting quantifiable data for statistical testing (Bryman 2001: 20). This type of data is needed to test the model and hypotheses, and provide an answer to the primary research question *What factors influence participant satisfaction with an open source application software project?*. This data will also be used to answer the second sub-question *Do the factors that influence satisfaction with an open source application software project differ for different roles? If they do, in what way?*

Other research methods that provide data suitable for hypothesis testing, such as conducting a laboratory or field experiment, are not appropriate for the proposed research, as one of the key characteristics of F/OSS projects is that participation is voluntary, and that the roles people take on are self-assigned.

3.4.1 Stage 2a: Developing the survey instrument

The results of Stage 1 will be used to revise the conceptual model and develop a web-based survey to test it. Where possible, existing scales will be used or modified for the model's constructs. New scales will be developed for the product and process openness, and may also be needed for community member participation.

3.4.2 Stage 2b: Pilot study

The survey will be tested using in a pilot study involving 8–10 respondents selected from a non-LIM F/OSS project. The main purpose of the pilot study will be to test the survey instrument, in order to improve the format, questions, and scales. Pilot study respondents will be asked to complete the survey and given an opportunity to comment on its length, wording, and structure. Their comments will be used to modify the instrument as necessary.

3.4.3 Stage 2c: Web-based survey

The last data collection will consist of a web-based survey to gather quantitative data to test the model and hypotheses.

Population and sample. As discussed in Section 3.2 *Population/sample* on page 46, the proposed research will use LIM F/OSS projects as its population. All active projects listed on the oss4lib portal will be included in this phase, where an active project is defined as one that has software available for downloading as well as an identifiable project manager/owner. In order to attract respondents who represent a range of participation levels, an invitation to complete the survey will be issued on each project's email discussion list or equivalent. This approach means that it will not be possible to calculate a response rate, and may make it difficult to measure any non-response bias. The technique of harvesting individual email addresses from email discussion list messages in order to issue individual invitations to participate, commonly used in F/OSS survey research (see Sagers 2004 for an example), would result in a bias toward active project participants. The approach of issuing an open invitation is better suited to the proposed research, which requires responses representing a range of participation.

The target number of responses will be determined by the number of constructs in the revised version of the model. Newton and Rudestam's rules of thumb will be used to estimate the target number of responses: $50 + (8 \times \text{number of variables})$ for calculating R^2; based on the proposed model, this means a minimum of 130 usable responses (1999, quoted in de Vaus 2002: 357). However, to increase the reliability of the research results, and to allow a more detailed analysis based on participant roles, a larger number will be required. The general rule of thumb is the larger the number of responses the better, up to a sample size of 1,000 (Bryman 2001: 95).

Duration. The survey will be available online for one month following the initial invitation to participate. After two weeks, a thank you/reminder message will be sent to the selected email discussion lists, in order to encourage laggards to participate.

Advantages of this approach are that the survey is simple to administer, and respondents can complete it at their convenience. Disadvantages are that it is not possible to control who answers, and the response rate is likely to be low. It is hoped that by issuing invitations only on email discussion lists for LIM F/OSS projects, potential respondents will have an interest in the research topic, and therefore be motivated to participate. Respondents might also abandon the survey before it is complete—this cannot be prevented, since participation in the survey will be voluntary. Another disadvantage is that responses are limited to the options presented, and that respondents might wish to offer further comments.This will be dealt with by having a final open-ended question on the survey form, asking if respondents have any further comments to make about the topics of the survey.

3.5 Ethical considerations

Both stages of the proposed research involve human subjects, and HEC permission will be required. In Stage 1b, interviewees will be initially invited to participate by phone or email. If they are interested, they will be sent a hard copy project information sheet and consent form. In Stage 2c, the project 'owner' will be asked for permission to send an invitation to participate in the web-based survey to the project's email discussion group or equivalent. If they agree, they will be sent an electronic project information sheet. Respondents to the web-based survey will see an electronic information sheet before beginning the survey, and will need to indicate their willingness to participate by clicking a 'continue' button.

All participants in both stages will be offered a summary of the research findings. In addition, they will be given information about the project blog, which will be updated regularly. The blog software will allow readers to comment on the research, and this opportunity may keep them interested in the research during the analysis phase.

4 Data analysis

4.1 Stage 1 analysis

Both phases of this stage will gather qualitative data, and content analysis will be the main approach to the analysis. This analytical technique involves identifying themes and patterns in the data (Gorman and Clayton 1997: 204).

4.1.1 Stage 1a: Observation of selected LIM F/OSS projects

Analysis of data collected during this stage will begin by using open coding to develop an initial list of participant activities. Axial coding will then be used to identify categories or groups of related activities. This approach is similar to the type of coding used in grounded theory research, as outlined in Bryman (2001: 392).

The results of this stage will provide a preliminary answer to the first research sub-question, *What types of contributions do participants make to open source application software projects?*

4.1.2 Stage 1b: Semi-structured interviews with participants in selected LIM F/OSS projects

The preliminary conceptual model for the research will be used to generate initial categories for coding the data; these will be refined and extended as the data analysis takes place. Face-to-face interviews will be recorded, and the recordings will be transcribed as soon as possible. Email interviews are ‘self-transcribed’, and the text will be immediately available for analysis.

Bryman recommends coding as soon as possible (2001: 398); in keeping with this recommendation, coding will begin when each interview is complete. When all interviews are complete and the data have been coded, the final part of the analysis will take place. This will involve examining the coded data for patterns that support the preliminary model of factors that influence participant satisfaction with F/OSS application software given in Section 2.8 on page 36, to validate constructs in the model. It will also involve identifying patterns that reflect constructs that are not in the model. Once this is done, the model will be revised as necessary, removing constructs that have not been identified by interviewees, and adding new ones as necessary.

4.2 Stage 2c analysis

This web-based survey used in Stage 2c will gather quantitative data suitable for statistical analysis. SPSS will be the main tool for the analysis.

Previous studies of models of user satisfaction have used a variety of statistical techniques, as shown in the following table:

Table 4-1. Statistical methods used in user satisfaction modelling

Technique	Used in
Multiple regression	Seddon and Yip (1992); Yoon and Guimaraes (1995); Guimaraes, Staples, and McKeen (2003)
Partial least squares	Igbaria and Tan (1997); Blili, Raymond and Rivard (1998); Venkatesh and others (2003)
Path analysis	Tait and Vessey (1988); Seddon and Kiew (1994)
Hierarchical regression	Franz and Robey (1986); Guimaraes, Igbaria, and Lu (1992)
Moderated regression analysis	McKeen, Guimaraes, and Wetherbe (1994)
Structural equation modelling	Etezadi-Amoli and Farhoomand (1996); Kim and McHaney (2000); McGill, Hobbs, and Klobas (2003); Evanschitzky and others (2004)

The nature of the model and hypotheses in part determine the most appropriate statistical technique to use, as does the nature of the data collected. Because the preliminary model has a single dependent variable, hierarchical multiple regression will be used to analyse the data. Path analysis and structural equation modelling are more suitable for more complex models with multiple dependent variables (Bryman and Cramer 2005: 313).

This will first involve testing the assumptions that underlie multiple regression (de Vaus 2003: 343-344). This will involve examining correlations between pairs of independent variables. If this shows significant correlations, multiple regression will be used to test for correlations between the independent variables—in other words, seeing if one of them depends on any of the others. Descriptive statistics (kurtosis and skewness) will be used to see if the values for each variable follow a normal distribution.

Using a hierarchical regression process will allow the order of the independent variables to be specified by the researcher. This will result in a value of R^2 for the first independent variable, and with a re-calculated R^2 for each subsequent one. This will show how much variance the variables in the direct relationships account for, excluding the effect of the moderating variables.

The interaction effects of the proposed moderator variables will be tested by splitting the data into subgroups by moderator value, and initially calculating Pearson's r for each group and comparing the values (Bryman and Cramer 2005: 295). An alternative would be to use moderated regression combined with subgroup analysis. Subgroup analysis involves splitting the data into different subgroups and analysing each separately. If the regression coefficients for the groups are significantly different, the contingency factor used to group the data is assumed to be a moderator variable (McKeen, Guimaraes and Wetherbe 1994: 440). Moderated regression involves calculating regression coefficients for three equations:

1. $S = a + b_1CP$
2. $S = a + b_1CP + b_2M$
3. $S = a + b_1CP + b_2M + b_3CPM$

where M is the moderator variable (based on McKeen, Guimaraes and Wetherbe 1994: 441). If M is a pure moderator of the relationship, equations 1 and 2 will be the same, and they will also be different from equation 3. If M is not a moderator, equations 2 and 3 will not be significantly different.

The results of this analysis will provide an answer to the main research question, *What factors influence participant satisfaction with an open source application software project*.

Further statistical analysis will be needed to answer the second sub-question *Do the factors that influence satisfaction with an open source application software project differ for different roles? If they do, in what way?*. This is will involve splitting the data into subgroups based on responses to questions about participation, and repeating the regression analysis for each resulting group. The regression coefficients will be compared with the results for the initial analysis, to see if significant differences exist.

4.3 Hypothesis testing and model revision

The results of the regression analysis will be used to test the hypotheses. Only statistically significant values ($p <= 0.05$) will be consider to support them. If any of the hypotheses are not supported by the findings, the conceptual model used for the research will be revised, and re-tested using the data from Stage 2c to find a ‘best’ fit model.

4.4 Reliability

Reliability is the degree to which the measures result in consistent responses (de Vaus 2002: 17). Stage 1 of the proposed research will achieve reliability by drawing on concepts taken from the literature on F/OSS project structure and factors that contribute to user satisfaction with information systems (Gorman and Clayton 1997: 60). The proposed research will be based on existing scales where possible in the Stage 2 survey instrument to increase reliability. In addition, the survey instrument is expected to have multiple items per construct; Cronbach’s alpha will be used to test the internal consistency of responses, as an indication of reliability. In keeping with standard recommendations, a minimum alpha value of 0.7 will be required to indicate a reliable set of questions (de Vaus 2002: 20).

4.5 Validity

Validity is defined as the extent to which the measures used match the construct (de Vaus 2002: 25); in other words, is the right thing being measured? Stage 1b will achieve validity by using an interview guide to ensure that all participants are asked the same questions. Stage 2 will increase validity by using existing measurement scales where possible. The data analysis will test discriminant and convergent validity using factor analysis, including correlations.

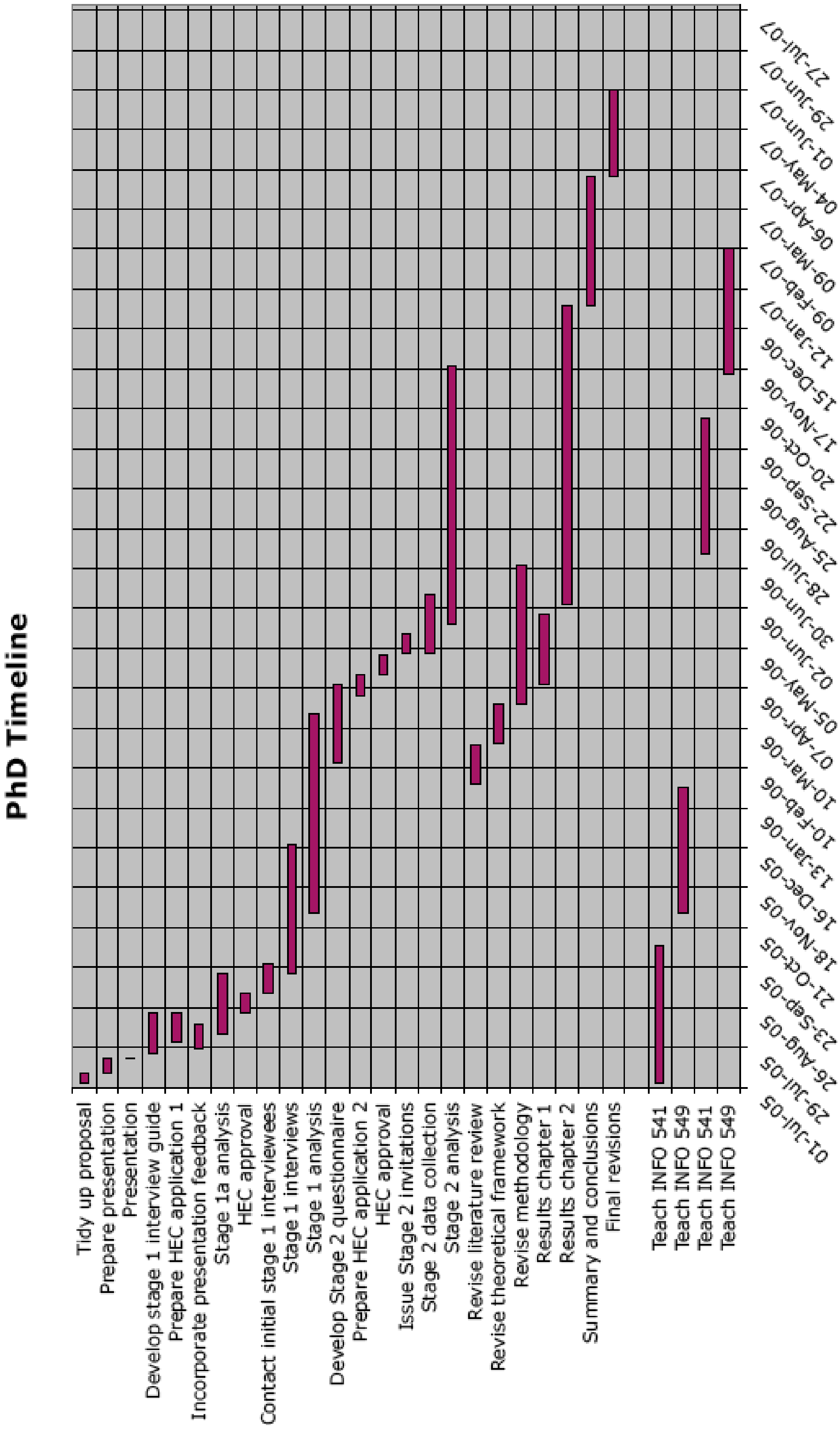
4.6 Limitations

All research has limitations, and the proposed project is primarily limited by its use of LIM F/OSS projects as its population. As these are a subset of all F/OSS projects, it is unlikely that the results can be generalised to all F/OSS contexts. The limitations will be minimised by encouraging a high number of responses and using purposive sampling to ensure a wide range of values in responses.

The use of a self-administered web-based surevy is also a potential limitation, as it assumes that all respondents will interpret the questions in the same way and will answer them honestly. This limitation has been addressed in the research design by proposing to conduct a pilot study before the main survey, in order to identify any difficulty in answering the questions and revise the survey accordingly.

The two stage approach proposed for this research is intended to overcome the limitations of a purely quantitative approach, in that it allows the preliminary model to be modified before the main data are gathered.

5 Project timetable



Appendix 1: Torkzadeh and Lee (2003) measures of end-user computing skills

1. Knowledge and use of hardware
2. Knowledge and use of advanced programming languages
3. Knowledge and use of operating systems
4. Knowledge and use of mainframe and its operating systems
5. Knowledge and use of packaged application software
6. Knowledge and use of databases
7. Ability to design input forms/screens
8. Ability to design output formats
9. Ability to define your own information requirements
10. Ability to provide the system designer(s) with information/knowledge required to develop a system
11. Ability to assess system needs or evaluate system features
12. The extent to which you create your own application
13. Analytical skill/knowledge such as model building, decision theory, financial, statistical
14. Use of management science techniques in your job
15. Knowledge of your business environment
16. Ability to perform your business tasks independently of your colleagues or supervisors
17. Experience in designing application software
18. Experience in developing application software
19. Experience in using application software
20. Experience in maintaining/modifying application software
21. Experience in acquisition of computer systems
22. Experience in implementing computer systems
23. Ability to use computers in your job
24. The extent to which you depend on help of others to understand or use computers (reverse scale)
25. The extent of your confidence in accepting new computer applications

26.Overall, how would you rate your computer ability?

Bibliography

Antoniades, I.P., I. Samoladas, I. Stamelos, L. Angelis, and G.L. Bleris. 2004. Dynamical simulation models of the open source development process. In *Free/open source software development*, edited by S. Koch, 174-202. Hershey, PA: Idea Group.

Applegate, Rachel. 1997. Models of satisfaction. In *Encyclopedia of library and information science*, edited by A. Kent, H. Lancour, W. Z. Nasri and J. E. Daily, 199-227. New York: Marcel Dekker.

Association of Computing Machinery. 2005. ACM software copyright and license agreement, June 1998 [cited 27 January 2005]. Available from http://www.acm.org/pubs/copyright_policy/softwareCRnotice.html.

Atkinson, Paul, and Amanda Coffey. 2003. Revisiting the relationship between participant observation and interviewing. In *Postmodern interviewing*, edited by J. F. Gubrium and J. A. Holstein, 109-122. Thousand Oaks, CA: Sage Publications.

Au, Norman, Eric W. T. Ngai, and T.C. Edwin Cheng. 2002. A critical review of end-user information system satisfaction research and a new research framework. *Omega* 30 (6):451-478.

Barki, Henri, and Jon Hartwick. 1994. Measuring user participation, user involvement, and user attitude. *MIS Quarterly* 18 (1):59-82.

Baroudi, Jack J., Margrethe H. Olson, and Blake Ives. 1986. An empirical study of the impact of user involvement on system usage and information satisfaction. *Communications of the ACM* 29 (3):232-238.

Basset, Thomas. 2004. Coordination and social structures in an open source project: VideoLAN. In *Free/open source software development*, edited by S. Koch, 125-151. Hershey, PA: Idea Group.

Behlendorf, Brian. 1999. Open source as a business strategy. In *Open sources: voices from the Open Source revolution*, edited by C. DiBona, S. Ockman and M. Stone, 149-170. Sebastopol, CA: O'Reilly.

Bentley, Jon. 1986. *More programming pearls: confessions of a coder*. Reading, Ma: Addison-Wesley. Original edition, 1986.

———. 1989. *Programming pearls*. Reading, Ma: Addison-Wesley. Original edition, 1986.

Berry, David M. 2004. Internet research: privacy, ethics and alienation: an open source approach. *Internet Research* 14 (4):323-332.

Bleek, Wolf-Gideon, and Matthias Finck. 2004. Migrating a development project to open source software development. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 9-13.

Blili, S., L. Raymond, and S. Rivard. 1998. Impact of task uncertainty, end-user involvement, and competence on the success of end-user computing. *Information & Management* 33 (3):137-153.

Boldyreff, Cornelia, David Nutter, and Stephen Rank. 2004. Communication and conflict issues on collaborative software research projects. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 14-17.

Bonaccorsi, Andrea, and Cristina Rossi. 2003. Why open source software can succeed. *Research Policy* 32 (7):1243-1258.

———. 2004. Contributing to OS projects: a comparison between individual and firms. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 18-22.

Boyle, Brendan. 2003. Open source software: briefing to the Minister of State Services. Wellington: State Services Commission.

Bragge, Johanna, Peentti Marttiin, and Tuure Tuunanen. 2005. Developing innovative information systems services together with wide audience end-users. Paper read at 38th Hawaii International Conference on System Sciences, 3-6 January, at Island of Hawaii, Hawaii.

Breeding, Marshall. 2002. The open source ILS: only a distant possibility. *Information Technology and Libraries* 21 (1):16-18.

Bretthauer, David. 2001. Open source software in libraries. *Library Hi Tech News* 18 (5):8-9.

———. 2002. Open source software: a history. ***Information Technology and Libraries*** 21 (1):3-10.

Bryman, Alan. 2001. *Social research methods*. Oxford: Oxford University Press.

Bryman, Alan, and Duncan Cramer. 2005. *Quantitative data analysis with SPSS 12 and 13*. London: Routledge.

Butler, Tom, and Brian Fitzgerald. 1997. A case study of user participation in the information systems development process. Paper read at Eighteenth international conference on Information systems, 15-17 December, at Atlanta, GA.

Capiluppi, Andrea. 2004. Improving comprehension and cooperation through code structure. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 23-28.

Capiluppi, Andrea, Patricia Lago, and Maurizio Morisio. 2003. Evidences in the evolution of OS projects through changelog analysis. In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 19-22.

Cavaye, Angèle L. M. 1995. User participation in system development revisited. *Information & Management* 28 (5):311-323.

Chawner, Brenda. 2002. Koha: an open source success story. *Library Link* November.

Chawner, Brenda. 2003. Open communication for open source software development: the Koha experience. Paper read at LITA National Forum, 3-5 October, at Norfolk, VA.

Chengalur-Smith, Shobha, and Anna Sidorova. 2003. Survival of open-source projects: a population ecology perspective. Paper read at Twenty-fourth international conference on Information systems, 14-17 December, at Seattle, WA.

Chin, Pauline O., and Donna Cooke. 2004. Satisfaction and coordination in virtual communities. Paper read at 10th Americas Conference on Information Systems, 6-8 August, at New York, NY.

Chudnov, Daniel. 1999. Open source software: the future of library systems. *Library Journal* 124 (13):40-43.

Cooper, Michael D. 1996. *Design of library automation systems: file structures, data structures, and tools*. New York: John Wiley.

Cresswell, John W. 2003. *Research design: qualitative, quantitative, and mixed methods approaches*. Thousand Oaks, CA: Sage Publications.

Crowston, Kevin, Hala Annabi, and James Howison. 2003. Defining open source software project success. Paper read at Twenty-fourth International Conference on Information Systems, 14-17 December, at Seattle, WA.

Crowston, Kevin, Hala Annabi, James Howison, and Chengetai Masango. 2004. Effective work practices for software engineering: free/libre open source software development. Paper read at Workshop on Interdisciplinary Software Engineering Research 2004, 5 November, at Newport Beach, CA.

———. 2004. Towards a portfolio of FLOSS project success measures. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 29-33.

———. 2005. Effective work practices for FLOSS development: a model and propositions. Paper read at HICSS-38: Hawaii International Conference on Systems Sciences, 3-6 January, at Big Island, Hawaii.

Cubranic, Davor, Reid Holmes, Annie T.T. Ying, and Gail C. Murphy. 2003. Tools for light-weight knowledge sharing in open-source software development. In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 25-29.

Damodaran, Leela. 1996. User involvement in the systems design process—a practical guide for users. *Behaviour & Information Technology* 15 (6):363-337.

Davis, Charles H., and Gerald R. Lundeen. 1981. *Illustrative computer programming for libraries: selected examples for information specialists*. Westport, Connecticut: Greenwood.

Davis, Fred D. 1989. Perceived usefulness, perceived ease of use, and use acceptance of information technology. *MIS Quarterly* 13 (3):319-340.

de Vaus, David. 2002. *Analyzing social science data: 50 key problems in data analysis.* London: Sage.

Dedrick, Jason, and Joel West. 2004. An exploratory study into open source platform adoption. Paper read at HICSS-37: Hawaii International Conference on Systems Sciences, 5-8 January, at Big Island, Hawaii.

DeLone, William H., and Ephraim R. McLean. 2002. Information systems success revisited. Paper read at HICSS-35: Hawaii International Conference on System Sciences, 7-10 January, at Big Island, Hawaii.

DiBona, Chris, Sam Ockman, and Mark Stone, eds. 1999. *Open sources: voices from the Open Source revolution.* Sebastopol, CA: O'Reilly.

Diker, Vedat. 2003. Toward a dynamic theory of open online collaboration communities. Paper read at 9th Americas Conference on Information Systems, 4-6 August, at Tampa, FL.

Divitini, Monica, Letizia Jaccheri, Eric Montiero, and Hellvard Traetteberg. 2003. Open source processes: no place for politics? In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003,* edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 39-43.

Doll, William J., and Gholamreza Torkzadeh. 1988. The measurement of end-user computing satisfaction. *MIS Quarterly* 12 (2):259-274.

Elliott, Margaret S. 2003. The virtual organizational culture of a free software development community. In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003,* edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 45-49.

Elliott, Margaret S., and Walt Scacchi. 2004. Free software development: cooperation and conflict in a virtual organizational culture. In *Free/open source software development,* edited by S. Koch, 152-172. Hershey, PA: Idea Group.

Erenkrantz, Justin R. 2003. Release management within open source projects. In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003,* edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 51-55.

Etezadi-Amoli, Jamshid, and Ali F. Farhoomand. 1996. A structural model of end user computing satisfaction and user performance. *Information & Management* 30 (2):65-73.

Eunice, Jonathan. 2005. The cathedral and the bazaar, 11 May 1998 [cited 28 January 2005]. Available from http://www.illuminata.com/cgi-local/pub.cgi?docid=cathedral.

Evanschitzky, Heiner, R. Iyer Gopalkrishnan, Josef Hesse, and Dieter Ahlert. 2004. E-satisfaction: a re-examination. *Journal of Retailing* 80 (3):239-247.

Feller, Joseph, and Brian Fitzgerald. 2002. *Understanding open source software development.* London: Addison-Wesley.

Fink, Martin. 2003. *The business and economics of Linux and open source*. Upper Saddle River, NJ: Prentics Hall PTR.

Fontana, Andrea. 2003. Postmodern trends in interviewing. In *Postmodern interviewing,* edited by J. F. Gubrium and J. A. Holstein, 51-65. Thousand Oaks, CA: Sage Publications.

Foster, S. Thomas, 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.

Fowler, Dennis. 2000. Open season. *netWorker* 4 (2):18-25.

Franz, Charles R., and Daniel Robey. 1986. Organizational context, user involvement, and the usefulness of information systems. *Decision Sciences* 17 (3):329-356.

Free Software Foundation. 1989. The General Public License as a subroutine. GNU's Bulletin 1 (6).

———. 2005. Various licenses and comments about them, 19 January 2005 [cited 25 January 2005]. Available from http://www.fsf.org/licenses/license-list.html.

Gacek, Cristina, and Budi Arief. 2004. The many meanings of open source. *IEEE Software* 21 (1):34-40.

Gacek, Cristina, Tony Lawrie, and Budi Arief. 2002. Interdisciplinary insights on open source. Paper read at Open Source Software Development Workshop, 25-26 February 2002, at Newcastle upon Tyne.

Galletta, Dennis F., and Albert L. Lederer. 1989. Some cautions on the measurement of user information satisfaction. *Decision Sciences* 20 (3):419-438.

Gallivan, Michael J., and Mark Keil. 2003. The user–developer communication process: a critical case study. *Information Systems Journal* 13 (1):37-68.

Geard, Jennifer. 2001. A matter of mindshare: open source software for libraries. *Library Life* (256):29-31.

Gelderman, Maarten. 1998. The relation between user satisfaction, usage of information systems and performance. *Information & Management* 34 (1):11-18.

German, Daniel, and Audris Mockus. 2003. Automating the measurement of open source projects. In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003,* edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 63-67.

Golden, Bernard. 2005. *Succeeding with open source*. Boston: Addison-Wesley.

González-Baharona, Jesús M., Luis Løpex, and Gregorio Robles. 2004. Community structure of modules in the Apache project. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 43-47.

González-Baharona, Jesús M., Gregorio Robles, Miguel Ortuño-Pérez, José Centeno-González, Vicente Matellán-Olivera, Eva Castro-Barbero, and Pedro de-las-Heras-Quirós. 2004. Analyzing the anatomy of GNU/Linux distributions: methodology and case studies. In *Free/open source software development*, edited by S. Koch, 27-58. Hershey, PA: Idea Group.

Gorman, G.E., and Peter Clayton. 1997. *Qualitative research for the information profession: a practical handbook*. London: Library Association.

Graham, Susan L. 2001. From research software to open source. In *Informatics: 10 years back, 10 years ahead*, edited by R. Wilhelm. Berlin: Springer-Verlag.

Grodzinsky, Frances S., Keith Miller, and Marty Wolf. 2003. Ethical issues in open source software, 351-366.

Gubrium, Jaber F., and James A. Holstein. 2003. Postmodern sensibilities. In *Postmodern interviewing*, edited by J. F. Gubrium and J. A. Holstein, 3-18. Thousand Oaks, CA: Sage Publications.

Guimaraes, Tor, Magid Igbaria, and Ming-te Lu. 1992. The determinants of DSS success: an integrated model. *Decision Sciences* 23 (2):409-430.

Guimaraes, Tor, D. Sandy Staples, and James D. Mckeen. 2003. Empirically testing some main user-related factors for systems development quality. *Quality Management Journal* 10 (4):39-54.

Hahn, Robert W., ed. 2002. *Government policy toward open source software*. Washington: AEI-Brookings Joint Center for Regulatory Studies.

Hahsler, Michael. 2004. A quantitative study of the adoption of design patterns by open source software developers. In *Free/open source software development*, edited by S. Koch, 103-123. Hershey, PA: Idea Group.

Hamerly, Jim, Tom Paquin, and Susan Walton. 1999. Freeing the source: the story of Mozilla. In *Open sources: voices from the Open Source revolution*, edited by C. DiBona, S. Ockman and M. Stone, 197-207. Sebastopol, CA: O'Reilly.

Hang, Jiayin, Heidi Hohensohn, Klaus Mayr, and Thomas Wieland. 2004. Benefits and pitfalls of open source in commercial contexts. In *Free/open source software development*, edited by S. Koch, 222-241. Hershey, PA: Idea Group.

Hann, Il-Horn, Jeff Roberts, and Sandra Slaughter. 2004. Why developers participate in open source software projects: an empirical investigation. Paper read at 25th International Conference on Information Systems, 12-15 December, at Washington, DC.

Herbold, R. J. 1960. Quad I. *Communications of the ACM* 3 (2):74.

Hertel, Guido, Sven Niedner, and Stefanie Hermann. 2003. Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. *Research Policy* 32 (7):1159-1177.

Himanen, Pekka. 2001. *The hacker ethic: a radical approach to the philosophy of business*. New York: Random House.

Holck, Jesper, and Niels Jørgensen. 2004. Do not check in on red: control meets anarchy in two open source projects. In *Free/open source software development*, edited by S. Koch, 1-26. Hershey, PA: Idea Group.

Holstein, James A., and Jaber F. Gubrium. 2003. Active interviewing. In *Postmodern interviewing*, edited by J. F. Gubrium and J. A. Holstein, 67-80. Thousand Oaks, CA: Sage Publications.

Hopkins, Tim. 2002. Renovating the Collected Algorithms from ACM. *ACM Transactions on Mathematical Software* 28 (1):59-74.

Howison, James, and Kevin Crowston. 2004. The problems and perils of mining SourceForge. Paper read at Mining Software Repositories Workship, International Conference on Software Engineering, 25 May, at Edinburgh.

Hunt, Francis, and Paul Johnson. 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, James E. 1997. Effects of user participation on systems development: a longitudinal field experiment. *MIS Quarterly* 21 (4):359-388.

Igbaria, Magid, Johani IIvari, and Hazem Maragahh. 1995. Why do individuals use computer technology? A Finnish case study. *Information & Management* 29 (5):227-238.

Igbaria, Magid, Francis N. Pavri, and Sid L. Huff. 1989. Microcomputer applications: an empirical look at usage. *Information & Management* 16 (4):187-196.

Igbaria, Magid, and M. Tan. 1997. The consequences of information technology acceptance on subsequent individual performance. *Information & Management* 21 (3):113-121.

IIvari, Johani. 1997. User information satisfaction: a critical review. In *Encyclopedia of library and information science*, edited by A. Kent, H. Lancour, W. Z. Nasri and J. E. Daily, 341-364. New York: Marcel Dekker.

Iivari, Netta. 2004. Exploring the rhetoric on representing the user: discourses on user involvement in software development. Paper read at 25th International Conference on Information Systems, 12-15 December, at Washington, DC.

International Institute of Infonomics and Berlecon Research GmbH. 2002. Free/Libre and Open Source Software: Survey and Study: FLOSS Final report 2002 [cited 15 July 2005]. Available from http://www.flossproject.org/report/index.htm.

Ives, Blake, Margrethe H. Olson, and Jack J. Baroudi. 1983. The measurement of user information satisfaction. *Communications of the ACM* 26 (10):785-793.

Jensen, Chris, and Walt Scacchi. 2004. Collaboration, leadership, control, and conflict negotiation in the Netbeans.org community. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 48-52.

Jiang, James, Edward Chen, and Gary Klein. 2002. The importance of building a foundation for user involvement in information system projects. *Project Management Journal* 33 (1):20-26.

Kappelman, Leon A. 1995. Measuring user involvement: a diffusion of innovation perspective. *The Data Base for Advances in Information Systems* 26 (2 & 3):65-86.

Kavanagh, Paul. 2004. *Open source software: implementation and management*. Amsterdam: Elsevier Digital Press.

Kim, Sung, and Roger McHaney. 2000. Validation of the end-user computing satisfaction instrument in case tool environments. *The Journal of Computer Information Systems* 41 (1):49-55.

Knuth, Donald. 1992a. The errors of TeX (1989). In *Literate programming*, 243-291. Stanford, CA: Center for the Study of Language and Information. Original edition, *Software Practice and Experience* 19:607-685.

———. 1992b. The error log of TeX (1978-1991). In *Literate programming*, 293-339. Stanford, CA: Center for the Study of Language and Information.

Koch, Stefan, ed. 2004. *Free/open source software development*. Hershey, PA: Idea Group.

Krishnamurthy, Sandeep. 2002. Cave or community? An empirical examination of 100 mature open source projects. *First Monday* 7 (6).

Kuhn, Thomas S. 1970. *The structure of scientific revolutions*. 2nd. ed. Vol. 2. International encyclopedia of unified science; vol. II, no. 2, ed. O. Neurath. Chicago: The University of Chicago Press.

Lakhani, Karim R., and Eric von Hippel. 2003. How open source software works: “free” user-to-user assistance. *Research Policy* 32 (7):923-943.

Lakhani, Karim R., Bob Wolf, and Jeff Bates. 2002. Hacker survey (Release 0.3) 2002 [cited 28 January 2005]. Available from http://www.osdn.com/bcg/BCGHACKER-SURVEY.pdf.

Lawrence, Michael, and Graham Low. 1993. Exploring individual user satisfaction within user-led development. *MIS Quarterly* 17 (2):195-208.

Lazar, Jonathan, and Jenny Preece. 2003. Social considerations in online communities: usability, sociability, and success factors. In *Cognition in a digital world*, edited by H. van Oosgendorp, 127-151. Mahwah, NJ: Lawrence Erlbaum Associates.

Lee, Jen-Fang, and Tzu-Ying Chan. 2004. Organizational structure of “user collaboration community”: insights from the case of an open source software project. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 58-63.

Lerner, Josh, and Jean Tirole. 2002. Some simple economics of open source. *The Journal of Industrial Economics* 1 (2):197-234.

Maass, Wolfgang. 2004. Inside an open source software community: empirical analysis on individual and group level. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 65-70.

Madey, Greg, Vincent Freeh, and Renee Tynan. 2002. The open source software development phenomenon: an analysis based on social network theory. Paper read at 8th Americas Conference on Information Systems, 9-11 August, at Dallas, TX.

Madey, Gregory, Vincent Freeh, and Renee Tynan. 2004. Modeling the free/open source software community: a quantitative investigation. In *Free/open source software development*, edited by S. Koch, 203-220. Hershey, PA: Idea Group.

Mahmood, Mo Adam, Janice M. Burn, Leopoldo A. Gemoets, and Carmen Jacquez. 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, Chris, and Fiona Stewart. 2003. Internet interviewing. *In Postmodern interviewing*, edited by J. F. Gubrium and J. A. Holstein, 81-105. Thousand Oaks, CA: Sage Publications.

Markham, Annette N. 2004. Internet communication as a tool for qualitative research. In *Qualitative research: theory, method and practice*, 2nd ed., edited by D. Silverman, 95-124. London: Sage Publications.

Markus, M. Lynne, Brook Manville, and Carole E. Agres. 2000. What makes a virtual organization work? *Sloan Management Review* 42 (1):13-26.

McGill, Tanya, Valerie Hobbs, and Jane Klobas. 2003. User-developed applications and information systems success: a test of DeLone and McLean's model. *Information Resources Management Journal* 16 (1):24-45.

McKeen, James D., and Tor Guimaraes. 1997. Successful strategies for user participation in systems development. *Journal of Management Information Systems* 14 (2):133-150.

McKeen, James D., Tor Guimaraes, and James C. Wetherbe. 1994. The relationship between user participation and user satisfaction: an investigation of four contingency factors. *MIS Quarterly* 18 (4):427-451.

Melone, Nancy Paule. 1990. A theoretical assessment of the user-satisfaction construct in information systems research. *Management Science* 36 (1):76-91.

Michlmayr, Martin, and Benjamin Mako Hill. 2003. Quality and the reliance on individuals in Free Software projects. In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 105-109.

Mockus, Audris, Roy T. Fielding, and James D. Herbsleb. 2002. Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology* 11 (3):309-346.

Monga, Mattia. 2004. From bazaar to kibbutz: how freedom deals with coherence in the Debian project. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 71-75.

Monge, Peter R., Susan G. Bachman, James P. Dillard, and Eric Eisenberg. 1983. Communicator competence in the workplace: model testing and scale development. In *Communication yearbook 5*, edited by M. Burgoon, 505-527. New Brunswick, N.J.: International Communication Association.

Montazemi, Ali R., David A. Cameron, and Kalyan Moy Gupta. 1996. An empirical study of factors affecting software package selection. *Journal of Management Information Systems* 13 (1):89-105.

Moody, Glyn. 2002. *Rebel code: Linux and the open source revolution*. London: Penguin.

Moore, Gary C., and Izak Benbasat. 1991. Development of an instrument to measure the perceptions of adopting an information technology innovation. *Information Systems Research* 2 (3):192-222.

Munir, Kamal A., and Matthew Jones. 2004. Discontinuity and after: the social dynamics of technology evolution and dominance. *Organization Studies* 25 (4):561-581.

Nakakoji, Kumiyo, Yasuhiro Yamamoto, Hoshiyuki Nishinaka, Kouichi Kishida, and Yunwen Ye. 2002. Evolution patterns of open-source software systems and communities. Paper read at Workshop on Principles of Software Evolution, May 2002, at Orlando, Florida.

Napier-Tyree, Stephanie. 2002. OCLC SiteSearch java source code available for download [email message], 19 March.

Narduzzo, Alessandro, and Alessandro Rossi. 2004. The role of modularity in free/open source software development. In *Free/open source software development*, edited by S. Koch, 84-102. Hershey, PA: Idea Group.

O'Mahony, Siobhán. 2003. Guarding the commons: how community managed software projects protect their work. *Research Policy* 32 (7):1217-1241.

Oh, Wonseok, and Sangyong Jeon. 2004. Membership dynamics and network stability in the open-source community: the Ising perspective. Paper read at 25th International Conference on Information Systems, 12-15 December, at Washington, DC.

Olson, Margrethe H., and Blake Ives. 1980. Measuring user involvement in information system development. Paper read at International Conference on Information Systems, 8-10 December, at Philadelphia, PA.

———. 1981. User involvement in system design: an empirical test of alternative approaches. *Information & Management* 4 (4):183-195.

Open Source Initiative. 2005. The open source definition version 1.9 2002 [cited 25 January 2005]. Available from http://www.opensource.org/docs/definition.php.

———. 2005. The approved licenses 2005 [cited 25 January 2005]. Available from http://www.opensource.org/licenses/index.php.

Østerlie, Thomas. 2004. In the network: distributed control in Gentoo Linux. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 76-81.

Pace, Andrew K. 2004. Technically speaking: starting off on a technicality. *American Libraries* 35 (4):92-93.

Palvia, Prashant C. 1996. A model and instrument for measuring small business user satisfaction with information technology. *Information & Management* 31 (3):151-163.

Palvia, Prashant C., and Shailendra C. Palvia. 1999. An examination of the IT satisfaction of small-business users. *Information & Management* 35 (3):127-137.

Pavlicek, Russell C. 2000. *Embracing insanity: open source software development*. Indianapolis, IN: Sams.

Preece, Jenny. 2000. *Online communities: designing usability, supporting sociability.* Chichester: John Wiley and Sons.

Raymond, Eric S. 2001. *The cathedral and the bazaar: musings on Linux and open source by an accidental revolutionary.* Rev. ed. Sebastopol, CA: O'Reilly.

———. 2001CatB. The cathedral and the bazaar. In *The cathedral and the bazaar: musings on Linux and open source by an accidental revolutionary,* Rev. ed., 19-63. Sebastopol, CA: O'Reilly.

Rizzo, John R., Robert J. House, and Sidney I. Lirtzman. 1993. Role conflict and ambiguity. In *Handbook of marketing scales: multi-item measures for marketing and consumer behavior research*, edited by W. O. Bearden, R. G. Netemeyer and M. F. Mobley, 283-285. Newbury Park, CA: Sage Publications.

Robles-Martínez, Gregorio, Jesús M. González-Baharona, José Centeno-González, Vicente Matellán-Olivera, and Luis Rodero-Merino. 2003. Studying the evolution of libre software projects using publicly available data. In T*aking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003,* edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 111-115.

Sagers, Glen W. 2004. The influence of network governance factors on success in open source software development projects. Paper read at 25th International Conference on Information Systems, 12-15 December, at Washington, DC.

Samoladas, Ioannis, Ioannis Stamelos, Lefteris Angelis, and Apostolos Oikonomou. 2004. Open source software should strive for even greater code maintainability. *Communications of the ACM* 47 (10):83-87.

Sandred, Jan. 2001. *Managing open source projects: a Wiley tech brief*. New York: John Wiley.

Scacchi, Walt. 2003. Issues and experiences in modeling open source software development processes. In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 121-125.

———. 2004. Free and open source development practices in the game community. *IEEE Software* 21 (1):59-66.

Schlumpf, Peter. 1999. Open source library systems. *Library Computing* 18 (4):323-326.

Schweik, Charles, and Andrei Semenov. 2003. The institutional design of open source programming: implications for addressing complex public policy and management problems. *First Monday* 8 (1).

Seddon, Peter B., Sandy Staples, Ravi Patnayakuni, and Matthew Bowtell. 1999. Dimensions of information systems success. *Communications of the Association for Information Systems* 2 (3).

Seddon, Peter, and Min-Yen Kiew. 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, Peter, and Siew-Kee Yip. 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.

Shaik, Maha, and Tony Cornford. 2003. Version management tools: CVS to BK in Linux Kernel. In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003,* edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 127-131.

———. 2004. Version control tools: a collaborative vehicle for learning in F/OS. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004, Edinburgh, Scotland*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 87-91.

Smith, Jonathan M., Michael B. Greenwald, Sotiris Ioannidis, Angelos D. Keromytis, Ben Laurie, Douglas Maughan, Dale Rahn, and Jason Wright. 2004. Experiences enhancing open source security in the POSSE project. In *Free/open source software development*, edited by S. Koch, 242-257. Hershey, PA: Idea Group.

Smith, L. Wheaton. 1961. What is proprietary in mathematical programming?—impressions of a panel discussion. *Communications of the ACM* 4 (12):542, 594.

Somers, Toni M., Klara Nelson, and Jahangir Karimi. 2003. Confirmatory factor analysis of the end-user computing satisfaction instrument: replication within an ERP domain. *Decision Sciences* 34 (3):595-621.

St. Laurent, Andrew M. 2004. *Understanding open source & free software licensing.* Sebastopol, CA: O'Reilly.

Stallman, Richard. 2002. Free software reality v. perception: letter to the editor. *Communications of the ACM* 45 (7):11-12.

Stewart, Katherine, and Sanjay Gosain. 2001. An exploratory study of ideology and trust in open source development groups. Paper read at 22nd International Conference on Information Systems, 16-19 December, at New Orleans, LA.

Stewart, Katherine J. 2004. OSS project success: from internal dynamics to external impact. In *Collaboration, Conflict and Control: Proceedings of the The 4th Workshop on Open Source Software Engineering, May 25 2004*, Edinburgh, Scotland, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 92-96.

Stewart, Katherine J., and Tony Ammeter. 2002. An exploratory study of factors influencing the level of vitality and popularity of open source projects. Paper read at 23rd International Conference on Information Systems, 15-18 December, at Barcelona, Spain.

Tait, Peter, and Iris Vessey. 1988. The effect of user involvement on system success: a contingency approach. *MIS Quarterly* 12 (1):91-108.

Teenan, Brian. 2005. Roles & responsibilities 2005 [cited 26 June 2005]. Available from http://www.mamboserver.com/index.php?option=com_content&task=view&id=50&Itemid=86.

Terry, Julian, and Craig Standing. 2004. The value of user participation in e-commerce system development. *Informing Science Journal* 7:31-45.

Thomas, Craig. 2003. Improving the verification, validation, and test of the Linux Kernel: the Linux Stabilization Project. In *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering, Portland, Oregon, 3 May 2003*, edited by J. Feller, B. Fitzgerald, S. Hissam and K. R. Lakhani, 133-136.

Thong, James Y.L., and Chee-Sing Yap. 1996. Information systems effectiveness: a user satisfaction approach. *Information Processing & Management* 32 (5):601-610.

Torkzadeh, Gholamreza, and Jungwoo Lee. 2003. Measures of perceived end-user computing skills. *Information & Management* 40:607-615.

Torvalds, Linus. 1991. Free minix-like kernel sources for 386-AT [comp.os.minix newsgroup], 5 October 1991 [cited 6 October 2003].

Torvalds, Linus, and David Diamond. 2002. *Just for fun: the story of an accidental revolutionary.* New York: HarperBusiness.

Venkatesh, Viswanath, and Fred D. Davis. 2000. A theoretical extension of the Technology Acceptance Model: four longitudinal field studies. *Management Science* 46 (2):186-204.

Venkatesh, Viswanath, Michael G. Morris, Gordon B. Davis, and Fred D. Davis. 2003. User acceptance of information technology: toward a unified view. *MIS Quarterly* 27 (3):425-478.

Vinson, Norman, and Janice Singer. 2001. Getting to the source of ethical issues. *Empirical Software Engineering* 6 (4):293-297.

von Krogh, Georg, Sebastian Spaeth, and Karim R. Lakhani. 2003. Community, joining, and specialization in open source software innovation: a case study. *Research Policy* 32 (7):1217-1241.

von Krogh, Georg, and Eric von Hippel. 2003. Editorial: Special issue on open source software development. *Research Policy* 32 (7):1149-1157.

Weber, Steven. 2004. *The success of open source*. Cambridge, MA: Harvard University Press.

Williams, Sam. 2002. *Free as in freedom: Richard Stallman's crusade for free software*. Sebastopol, CA: O'Reilly.

Witten, Ian H. 2003. Examples of practical digital libraries: collections built internationally using Greenstone. *D-Lib Magazine* 9 (3).

Woodroof, Jon, and William Burg. 2003. Satisfaction/dissatisfaction: are users predisposed? *Information & Management* 40 (4):317-324.

Ye, Yunwen, Kumiyo Nakakoji, Yasuhiro Yamamoto, and Kouichi Kishida. 2004. The co-evolution of systems and communities in free and open source software development. In *Free/open source software development*, edited by S. Koch, 59-82. Hershey, PA: Idea Group.

Yoon, Youngohc, and Tor Guimaraes. 1995. Assessing expert systems impact on users' jobs. *Journal of Management Information Systems* 12 (1):225-249.

Zeitlyn, David. 2003. Research note: Gift economies in the development of open source software: anthropological reflections. *Research Policy* 32 (7):1287-1291.

Zhao, Luyin, and Fadi P. Deek. 2004. User collaboration in open source software development. *Electronic Markets* 14 (2):89-103.

Zhao, Luyin, and Sebastian Elbaum. 2003. Quality assurance under the open source development model. *The Journal of Systems and Software* 66 (1):65-75.

Zviran, Moshe, and Zippy Erlich. 2003. Measuring IS user satisfaction: review and implications. *Communications of the Association for Information Systems* 12.