Free Software: A Case Study of Software Development in a Virtual Organizational Culture

Margaret S. Elliott
Institute for Software Research
University of California, Irvine
Irvine, CA 92697
949 824-7202
melliott@ics.uci.edu

Walt Scacchi
Institute for Software Research
University of California, Irvine
Irvine, CA 92697
949 824-4130
wscacchi@ics.uci.edu

April 2003

Abstract

This study is part of an ongoing comparative study of various types of open software communities including both free and open source software projects. This study examines how the organizational cultural beliefs and values of a free software virtual organization influence software development processes. It provides examples that illustrate the importance of personal motivation and a sense of working as a team in the perpetuation of a virtual work community. It presents the world of the GNUenterprise.org project as a virtual organizational culture that embodies the beliefs of free software and freedom of choice, and the values of community building and cooperative work. A close study of this project shows how these beliefs and values are manifested in software development methods, artifacts, and tool choice, as well as how dispersed developers cooperate and resolve conflict in a virtual community. Data collection includes the content analysis of Internet Relay Chat archives; kernel cousins archives (summary digests of IRC and mailing list archives); mailing list archives; email interviews; Web site documents and observations; and personal interviews conducted at two open source conferences. Two cases from IRC and mailing list archives of the GNUe virtual community at work are presented for in-depth analyses and comparison. Cultural beliefs and values combined with motivations directly influence the processes of free software development. Results show evidence of consensus-building and consistency across practices and artifacts. In addition, the beliefs and values are consistent with each other – all working in concert to form the ideology that promotes and perpetuates the free software movement and its many communities.

Free Software: A Case Study of Software Development in a Virtual Organizational Culture

Margaret S. Elliott
Institute for Software Research
University of California, Irvine
Irvine, CA 92697
949 824-7202
melliott@ics.uci.edu

Walt Scacchi
Institute for Software Research
University of California, Irvine
Irvine, CA 92697
949 824-4130
wscacchi@ics.uci.edu

1 Introduction

Open source software development (OSSD) projects are growing at a rapid rate. The SourceForge Web site estimates 500,000+ users with 700 new ones joining every day and a total of 50,000+ projects with 60 new ones added each day. Thousands of OSSD projects have emerged within the past few years (DiBona *et al.*, 1999; Pavlicek, 2000) leading to the formation of globally dispersed virtual communities (Kollock and Smith, 1999). Examples of open software projects are found in the social worlds that surround computer game development; X-ray astronomy and deep space imaging; academic software design research; business software development; and Internet/Web infrastructure development (Elliott, 2003;Elliott and Scacchi, 2002; Elliott and Scacchi, 2004; Scacchi, 2002a, 2002b, 2002c). In communities such as these, OSS developers work as peers relying on Web-based computing environments to support and coordinate their development work in decentralized settings. Working together in a virtual community in non-collocated environments, OSS developers communicate and collaborate using a wide range of web-based tools including Internet Relay Chat (IRC) for instant messaging, CVS for concurrent version control (Fogel, 1999), electronic mailing lists, and more (Scacchi, 2002c).

Proponents of OSSD hail advantages such as improved software validity, simplification of collaboration, and reduced software acquisition costs. However, few empirical studies have been conducted to validate or explore claims like these (e.g., Mockus *et al.*, 2000, 2002). Research has focused on the quantitative side of OSSD projects, such as aspects of developer defect density, core team size, and other variables (Koch and Schneider, 2000; Mockus *et al.*, 2000, 2002). Few researchers have gone beyond the quantitative approach to focus on open software projects as social phenomena (Mackenzie *et al.*, 2002). For example, it is inconclusive how quantitative studies might reveal answers to questions such as: How do people working in disparate, virtual organizations organize themselves so that the work is completed? What social arrangements arise that facilitate

the mitigation and resolution of conflict? How does the work culture of a virtual community influence OSSD? More studies are needed using a socio-technical perspective to develop empirically grounded understandings of the social circumstances surrounding the technical system configurations and virtual organizational contexts that comprise an open source project (Elliott, 2003; Elliott and Scacchi, 2002; Elliott and Scacchi, 2004; Scacchi, 2002a; 2002b; 2002c).

Open source projects follow a different trajectory for software development than closed source projects. Open source developers (Dibona, *et al.*, 1999; Williams, 2002) work in globally dispersed virtual communities with few face-to-face meetings, utilize informal requirements gathering (Scacchi, 2002c), and practice software development techniques that veer from typical software development practices in closed source environments (Kotoyna and Sommerville, 1998). This paper presents results of a study of the culture of a free software project whose goal is to develop a free version of an enterprise resource planning (ERP) system. We show how the work culture of a free software development project influences software development practices and how conflict is mitigated and resolved using computer-mediated communication (CMC) in a virtual organization.

This study is part of an ongoing comparative study of various types of open software communities (Elliott, 2003; Elliott and Scacchi, 2002; Elliott and Scacchi, 2004; Scacchi, 2002a, 2002b, 2002c) including both free and open source software projects. It is important to distinguish between the terms free software (Stallman, 1999) and open source (DiBona *et al.*, 1999). Free software refers to software that is open to anyone to copy, study, modify, and redistribute (Stallman, 1999b). The Free Software Foundation (FSF), founded by Richard M. Stallman (known as RMS in open source communities) (Williams, 2002) in the 1970s, advocates the use of its GNU General Public License (GPL) as a copyright license which creates and promotes freedom. The FSF is at the forefront of the free software movement, based on the concept of source code being fundamental to the furthering of computer sciece and that free source code is necessary for innovation to flourish in computer science (DiBona *et al.*, 1999). For more information on the FSF, see (http://www.gnu.org). A popular term heard in the free software community is "Think free speech, not free beer". It is used to emphasize the importance of the defense of freedom, not just the ideal of promoting software that is free of cost.

The term open source was coined by a group of people concerned that the term "free software" was anathema to businesses. This resulted in the formation of the Open Source Initiative (OSI), a non-profit corporation dedicated to managing and promoting the Open Source Definition for the good of the community (http://www.osi.org). The major difference between free software movement and the OSI is in the licensing requirements. The OSI promotes more liberties with open source licensing than the FSF. For example, the OSI supports licenses that accept combinations of open source software with proprietary software while the FSF promotes the use of the GPL, which requires software to be redistributed as free software exclusively.

The free software movement has spawned a number of free software projects in which software developers advocate and follow the principle of creating and using free software exclusively. They follow the principles of RMS whose philosophy emphasizes the moral imperative to produce free software and the immoral action of creating non-free software. In this study, RMS is considered the founder of a virtual organizational culture with subcultures forming within each free software project sharing his beliefs and values.

Popular literature has described open source developers as members of a "geek" culture (Pavlicek, 2000) notorious for nerdy, technically savvy, yet socially inept people, and as participants in a "gift" culture (Raymond, 2001) where social status is measured by what you give away. However, no empirical research has been conducted to study open software developers as virtual organizational cultures (Martin, 2002, Schein, 1992) with beliefs and values that influence their work. Researchers have theorized the application of a cultural perspective to understand IT implementation and use (Avison and Myers, 1995) but few have applied this to the workplace itself (Dube´ and Robey, 1999; Elliott, 2000). In this paper, we present findings from an ethnographic study of the work culture of a virtual organization whose purpose is to develop and maintain a free software system to support business applications.

A free software development community known as GNU (GNU's Not Unix) Enterprise, or more simply, GNUe, is the research site. GNUe is a meta-project of the GNU (http://www.gnu.org) Project, designed to collect Enterprise software in one location on the web. The plans are for GNUe to consist of three items:

- a set of tools that provide a development framework for enterprise information technology professionals to create or customize applications and share them across organizations;
- a set of packages written using the set of tools to implement a full Enterprise Resource Planning (ERP) system; and
- a general community of support and resources for developers writing applications using GNUe tools.

As with typical organizations (Martin, 1992, Schein, 1992), virtual organizations develop work cultures, which have an impact on how the work is completed. As with typical business organizations with a founder who leads the organization's culture (Schein, 1991), the free software movement, with RMS as its founder, has inspired the creation of virtual organizations with cultural beliefs and values of free software development manifested into work practices. In this paper, we present the results of a qualitative study of the GNUe community using grounded theory (Glaser and Strauss, 1967).

We present the GNUe world as a virtual organizational culture (Martin, 2002; Schein, 1992) that embodies the beliefs of free software and freedom of choice, and the values of community building and cooperative work. We show how these beliefs and values are manifested in software development methods, artifacts, and tool choice, as well as how dispersed developers cooperate and resolve conflict in a virtual community. Data collection includes the content analysis of IRC archives; kernel cousins archives (summary digests of IRC and mailing list archives); mailing list archives; email interviews; Web site documents and observations; and personal interviews conducted at two open source conferences. Two cases from IRC and mailing list archives of the GNUe virtual community at work are presented:

- a newcomer who criticizes the choice of a non-free graphics tool to create a Web site screenshot and causes a debate over tool choice; and

- a group of insiders (frequent contributors) debate the issue of using non-free software to develop documentation.

Conclusions from this study indicate that the recording and archiving of GNUe IRCs contribute to the mitigation and resolution of conflict while, at the same time, contributes to the persistence and renewal of cultural beliefs and values. We show that text-based computer mediated communication (CMC) in the form of IRC, kernel cousins, and mailing lists enhances management and resolution of conflict in virtual communities and that strong organizational cultural beliefs aid in conflict management and resolution in a virtual community.

In Section 2 we discuss free/open source software development followed by a discussion of the GNUe software project in Section 3. Section 4 presents background information on conflict management research in virtual communities, Section 5 discusses the organizational culture perspective, and Section 6 presents research methods used in the GNUe study. Section 7 describes the observed variables and codings used in the data analysis for this study, followed by a presentation of the data in Sections 8 and 9, for Case One and Case Two respectively. Section 10 includes a comparison of the two cases, Section 11 discusses the implications for CSCW, and finally Section 12 presents the conclusions.

2 Free/Open Source Software Development

This section discusses a brief historical account of the formation of the free software movement and FSF followed by details regarding free/open source software development practices, products, and environments.

2.1 Free Software Movement

The free software movement was envisioned by RMS in 1984 when he began work on GNU software with the intention of sharing it with others as free software. RMS is a key figure in the free software movement. In 1984, he started the GNU (GNU's Not Unix) project by developing and distributing a UNIX-like operating system as free software. This system has evolved into the GNU/Linux system using the Linux kernel combined with GNU utilities. The GNU project led to the formation of the FSF in 1985. The FSF is a tax-exempt charity whose purpose is to promote computer users' right to use, copy, modify, and redistribute computer programs. The FSF is dedicated to furthering the principles of free software with the goal of eliminating altogether the need to use proprietary systems and programs. The free software movement has evolved from the beliefs of the FSF and is based on the concept of source code being fundamental to the furthering of computer science and that free source code is necessary for innovation to flourish in computer science (DiBona *et al.*, 1999). For more information on the FSF, see (http://www.gnu.org).

2.2 What is Free/Open Source Software Development?

OSSD represents a relatively new approach to the development of complex software systems (Feller and Fitzgerald, 2002). OSSD generally relies on a global community of software developers and users who seek faster, better, and cheaper alternatives to closed proprietary systems. In most OSSD situations, the resulting software system and its associated Web-based documents or development artifacts are globally accessible at little or no direct cost. Furthermore, a defining requirement of OSSD is how their intellectual property rights are assigned and protected. The terms and conditions

of the copyright or license associated with OSS typically assert the following kinds of property rights or "freedoms" to anyone who seeks to employ or use the software (DiBona *et al.*, 1999; Pavlicek 2000; Williams 2002):

- Freedom to run the program for any purpose;
- Freedom to study how the program works and adapt it to their needs;
- Freedom to redistribute copies of the software at will;
- Freedom to improve the OSS program and to distribute the altered version;
- Required distribution of the originating license that specifies the freedoms and rights concerning the preceding properties.

These rights or freedoms do not prohibit charging fees to access or acquire OSS, though typically there are no direct cost for the OSS itself. Instead there may be costs associated with acquiring the media (e.g., CDROM, books) through which the OSS is distributed. Similarly, the rights or freedoms do not restrict who may provide support, system integration, or consulting services for the OSS. This is in contrast to the strictly controlled provision of support or services offered for closed source, proprietary software systems. These rights and freedoms stand in marked contrast to those offered with the selection, customization, and deployment of commercial software.

2.2.1 Free/Open Source Software Development Practices

Open source is not the same concept as "open systems". Open source is a broader, more encompassing technique for exposing access to the underlying functionality, operation, or interoperation of a software system. Open systems traditionally refer to a technology scheme that provides customers, external developers, or end-users to access the internal functions of a complex system via "public interfaces." In OSS, the source code, as well as its surrounding documents and artifacts, all serve as the public interface to the system. Access to system functionality is not limited to functions calls through "application programming interfaces" (APIs). Access to functionality, as well as the ability to enhance, restructure, tune, debug, or re-host system functionality is realized through access to open source code, documents, and artifacts. An open system may consist of hardware, software, and network system components. Subsequently, the potential exists for making all components functionality accessible, transparent, and open through public interfaces that consist of the components "source code", documents, and artifacts.

2.2.2 Open Source Software Products

OSS program code is the typical focus of most OSSD activities. These computer programs are written in a programming language like C, C++, Perl, Python, and others. Documents that specify or describe how these programs function or interoperate are also products of open source software development. These documents may include specification or design diagrams, end-user manuals, program installation scripts, threaded email discussion forums, Web-based source code repositories, and other Web site contents. OSSD projects rely on a diversity of software *informalisms* (Scacchi, 2002c) as information resources, documents, artifacts, or products that can be browsed, cross-linked, and updated on demand. These informalisms are socially lightweight information structures for managing, communicating, and coordinating globally dispersed knowledge about who did what, why, and how. These informalisms are easy to learn

and use as semi-structured representations that capture software requirements, system design, and design rationale. As OSS developers are themselves end-users of their systems, then software system requirements and design take less time to articulate and negotiate, compared to software engineering projects that must elicit requirements and validate system design with end-users who are generally not software system specialists.

OSSD projects are iteratively developed, incrementally released, reviewed and refined by OSS developers working as peers in an ongoing agile manner. These methods ensure acceptable levels of quality, coherence, and security of system-wide software via continuous distributed peer review, testing and profiling. OSSD efforts are hosted within decentralized communities of peers (Kogut and Metiu, 2001; Scacchi, 2002a, 2002b,2003c; Sharman, 2002) that are interconnected via Web sites and OSS repositories. Community oriented OSSD gives rise to new kinds of requirements for community building, community software, and community information sharing systems (Web site and interlinked communication channels for email, forums, and chat). In contrast, most system engineering projects associated with major software development efforts are targeted for a centralized corporate setting, where access and visibility may be restricted to local participants. OSSD standards (Freericks, 2001) are apparently easier to access and follow due to their Web-based deployment, and a long history of community oriented participation in developing implementation-oriented standards in an open source manner. These compare favorably to the institutionally oriented processes used to develop software engineering standards that are much more cumbersome and often less effective at ensuring system quality (Scacchi, 2002b).

2.2.3 Open Source Support Environments

OSS emerges from the efforts of developers who are distributed across space and time. They do not work in a single or central workplace, and often there is no formal management hierarchy in place to schedule, plan, and coordinate who does what, with what resources, etc. Instead open source developers contribute their effort to projects that they find interesting, significant, or otherwise professionally compelling. OSS developers generally have regular paid jobs, though they may or may not be paid to work on an open source project. Thus, traditional organizational models for how to motivate employees or how to organize and manage technical staff may not apply. Nonetheless, open source development projects thrive, as it now appears that tens of thousands of OSSD projects are underway.

OSSD projects are "organized" as a loosely knit community of interested developers and end-users who work and interact online via Web-based computing environments (Scacchi, 2002b). These environments provide access to a global information infrastructure that includes routine support for Email and electronic bulletin board, and Web sites for posting or sharing open source artifacts. They also provide public access to centrally administered multi-version source code directories, software extension schemes and mechanisms (e.g., multi-application scripting languages, like Perl and Python, to enable interoperating systems), and more (Scacchi, 2002b). Developing trust, "geek fame", and being recognized by peers for technical contributions (Pavlicek, 2000) are part of the "glue" that binds open source developers together with their global information infrastructure to create the productive units or virtual organizations (Noll and Scacchi, 1999) that populate the world of OSSD.

OSSD tools are inexpensive/free and comparatively easy to use and learn. They are both given and received as public goods or gifts to the community (Bergquist and Ljungberg, 2001). The most widely used OSS tools support concurrent version control and repository management (Fogel,1999), Web servers and browsers, communication applications (threaded email discussion forums, instant messaging), bug/issue reporting and resolution tracking, and various code development tools (text editors, integrated development environments, etc.). Access to and availability of OSS tools is generally not a problem or barrier to participation in an OSSD project.

Faster and better OSSD conditions tend to drive down the cost of developing software in terms of schedule and budget resources. Most OSSD projects are voluntarily staffed by people who want to work on the project, who will potentially commit their own time and effort, and who find personal and professional benefit from the OSSD development efforts (Scacchi,2002a). Minimal management or governance forms (Sharman *et al.,* 2002) are used to direct OSSD efforts, compared to the more rigid hierarchically managed, planned, staffed, controlled, and budgeted project activities typical for traditional development efforts in corporations. We show in this paper the effects of informal management on the GNUe software development efforts.

3 GNU Enterprise Software Project

The research site is a free software development community, the GNU Enterprise (GNUe) (http://www.gnuenterprise.org). GNUe is a meta-project of the GNU (http://www.gnu.org) Project. GNUe is designed to collect Enterprise software in one location on the web. The plans are for GNUe to consist of three items:

1) a set of tools that provide a development framework for enterprise information technology professionals to create or customize applications and share them across organizations;
2) a set of packages written using the set of tools to implement a full Enterprise Resource Planning (ERP) system; and
3) a general community of support and resources for developers writing applications using GNUe Tools. The GNUe website advertises it as a "Free Software project with a corps of volunteer developers around the world working on GNUe projects. This provides the added benefits of easy internationalization of applications. The project is working to provide a worldwide GNUe community, allowing everyone who is involved in the project access to talented business information technology professionals."

GNUe is an international virtual organization for software development (Crwoson and Scozzi, 2002; Noll and Scaccni, 1999) based in the U.S. and Europe. This organization is centered about the GNUe Web portal and global Internet infrastructure that enables remote access and collaboration. Developing the GNUe software occurs through the portal, which serves as a global information sharing workplace and collaborative software development environment. Its paid participants are sponsored by one or more of twelve companies spread across the U.S. and Europe. These companies provide salaried personnel, computing resources, and infrastructure that support this organization. However, many project participants support their participation

through other means. In addition, there are also dozens of unpaid volunteers who make occasional contributions to the development, review, deployment, and ongoing support of this organization, and its software products and services. Finally, there are untold numbers of "free riders" who will simply download, browse, use, evaluate, deploy, or modify the GNUe software with little/no effort to contribute back to the GNUe community (Olson, 1971).

As of the writing of this paper, GNUe contributors consist of 6 core maintainers (co-maintainers who head the project); 18 active contributors; and 18 inactive contributors. Companies from Austria, Argentina, Lithuania, and New Zealand support paid contributors but most of the contributors are working as non-paid participants.

GNUe is a community-oriented project, as are many OSSD efforts (Scacchi, 2002c; Sharman *et al.*, 2002). The project started in earnest in 2000 as the result of the merger of two smaller projects both seeking to develop a free software solution for business applications. More information and the history of the GNUe project can be found on their Web site. The target audience for the GNUe software modules are envisioned as primarily small businesses that are underserved by the industry leaders in ERP software, perhaps due to the high cost or high prices that can be commanded for commercial ERP system installations. Many of these target companies might also be in smaller countries that lack a major IT industry presence.

GNUe is a free software project affiliated with the FSF and the European FSF. The ERP and related software modules, and overall system architecture are called the GNUe software. All the GNUe software is protected using the GNU Public License (GPL) (DiBona *et al.*, 1999; Pavlicek, 2000; Williams, 2002). This stands in contrast to the ERP software from Compiere, which depends on the use of a commercial Oracle DBMS. Thus, GNUe is a free open source project, rather than simply an open source development project (Feller and Fitzgerald, 2002).

GNUe is not in business as a commercial enterprise that seeks to build products and/or offer services. GNUe is more of a pre-competitive alliance of companies and individuals that want to participate in the development, use, or evolution of free ERP software modules. As such, it has no direct competitors in the traditional business sense of market share, sales and distribution channels, and revenue streams.

Developers contributing to the ongoing evolution of the GNUe software in general provide their own personal computing resources. This is especially true for unpaid volunteer contributors, but also true of salaried participants who are paid to work on the GNUe software, particularly for their work at home. There is no standard or common personal computer configuration that is defined as the development platform, other than the requirement that a computer can run either Microsoft Windows or GNU/Linux operating systems, and that it can access the Internet or Web as needed. Thus, all GNUe community members must provide their own way into the project, via personal resource subsidies.

Beyond this, individual participants and contributors in the project are also expected to provide their own personal software development tools. These tools are generally expected to include those for source code development (e.g., code text editors, compiler collections, debugging tools,

document formatters, local file repositories) and communications (Email clients, Web browsers). In addition, software contributors routinely use the shared project coordination tools such as the CVS (Content Versioning System) software version repository manager (Fogel, 1999), Internet Relay Chat (IRC) for instant messaging and message logging, and emerging GNUe software modules. However, within the GNUe community, there is a strong, often reiterated belief that project contributors should only use software tools that are also free, open source software or that support non-proprietary data exchange formats, rather than proprietary, closed source products or data formats.

There are shared computing resources that help support and embody the GNUe effort. These include the Web servers that host the content, communications, and related software development artifacts associated with GNUe community. The hardware side of the Web servers and their Internet service connection and fees are provided by companies that sponsor the GNUe project. The project's Web servers include use of the Apache Web server and the PhP-Nuke content management system, which together provide Web portal capabilities while also serving as the community's information infrastructure (Deltor, 2000).

3.1 GNUe Software Development Tools

The GNUe project uses the CVS (Fogel, 1999), a client-server set of tools for maintaining source code for GNUe and keeping track of all changes to the source code. They also use Double Chocco Latte (DCL), a software package providing project management capabilities, time tracking on tasks, call tracking, email notifications, online documents, statistical reports, a report engine, and additional features to be developed in the future. DCL is a free software project which was merged into the overall GNUe project in March 2002. In addition, GNUe contributors use a variety of Linux distributions and computer platforms to develop and test GNUe software.

4 Conflict Management in Virtual Communities

Conflict is an integral part of cooperative work in many work settings (Easterbrook, 1993) and is inevitable in software development, especially in virtual organizations where assignments are loosely made, projects are managed informally, and where users are communicating from across the world in mainly text-based venues. Since computer-supported cooperative work (CSCW) research is concerned with the design of systems to support interactions between individuals or groups, an analysis of conflict and its role in open source software development would be useful. Conflicts arise between people engaging in collaborative activities and CSCW should include an understanding of how collaboration may break down and how it can continue in the presence of conflict (Easterbrook *et al.*, 1993). Understanding how conflict is mitigated and resolved in open source and free software development communities is beneficial to CSCW researchers interested in developing open source support systems and for managers considering the initiation of open source software development in their organization.

Few researchers have attempted to understand conflict management in virtual communities (Smith, 1999). Smith (1999) studied conflict management in MicroMUSE, a game world dedicated to the simulation and learning about a space station orbiting the earth. There were two basic classes of participants: users and administrators. Disputes arose in each group and between the two groups

regarding issues like harassment, sexual harassment, assault, spying, theft, and spamming. These problems occurred due to the different meanings attributed to MicroMUSE by its players and administrators and due to the diverse values, goals, interests, and norms of the group. Smith concluded that virtual organizations have the same kinds of problems and opportunities brought by diversity as real organizations do, and that conflict is more likely, and more difficult to manage than in real communities. Factors contributing to this difficulty are: wide cultural diversity; disparate interests, needs and expectations; nature of electronic participation (anonymity, multiple avenues of entry, poor reliability of connections and so forth); text-based communications; and power asymmetry among users.

The term "conflict" has been used in many different ways (Easterbrook, *et al.*, 1993) in both general and specific ways. For purposes of this paper, we adopt a general definition of conflict as stated in (Easterbrook *et al.*, 1993):

> "...the interaction of interdependent people who perceive opposition of goals, aim, and values, and who see the other party as potentially interfering with the realization of these goals … (This) definition highlights three general characteristics of conflict: interaction, interdependence, and incompatible goals" (Putnam and Poole, 1987, p 552).

Easterbrook *et al.* (1993) refer to this definition as a phenomenon that may arise whether people are cooperating or not. They list a set of assertions with supporting theories and literature about conflict and cooperation in organizations. In this paper we discuss and further develop a subset of their assertions pertaining to virtual communities. Their assertions are grouped into the following categories: occurrence of conflict, causes of conflict, utility of conflict, development of conflict, management and resolution of conflict, and results of conflict. We have selected three assertions from two of those categories. We present them briefly here and analyze them in depth at the end of the paper in addition to adding an assertion derived from our research.

4.1 Causes of Conflict

This category refers to the particular causes of conflict that arise in group work. The following assertion is related to the potential for individuals to remain anonymous in email exchanges. This anonymity is also applicable to the free/open source communication. Our research challenges this conclusion.

4.1.1 Anonymity and physical separation contribute to conflict.

Sproul and Kiesler (1986) showed email reduces social context cues and hence people behave irresponsibly more often and focus on themselves rather than others in salutations and closings. Email creates a world with a lack of status and social cues, social anonymity and lack of a mature etiquette. Surprisingly, despite the drawbacks of anonymity and physical separation, in the GNUe community, people strive to cooperate and resolve conflict through the use of IRC and e-mail.

4.2 Management and Resolution of Conflict

This category involves assertions related to the management and resolution of conflict. Our data refutes the first one listed and supports the second.

4.2.1 Conflicts are unlikely to be resolved if participants argue from entrenched positions.

Easterbrook *et al.* (1993) argue that if participants become entrenched in their opinions, it becomes difficult to explore the middle ground and, in turn, resolve conflicts. Contrary to their view, our research suggests that free software development communities strive to cooperate when resolving conflicts despite sometimes conveying extreme positions regarding the sole use of free software for development.

4.2.2 Articulation of conflict helps in its resolution.

Research has shown that groups who discuss their work will perform significantly better in areas such as group cohesiveness or commitment to task and productivity than those who do not. Our results show that articulation of issues in an open manner using IRC and e-mail archives contributes to successful agreement among core GNUe contributors despite the amorphous nature of community membership.

4.3 Conflict Among "Geeks" in Open/Free Software Development Communities

The management and resolution of conflict is a key ingredient in an open/free software project. In order to understand how the organizational culture of GNUe influences software production, one needs to study the programmers, designers, and lurkers themselves as a community. Popular literature has evoked images of the culture of "geeks", a term used to describe OSS developers:

> "The geeks who write Open Source software comprise a community. They tend to value certain basic concepts. They often debate particular issues which are considered important, such as freedom, appropriate licensing, or technical toolkits....The geek culture is the core of the matter of understanding the Open Source movement. The very existence of geek culture may take some people by surprise. In the general world, geeks are often characterized as being antisocial individuals. Not only is that characterization inaccurate, it is absolute nonsense. Geeks are very social people, as we will discuss in detail in the next chapter. But their social interactions tend to follow the rules of geek culture much more than those of the society at large (p. 48, Pavlicek, 2000)."

Pavlicek (2000) outlines in detail the values ascribed to geek culture and how those values influence the quality of software development:

> "To understand the cultural priorities of the geek, you must keep in mind the appropriate perspective. You must be mindful of the geeks within the culture. Among the highest goals is the continued production of high-quality, Open Source software. It stands to reason, therefore, that the core values of the culture should support the things needed to accomplish that goal. It should not be surprising, then, that one of the key values for he community is truth...If someone fails to speak the truth, the process of creating software will be greatly impaired (Pavlicek, 2000)".

Research is needed to understand how this culture surrounding free software development persists and directs work in free software and open source projects. In the next section we describe the

organizational culture perspective and its application to virtual communtiies.

5 Organizational Culture Perspective – What is it?

Much like societal cultures have beliefs and values manifested in norms that form behavioral expectations, organizations have cultures that form and give members guidelines for "the way to do things around here." An organizational culture perspective (Martin, 2002; Schein, 1992; Trice and Beyer, 1993) provides a method of studying an organization's social processes often missed in a quantitative study of organizational variables. Organizational culture is a set of socially established structures of meaning that are accepted by its members (Ott, 1989). An organizational culture perspective looks at the nonrational aspects of an organization. If rational theories of organizations and management are utilized without attention to the cultural perspective, one can get misleading results because each rational theory tends to simplify the complexities and diversity of organizational life:

> "Cultural research tries to apprehend and analyze larger chunks of reality and preserve the context in which it occurs as an integral part of that reality. In effect, it tries to encompass more of the complexities and messiness of real life - including its nonrational aspects. Because of this inclusiveness, cultural research yields results that are rich, concrete, and interesting to scholars and practitioners alike. (Trice and Beyer, 1993, p. xiv)."

As in a societal culture, an organizational culture helps individuals and groups deal with uncertainties and ambiguities while offering some degree of order in social life. The substances of such cultures are formed from ideologies, the implicit sets of taken-for-granted beliefs, values, and norms. Ideologies are more emotionally charged and resistant to change than rational forms because they help people cope with uncertainties and because they form due to situations not expected by rational means. Members express the substance of their cultures through the use of cultural forms in organizations, acceptable ways of expressing and affirming their beliefs, values and norms. When beliefs, values, and norms coalesce over time to form stable forms that comprise an ideology, they provide causal models for explaining and justifying existing social systems.

Cultural forms in organizations can be characterized into four categories (Trice and Beyer, 1993) - symbols, language, narratives, and practices. Table 1 shows the categories and examples of cultural forms borrowed from Trice and Beyer (1993, p. 78).

Table 1 - Categories and Examples of Cultural Forms

Category	**Examples**
Symbols	Objects, natural and manufactured Settings Performers, functionaries
Language	Jargon, slang Gestures, signals, signs Songs Humor, jokes, gossip, rumors Metaphors Proverbs, slogans
Narratives	Stories, legends Sagas Myths
Practices	Rituals, taboos Rites, ceremonials

Organizational cultures, like other cultures, evolve as groups of people struggle together to make sense of and cope with their worlds (Trice and Beyer, 1993). It is through the interaction between ideologies and cultural forms that cultures maintain their existence. Cultural forms facilitate how people make sense of their world. The reality of the world people cope with becomes socially constructed (Berger and Luckmann, 1967). The actual activity of sense making involves several processes that are ordinarily treated as distinct (Trice and Beyer, 1993, p. 81):

> "Sense making is a cognitive process in that it involves knowing and perceiving, it is a behavioral process in that it involves doing things, and it is a social process in that it involves people doing things together."
>
> "Sense making can be at a nonconscious level or a conscious level. It becomes a more active enterprise when people need to cope with uncertainties in their social worlds. Novelties, discrepancies and requests for active thinking are three of the conditions likely to produce a switch from automatic to conscious sense making. Examples of novel situations are mergers, technological change, and organizational birth. (Trice and Beyer, 1993, p. 82)."

Sense making happens at both an individual and group level. Group level sense making is that which shapes culture through the shared views of the world.

The fact that cultural forms and ideologies persist long after the original people responsible for their inception is a sign that cultures are collective phenomenon, not just individual happenings. It is through the use of cultural forms such as symbols, rituals, language, rites, and work practices that organizational cultures make sense of their current situations by drawing on

previous experiences. Most organizational culture researchers view work culture as a consensus-making system (Ott, 1989; Trice and Beyer, 1993; Schein, 1993). However, some researchers view organizational culture as an emergent process (Martin, 1992; Martin, 2002; Smircich, 1983).

Meyerson and Martin (1987) base their perspective on treating culture as a metaphor for organization, not just as a discrete variable to be manipulated at will. Organizations are viewed as patterns of meaning, values, and behavior and their approach to organizational change involves changes in patterns of behavior, values, and meanings. The three paradigms recommended to explain cultural change are:

- **Integration**: Culture is defined as that which is shared by a given organization. Emphasis here is on leadership-oriented cultural change and/or on consistency and consensus among cultural members. Cultural change is viewed as an organization-wide process.
- **Differentiation**: Culture is viewed as resulting in inconsistencies, lack of consensus and non-leader centered sources of cultural content. Emphasis is on sub-groups, both groups and individuals and on the interaction of subcultures within an organization. Using an open-system perspective, culture is formed by influences from inside and outside the organization. Cultural change is emphasized as continual changes to subcultures and changes between subcultures and the dominant culture.
- **Ambiguity**: Unlike integration and differentiation, this paradigm welcomes ambiguity. Culture is viewed as having no shared values except one: the awareness of ambiguity. In this view, culture is continually changing.

Martin (1992) furthered this approach to analyzing cultural change by explaining it in more detail and applying it to empirical data. The term Ambiguity was changed to Fragmentation:

> "The Fragmentation perspective brings these sources of ambiguity to the foreground of a cultural description. Building on the complexities introduced by the nexus approach to understanding culture, Fragmentation studies see the boundaries of subcultures as permeable and fluctuating, in response to environmental changes in feeder cultures. The salience of particular subcultural memberships wax and wane, as issues surface, get resolved, or become forgotten in the flux of events. In this context, the manifestations of a culture must be multifaceted - their meanings hard to decipher and necessarily open to multiple interpretations. From the Fragmentation viewpoint, both the unity of Integration studies and the clearly defined differences of the Differentiation perspective seem to be myths of simplicity, order, and predictability, imposed on a socially constructed reality that is characterized by complexity, multiplicity, and flux (Martin, 1992, p. 132)."

Although this multi-paradigmatic approach to organizational culture analysis shows promise, only a few researchers have applied it to real-world problems (Dubé and Robey, 1999; Elliott, 2000; Martin, 1992; Meyerson and Martin, 1987). Dubé and Robey (1999) used this three-perspective approach (Martin, 1992) to interpret stories told to them by employees of a software

development company's management practices. Results from their study indicated the importance of understanding cultural foundations of management practices.
Meyerson and Martin (1987) also used the three paradigms for an analysis of the organizational culture of the Peace Corps/Africa during the Kennedy and Johnson administrations. They showed how each paradigm draws attention to a particular set of organizational concerns and at the same time ignores others. They recommend using more than one paradigm for an organizational analysis to avoid the blinders that a researcher is likely to use if only one cultural perspective is utilized. Martin (1992) expanded the theory and used it to analyze the culture of a Fortune 500 firm of over 80,000 people worldwide. This firm has been used frequently in cultural studies. By using the three paradigms, she showed cultural change can be viewed differently from several viewpoints than from using one perspective. Elliott (2000) used the three perspective approach to analyze the influence of organizational culture on the implementation and use of case management computer systems in the Los Angeles County criminal courts, and how the organizational culture altered the use of the systems.

In this study, we view organizational culture as an emergent phenomenon manifested in an organization's work practices, norms, artifacts and symbols. Table 2 lists typical cultural manifestations found in organizational culture studies (Martin, 2002).

Table 2 - Descriptions of Organizational Cultural Manifestations

Category	**Examples**
Cultural Artifacts	Rituals, Organizational Stories, Jargon, Humor, Physical Arrangements (architecture, interior décor, dress codes)
Formal Practices	Organizational structure (e.g. mechanistic or organic; hierarchical or flat), Task and technology, Rules and procedures (e.g. handbooks), and Financial controls (accounting, pay, budgeting, etc.).
Informal Practices	Norms and Social rules (not written down)
Content Themes	Cognitive (Beliefs or tacit assumptions) or Attitudinal (Values) that underlie interpretations of cultural manifestations

In this study, we analyze the connection between content themes and cultural manifestations in the GNUe community by using the matrix approach (Martin, 2002). The matrix approach to understanding culture aids in showing the interpretations of how cultural manifestations relate to each other, forming the pieces of a cultural puzzle. A matrix analysis is especially helpful in summarizing how a cultural study has been operationalized by a researcher and can also be used to compare cultural studies. It is a compliment to the detailed discussion of the analytical framework resulting from the cultural study. The columns in the matrix represent the cultural manifestations and the rows show the content themes. Content themes are the beliefs and values of a culture that combine to bind the members together and are enacted in cultural manifestations. The substance of a culture is its ideology – shared, interrelated sets of

emotionally charged beliefs, values and norms that bind people together and help them to make sense of their worlds (Trice and Beyer, 1993). While generally closely interrelated in behavior, beliefs, values, and norms are unique concepts as defined below (Trice and Beyer, 1993):

- **Beliefs** – Express cause and effect relations (i.e. behaviors lead to outcomes).
- **Values** – Express preferences for certain behaviors or for certain outcomes.
- **Norms** – Express which behaviors are expected by others and are culturally acceptable ways to attain outcomes.

Table 3 shows the GNUe matrix with empty cells. In this study, as in other studies, norms are considered part of informal work practices (Martin, 2002). A corresponding GNUe summary matrix with the cells defined is presented later in the Summary and Conclusions section. The observed variables and codings used to fill in the boxes are discussed in the section on Observed Variables and Codings. The meanings of the column headings are discussed below.

Table 3 – Sample Matrix for GNUe Organizational Culture Without Data

Content Themes		**Practices**		**Artifacts**
Espoused	**Inferred**	**Formal**	**Informal/Norms**	**Electronic Artifacts**
Belief in Free Software				
Belief in Freedom of Choice				
	Value in Community	.		
	Value in Cooperative Work			

Content themes. Content themes are common threads of concern that underlie interpretations of cultural manifestations used in a cultural study. They can be of the cognitive nature (beliefs) or be attitudinal (values). Content themes can also be espoused (as in a company's brochure) or inferred deductively by a researcher or interviewee. Espoused themes tend to be more abstract, used to attract potential employees or create a positive image of a company. For example, in the GNUe culture, the two espoused beliefs – belief in free software and belief in freedom of choice - are discussed in literature about open source (Pavlicek, 2000; Williams, 2002) and on the GNUe website(www.gnuenterprise.org). The two tacit themes – value in community and cooperative work – were derived from the data and are presented with supporting data later in the paper. Potential contributors for free software projects typically embrace the ideology of the free software movement and, consequently, their beliefs have great influence over their decision to contribute to a free software project. For this study, the content themes are analyzed across formal practices, informal practices, norms, and electronic artifacts.

Formal and InformalPractices. Formal and informal practices are an integral part of organizational research. Formal practices are written down such as in a company policy manual while informal practces evolve through interaction and are usually not written down (e.g. social rules). There are four different types of formal practices which have been of interest to cultural reserachers (Martin, 2002):

- **Structure** – This refers to the organizational structure of the organization such as mechanistic (detailed job descriptions) versus organic (loosely defined job roles or cross-functional teams, etc.). Structure may also include shape of a hierarchy (steep or flat), the criteria for differentiation, and the balance of devices for integration and differentiation. For the GNUe study, structure is considered from the organic model with a flat hierarchy.
- **Technology and Task** – This is related to what it takes to produce an organization's product or services. For example, in GNUe, we analyze the software development practices needed to produce their free software product.
- **Rules and Procedures** - Rules and procedures are often written down in elaborate company handbooks. GNUe's website serves this purpose.
- **Financial Controls** – These are the company's accounting policies and pay scales, largely absent from the free software business.

The informal work practices analyzed in organizational cultures often contrast with formal rules. For example, a company may espouse a policy of teamwork among employees, yet informally people work in isolated offices with individual assignments causing deep competition among employees. Cultural studies focus on one or both of these types of content themes. For this study, we focus on both formal and informal practices including norms.

Norms. In a typical organizational culture, norms deptict "the way to do things around here. They provide methods for doing things. In a virtual community, the norms are the expected social rules established via interaction on the Web (e.g. creating open or free software in a virtual community). For example, in the GNUe community, it is the norm to demonstrate open disclosure of all software development work and documentation.

Electronic Artifacts. Electronic artifacts are particularly important to a virtual organization whose purpose is to develop software. We consider these electronic artifacts such as IRC archives and kernel cousins (mailing list and IRC digests) to supplant the "physical" artifacts that one would normally find in a typical organization. For example, in a typical organizaiton an organizational culture study would include the acoutrements of employees' attire and offices, even the architecture of the building. In a virtual environment we rely on textual and graphical website contents and structure to illuminate the organizational culture.

Definition of organizational culture. Researchers have defined organizational culture in myriad ways (Martin, 2002). We treat culture as a metaphor for organization, not as a discrete variable within an organization. The virtual community *is* the organizational culture. In other words, the GNUe project group would not exist without the beliefs and values of the FSF facilitating its work purpose – to create a free ERP system. We present the GNUe virtual

organization as a subculture of the FSF inculcating the beliefs and values of the free software movement into their everyday work practices in free/open software development of GNUe. Since we are studying a virtual organization without formal ownership or management, many of the organizational culture concepts do not apply in a virtual work world (e.g. there is no boss requiring work to be completed in a specific way or within a certain time frame). We use the following organizational culture definition as the background of this study:

> "[Culture is] the pattern of shared beliefs and values that give members of an institution meaning and provide them with the rules for behavior in their organization" (Davis, 1984, p.1). (as listed in Martin, 2002).

The GNUe virtual community has a pattern of shared beliefs and values that give their work meaning and provide them with rules of behavior related to their online communication and software development. In this paper, we apply the Integration perspective (Martin, 2002) to the GNUe community showing how the beliefs and values of the free software movement tie the virtual organization together in the interests of completing the GNUe free software project. While the Integration perspective most closely characterizes the consensus-making aspects of the free software movement, the Differentiation and Fragmentation perspectives (Martin, 2002) will be applied to the GNUe organizational culture in our future research. In the next section we discuss previous studies of information technology and the organizational culture perspective.

5.1 Organizational Culture Perspective and Information Technology

Studies combining the organizational culture perspective with information technology (IT) are rare. Researchers have theorized the application of a cultural perspective to understand IT implementation and use (Avison and Myers, 1995; Cooper, 1994; Robey and Azevedo,1994; Scholz, 1990) but few have applied this to the workplace itself (Dubé and Robey, 1999; Elliott, 2000). Popular literature (Pavlicek, 2000) has characterized the "geek" culture prevalent in open source communities but research is needed in applying the organizational culture perspective to virtual worlds (Martin, 2002).

Researchers have characterized IT professionals as occupational subcultures. Occupational subcultures are based on people's occupations and form as diffuse subcultures in society and as subcultures within organizations. Schein (1992) has used the concept of occupational subcultures to analyze the occupation of IT designers and how they view potential users of systems they design. He characterizes the occupational subculture of IT designers within an organization as conflicting with management goals and as not relating to how end-users view technology when designing systems for them (i.e. not using user-centered design). When occupational subcultures become an integral part of everyday life in and out of the workplace, they can be viewed as occupational communities (Van Maanen and Barley, 1986). Gregory (1983) used the concept of occupational communities to depict IT professionals as similar in beliefs and values within the varying cultures of jobs held by technical professionals in a Silicon Valley engineering company (e.g. software engineers, hardware engineers, marketing engineers).

Other researchers suggest that valuable insights can be gained by viewing IT and organizational cultures from an anthropological approach (Avison and Meyer, 1995) in addition to disciplines

like information theory, semiology, organization theory, sociology, computer science, and engineering. Avison and Meyer (1995) point out three areas in which IS researchers have recognized the value in using anthropological concepts and methods to facilitate the description and analysis of the social world in which IS are developed and used:

1) Culture is a basic and central concept of cultural anthropology.
2) A related concept is symbolism that is widely used in anthropological circles to help interpret actions of social actors.
3) Research methods are another area of contribution to the discipline of IS. The method used in anthropology is ethnography, a fast growing application in IS research and design.

However, more work is needed for anthropological methods to make an impact.

Avison and Myers (1995) review IT and organizational culture in the literature. They show how the term "culture" has been used rather narrowly. Olson (1982) was one of the first IS researchers to refer to the relationship between IT and organizational culture. In her article, the term "organizational culture" is taken for granted, for nowhere is the concept explicitly defined. Rather she discusses how technology can affect organizational behavior, giving a narrow interpretation of organizational culture, much like that used in social psychology. Schein (1984)'s article on organizational culture may be one of the most influential on IS research. Schein defines organizational culture in the following way and this definition is used widely in the literature:

> "Organizational culture is the pattern of basic assumptions that a given group has invented, discovered, or developed in learning to cope with its problems of external adaptation and internal integration, and that have worked will enough to be considered valid, and, therefore, to be taught to new members as the correct way to perceive, think, and feel in relation to those problems (Schein, p. 3, 1984)."

The authors point out that Schein's definition, like Olsen's, is more like that used in social psychology. Schein argues that culture is something that identifies and differentiates a social group; in some organizations, culture is something that can be managed and changed. This definition is used throughout the IS and organizational literature. However, Avison and Meyer (1995) suggest that researchers adopt a more critical, anthropological view of the concept rather than characterizing culture in the social psychological vein. Often the culture concept is not clearly defined and discussed in a common-sense way. Even within anthropology itself, there is a dearth of consensus on what the concept really means (Smircich, 1983). Avison and Meyer (1995) conclude that culture is "seen as contestable, temporal and emergent, it is constantly interpreted and reinterpreted, and is produced and reproduced in social relations."

Avison and Meyer (1995) suggest a number of implications for IS researchers examining the relationship between IT and organizational culture:

- The prevailing taken-for-granted view of the culture concept within IS research community needs to be abandoned. The definition (Schein, 1984) that cultures are separate, distinct

entities that identify and distinguish one group from another is too simplistic. The authors suggest that cultures are contested, ever-changing, and emergent. They are invented and reinvented in social life. Much as anthropologists have studied traditional cultures in many countries, IS researchers should consider the ways in which people within organizations create and recreate meaning through the use of IT.

- Another oversimplification is the assumption that culture can be seen as a variable or set of variables. Culture is an ever-changing emergent phenomenon.
- Schein's suggestion that culture is something that can be managed and changed is shown to be difficult, if not impossible.

IS research would benefit by viewing culture as contested, temporal and emergent with the potential to offer rich insights into how new IT affect or mediate organizational and national cultures, and vice versa, how culture affects the adoption and use of IT. There is a need for more conceptual and empirical field research to explore these issues (Avison and Meyer, 1995). In this paper we present the virtual community of GNUe as an emergent organization with a rich culture based on strong consensus-building beliefs in free software.

6 Research Methods

This ongoing ethnography of a virtual organization (Hine, 2000; Olsson, 2000) is being conducted using the grounded theory approach (Glaser and Strauss, 1967; Strauss and Corbin, 1990) with participant-observer techniques. The sources of data include books and articles on OSSD, instant messaging (Herbsleb and Grinter, 2000, Nardi *et al.*, 2000) transcripts captured through IRC logs, threaded email discussion messages, and other Web-based artifacts associated with GNUe such as kernel cousins (summary digests of the IRC and mailing lists). This research also includes data from email and face-to-face interviews with GNUe contributors, and observations at Open Source conferences. The first author spent over 100 hours studying and perusing IRC archives and mailing list samples during open and axial coding phases of the grounded theory. During open coding the first case study presented here was selected as representative of the strong influence of cultural beliefs on GNUe software development practices. By using the kernel cousins, another example of an IRC involving conflict was located and, subsequently, added to the axial coding analysis. In this paper, we present the results of selective coding of these two examples. Future work includes the ongoing comparison of a GNUe no-conflict case with the two cases involving conflict. In this case, a newcomer comes onboard to contribute to GNUe without the debate issues of free versus non-free software. The data from the GNUe project will also be compared to other open software communities in the larger comparative study supporting this research. Note that throughout the discussion of the data, frequent contributors are referred to as insiders and newcomers or infrequent contributors as outsiders.

The initial research questions that formed the core of the grounded theory are:

1) How do people working in virtual organizations organize themselves such that work is completed?
2) What social processes facilitate open source software development?
3) What techniques are used in open source software development that differ from typical

software development?

We began this research with the characterization of open source software communities as communities of practice. A community of practice (COP) is a group of people who share similar goals, interests, beliefs, and value systems in a common domain of recurring activity or work (Wenger, 1998). This characterization is used by researchers to investigate groups of people working together and using information technology (IT) systems (Brown and Duguid, 1993; Star, 1996). COPs are informally bound together by shared expertise and passion for a joint enterprise. COPs are considered more tractable than cultures and may develop within the confines of larger contexts - historical, social, cultural, institutional (Wenger, 1998). An alternative way of viewing groups with shared goals in organizations is to characterize them as organizational subcultures (Trice and Beyer, 1993; Schein, 1992; Martin, 2002). As the grounded theory evolved, we discovered rich cultural beliefs and norms influencing "geek" behavior (Pavlicek, 2000). This led to us to the characterization of the COPs as virtual organizations having organizational cultures.

During the open coding, we interpreted books and documents as well as website descriptions of the OSSD process. We discovered strong cultural overtones in the readings and began searching for a site to apply an analysis of how motivations and cultural beliefs influenced the social process of OSSD. We selected GNUe as a research site because it exemplified the essence of free software development providing a rich picture of a virtual work community. The GNUe website offered access to downloadable IRC archives and mailing lists as well as lengthy documentation - all facilitating a virtual ethnography. During the axial coding phase of several IRC chat logs, mailing lists and other documentation, we discovered relationships between beliefs and values of the work culture and manifestations of the culture. In the next section we present the techniques used to apply the organizational culture perspective to the virtual organization of GNUe.

6.1 Organizational Culture Perspective Applied to GNUe

We view culture as both objectively and subjectively constrained (Martin, 2002). In a typical organization, this means studying physical manifestations of the culture such as dress norms, reported salaries, annual reports, and workplace furnishings and atmosphere. In addition, subjective meanings associated with these physical symbols are interpreted. In a virtual organization, these physical cultural symbols are missing so we focus on unique types of physical manifestations of the GNUe culture such as website documentation and downloadable source code. We use subjective interpretations in our analysis of website documents; IRC archives; mailing lists; kernel cousins; email interviews; and observations and personal interviews from open source conferences. We invoke an ideational approach to cultural analysis interpreting how the beliefs and values of the free software movement manifest themselves into the work practices of a virtual community whose purpose is to develop free software.

Figure 1 shows our interpretation of the relationship of the GNUe subculture to the social world (Kling and Iacono, 1984) of free software developers. As members of the FSF, free software developers share an ideology based on the belief in freedom and the belief in free software. This shared ideology has become a way of life in free software development and is best described

below as a stable community:

> "When beliefs, values, and norms develop over time into the relatively stable, unified, and coherent clusters that comprise ideologies, they provide causal models for explaining and legitimating collective and individual behaviors. Ideologies explain and justify existing social systems in ways that make them seem natural, logically compelling, and morally acceptable (Trice and Beyer, 2000, p. 34)".

Within this virtual community of free and open software developers, subcultures have formed to build free software such as GNUe and other free software projects. Within each of these subcultures, manifestations of the beliefs and values of the free software movement into work practices may be similar, while others may be unique to a particular project. For example, they may all adhere to the tenant that free software should be developed using the GPL in software development practices, but some may be stricter than others in requirements for exclusive use of free software tools to develop free software.

Our goal in this study of the GNUe virtual community is to present a picture of how free software is produced and how the social processes formed in its culture influence this production. We offer a thick description (Geetz, 1973) of their work culture defining it as a metaphor for organization with context-specific knowledge (Smircich, 1983).

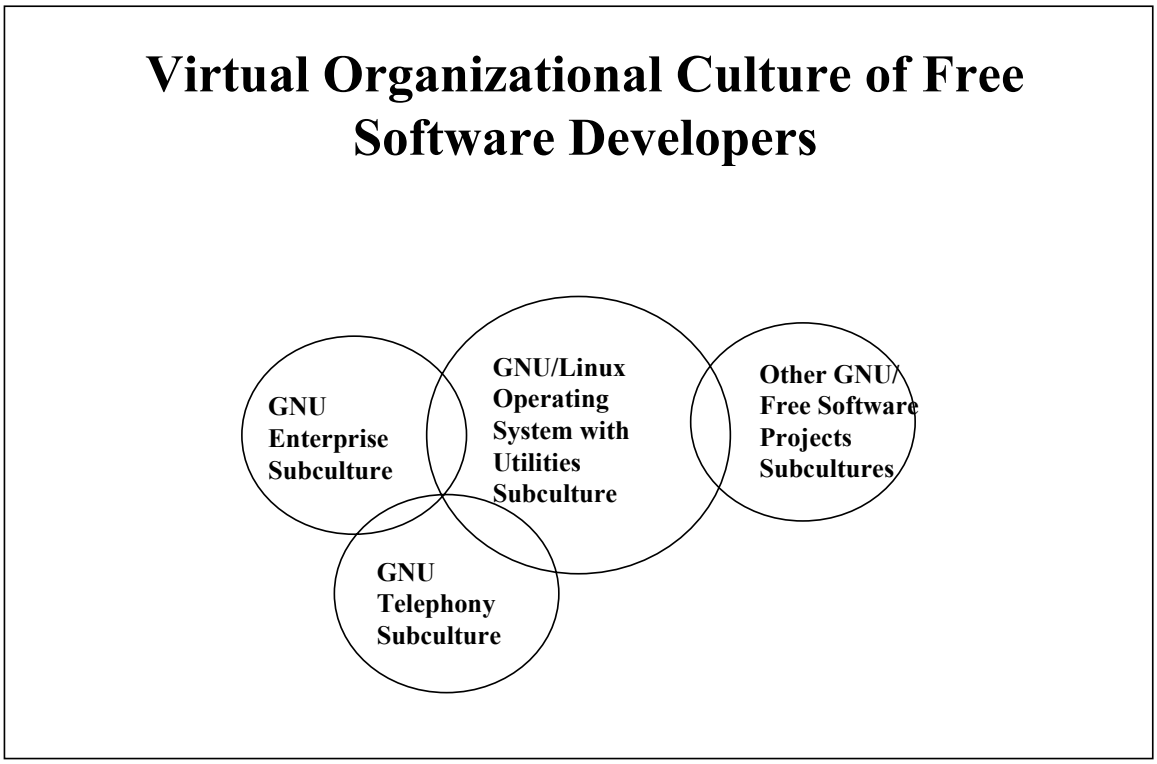

Figure 1 – Free Software Organizational Culture

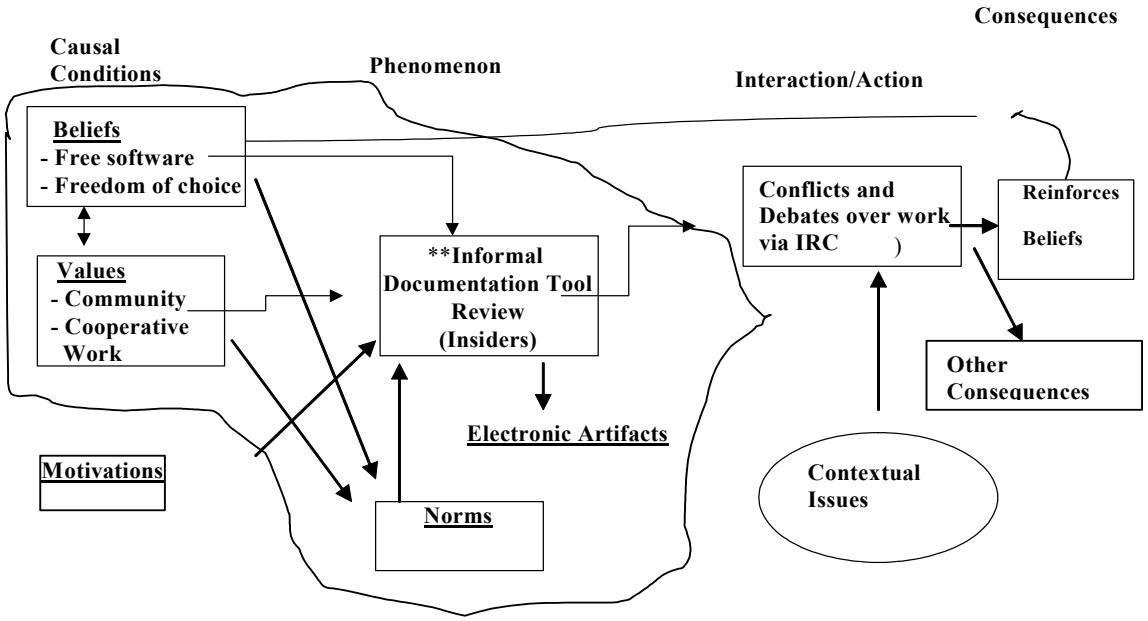

Figure 2 Summary Schematic of GNUe Research Schema

Using a single-perspective approach, we utilize the integration perspective for our cultural analysis. The integration perspective focuses on cultural manifestations that have mutually consistent interpretations. The focus is on organization-wide consensus although not necessarily unanimous beliefs throughout an organization. In Section 7, we describe the codings and variables derived in the grounded theory that form the analytical schema for our research. Then in Sections 8 and 9, two in-depth case studies are presented as exemplars of these codings. Future comparison of this culture with other open source cultures may lead to generalizations but this particular case study is that of a single culture.

GNU Enterprise - Mozilla {Build ID: 2002052306}

File Edit View Go Bookmarks Tools Window Help

http://www.gnuenterprise.org/sections.php?op=viewarticle&artid=24 Search

Home Bookmarks The Mozilla Or... Latest Builds

GNUe http://www.gnuenterprise.org

Search

Free software for your business

GNUe Project

Home

What is GNUe?

Downloads!
User Docs
Bug Tracking
Screenshots
Mailing Lists
IRC
Kernel Cousins
Sponsors
Contact Us
Contributors
GNUe History
Official FAQ

Misc
External Links
Recommend GNUe
Your Account
Submit News

Other Options
- FAQ

GNUe Tools
Forms
Designer
Reports
Application Server
Integrator

What is GNUe? An Overview.

(347 total words in this text)
(21220 reads)

GNU Enterprise (GNUe) is a suite of tools and applications for solving the needs of the enterprise. From human resources, accounting, customer relationship management and project management to supply chain or e-commerce, GNUe can handle the needs of any business, large or small. If you are looking for a full-function ERP, GNUe is the package for you.

Beyond applications, GNUe is a development framework that enables enterprise information technology professionals to customize applications for their businesses. The GNUe platform boasts an open architecture and easy maintenance. It gives users a modular system and freedom from being stuck with a single-source vendor. Plus, users get consistency and the ability to tap into a network of best practices from other enterprises, saving valuable development time.

GNUe is a Free Software project with a corps of volunteer developers around the world working on GNUe projects. This provides the added benefits of easy internationalization of applications. The project is working to provide a worldwide GNUe community, allowing everyone who is involved in the project access to other talented business information technology professionals.

GNUe supports multi-currency processing (including euro support) and multi-language interfaces.

Figure 3 Display of background information on the GNUenterprise.org and its GNUe software (Source: http://www.gnuenterprise.org/sections.php?op=viewarticle&arid=24, June 2002)

7 Observed Variables and Codings

Grounded theory is a qualitative research method based on a set of procedures to develop a grounded theory about a phenomenon (Strauss and Corbin, 1990). Using this technique, the research findings constitute a theoretical framework "grounded" in the data, with built-in testing of the theory incorporated into the results. The grounded theory techniques have been discussed in detail in the research methods literature (Glaser and Strauss, 1967; Strauss and Corbin, 1990). Here we briefly discuss them to clarify the presentation of the analytical framework. The first step in analyzing data collected is to perform open coding, the process of breaking down, examining, comparing, conceptualizing, and categorizing the data (Strauss and Corbin, 1990). The next step is axial coding where the data is reexamined looking for connections between categories utilizing the grounded theory coding paradigm consisting of the following:

- Causal Conditions – Events, incidents, or happenings that lead to the occurrence or development of a phenomenon.
- Phenomenon – Identification of the main event, idea, or happening about which a set of actions or interactions are related to, or directed at managing,
- Interaction/Action – Strategires created to manage, handle, or respond to a phenomenon under a certain set of perceived conditions.
- Consequences – Outcomes or results of action and interaction.
- Intervening Conditions – Structural conditions that influence action/interactional strategires pertaining to a phenomenon.
- Context – Specific set of properties pertaining to a phenomenon along a dimensional range.

Figure 2 shows a summary schematic of the GNUe research paradigm. The virtual organizational culture is shown as a cross-section of the entire diagram. The causal conditions consist of the beliefs (free software and freedom of choice) and the values (cooperative work and community) along with motivations for joining GNUe. The phenomenon is the free software development process – its formal and informal work practices. The interaction/action occurs on the IRC and mailing lists. It consists of the conflicts and debates over coding and documentation issues. Finally, one consequence is that the cultural beliefs are reinforced and thereby perpetuate the community. There are other consequences that are discussed in detail in the conclusion. In the next sections, beliefs, values, motivations, norms, and informal practices will be discussed.

7.1 Beliefs

Beliefs form the core of ideologies and as such, are an important component of any cultural study. As an expression of cause and effect relations, they are one of the causal conditions leading to the phenomenon of free/open source software production. They can be espoused by an organization or inferred by a researcher.

7.1.1 Belief in Free Software

The belief in free software appears to be a core motivator of free software developers. GNUe developers extoll the virtues of free software on its Web site and in daily activity on the IRC

logs. This belief is manifested in electronic artifacts such as the Web pages, source code, software design diagrams, and accompanying articles on their website and elsewhere. The data analysis of the GNUe cases showed that this belief varies from moderate to strong in strength. For example, those who have a strong belief in free software refuse to use any form of non-free software (even for development purposes such as a commercial text editor). The GNUe Web site advertises that it is "a free software project with a corps of volunteer developers around the world working on GNUe projects". Figure 3 shows a GNUe website page advertising itself as a world-wide enterprise. The Web site provides a link to an article by RMS. The GNU*e* project is advertised as "putting the free back into free enterprise" on its Web site. The GNUe software is licensed under the GNU General Public License (GPL) (http://www.gnu.org/copyleft/gpl.html). The preamble to this license states the philosophy behind the free software approach:

> "The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to *guarantee your freedom to share and change free software*--to make sure the software is free for all its users ... When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and *that you know you can do these things*." [Emphasis added] (http://www.gnu.org/copyleft/gpl.html).

The belief in free software is manifested formally, through the rights and imperatives afforded in the GPL that one realizes if employing free software, and informally in the moral imperatives (emphasized in the quote) that contextualize the software development work practices of free software contributors.

Belief in freedom is a recurring theme throughout the data. In fact, the free software movement is claimed by RMS to be inspired from the ideals of the American Revolution of 1776: freedom, community, and voluntary cooperation (Stallman, 2001a). During a lengthy heated debate regarding the use of free versus non-free software to develop documentation, one GNUe developer claimed that he is working on GNUe for the "freedom of his son". The GPL was designed by RMS for the following reason:

> "to uphold and defend the freedoms that define free software – to use the words of 1776, it establishes them as inalienable rights for programs released under the GPL. It ensures that you have the freedom to study, change, and redistribute the program, by saying that nobody is authorized to take these freedoms away from you by redistributing the program under a restrictive license (Stallman, 2001a).

The following is an excerpt from an essay titled "The Free Software Definition" found on the FSF Website explaining the concept of free software:

> "We maintain this free software definition to show clearly what must be true

about a particular software program for it to be considered free software.
"Free software" is a matter of liberty, not price. To understand the concept, you should think of "free" as in "free speech", not as in "free beer".
Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:

- The freedom to run the program, for any purpose (freedom 0).
- The freedom to study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help your neighbor (freedom 2).
- The freedom to improve the program, and release your improvements to the public, so that the whole community benefits. (freedom 3). Access to the source code is a precondition for this.

A program is free software if users have all of these freedoms. Thus, you should be free to redistribute copies, either with or without modifications, either gratis or charging a fee for distribution, to anyone anywhere. Being free to do these things means (among other things) that you do not have to ask or pay for permission. You should also have the freedom to make modifications and use them privately in your own work or play, without even mentioning that they exist. If you do publish your changes, you should not be required to notify anyone in particular, or in any particular way. (http://www.fsf.org/philosophy/free-sw.html)".

As evidence that the GNUe community believes in the essence of the GPL, throughout the GNUe digests and IRC logs, there are numerous references to the importance of adhering to the principles of free software. In the following excerpt from a GNUe IRC transcript, an infrequent contributor CyrilB identifies a diagram created with non-free software that is posted on the GNUe website to document future GNUe screen images. He questions this and calls it "quite shocking". Surprisingly, core contributors discuss it with CyrilB and eventually, a solution is found to recreate the graphic with free software:

<CyrilB> Hello
<CyrilB> Several images on the GNUe website seems to be made with non-free Adobe softwares, I hope I'm wrong: it is quite shocking. Does anybody now more on the subject ?
<CyrilB> lynx -source http://www.GNUe/modules/NS-My_eGallery/gallery/GNUe/GNUePkgArchitecture.png | strings | head
<CyrilB> We should avoid using non-free software at all cost, am I wrong ?
<CyrilB> Anyone awake in here ?

In another case, chillywilly, a frequent contributor (insider), brings up the subject of creating documentation using non-free tools. His problem is that in order to use the required free package, lyx, to read and edit the original documentation, he would also have to install some non-free graphics tools.

Action: chillywilly trout whips jamest for making lyx docs
Action: jcater troutslaps chillywilly for troutslapping jamest for making easy to do docs
<chillywilly> lyx requires non-free software
<Maniac> lyx rules
<chillywilly> should that be acceptable for a GNU project?
<Maniac> chillywilly: did he type it on a non-free computer?
<Mr_You> heh
<chillywilly> Maniac: you make no **** sense
<Maniac> :)
<jcater> chillywilly: basically, given the time frame we are in, it's either LyX documentation with this release, or no documentation for a while (until we can get some other stinking system in place)
<jcater> pick one :)
<chillywilly> use docbook then

7.1.1.1 Belief in Freedom of Choice

Open source software developers are attracted to the occupation of OSSD for its freedom of choice in assignments. Both paid and unpaid GNUe participants to some degree can select the work they prefer. This belief is manifested in the informal methods used to assign or select work in an open source project. Pavilcek (2000) describes this motivation as:

> "Another cherished priority in geek culture is the ability of the geek to pursue her passions and ideas. Their bosses assign most people working in the software industry to projects. In geek culture as well, people are often willing to take on tasks that need to be done, even if it is a task they do not relish the thought of pursuing. But geek culture recognizes that there are also tasks that need to be done not because a project requires it, but because the task is burning in the heart and mind of the geek" (Pavlicek, 2000, p. 56)".

During an interview with one of the core contributors of GNUe at a LinuxWorld conference in August 2002, we asked how assignments were made and monitored. Derek answered with:

> "The number one rule in free software is 'never do timelines or roadmaps'. This is a problem in open source projects. We could use a better roadmap - not having one hinders us. The features we add come about by need during consulting implementations. We may need some kind of roadmap in the future as we expand with more people."

The belief in freedom of choice also refers to the ability to select the tool of choice to develop free software. Some OSS developers believe that a mix of free versus non-free software tools is acceptable when developing free software. Others adhere to a strict belief in free software and believe that OSS developers should chose free tools only for software development.

In fact, in the first case, during the IRC discussion regarding the non-free tool choice, Neilt, the originator of the diagram, expressed his concern that strict belief in free software use limits the freedom of choice that should be inherent in free software projects:

<CyrilB> reinhard: neilt just said Adobe non-free software make him avoid loosing time.
<CyrilB> reinhard: and free software DO make him loose time
<neilt> CyrilB: i agree to goal GNUe, that is to use free software for stuff in cvs
Action: CyrilB is shocked
<neilt> so if there is a free software alternative, i will support that
<CyrilB> neilt: I've used xfig a couple of time, I can help you.
<neilt> i did not know xfig existed
<neilt> i am installing now
<neilt> i'll let you kow
<neilt> know how it works
<neilt> otoh i see no reason to avoid non-free software either
<neilt> if this is really a freedom thing then we should be free to use whatever we w

7.1.2 Values

Values are preferences for particular outcomes and the values in community and cooperative work are inferred from the research.

7.1.2.1 **Value in Community**

The GNUe online community exists for the purpose of developing a free ERP system. The beliefs in free software and freedom of choice foster a value in community building as part of routine work. In some cases, this value is espoused as stated below:

> "Many free software folks think IRC is a waste of time as there is 'goofing off', but honestly I can say its what builds a community. I think a community is necessary to survive. For example GNUe has been around for more than 3 years. I can not tell you how many projects have come and gone that were supposed be competition or such. I put our longevity solely to the fact that we have a community." (Derek, email interview (2002))

In other cases, it is inferred from the research and is evident in the IRC archives when newcomers join GNUe offering contributions and GNUe contributors quickly accept them as part of the community.

7.1.2.2 **Value in Cooperative Work**

The GNUe community's beliefs in free software and freedom of choice combined with the value in community fosters a value in cooperative work. As with previous researchers (Easterbrook *et al.*, 1993), our results indicate that conflict arises during the course of cooperative work. This value is evident when GNUe contributors work together to resolve technical and social issues on the IRC and mailing lists.

7.2 Norms

Norms are the socially accepted means of attaining outcomes. Here we discuss the GNUe norms of open disclosure, informal management, and immediate acceptance of outsiders.

7.2.1 Open Disclosure

Open disclosure refers to the open content of the GNUe Website including the software source code, documentation, and archived records of IRC, kernel cousins, and mailing list interchanges. The GNUe contributors join others online via IRC on a daily basis and record the conversations for future reference. All documentation and source code are easily downloaded from the GNUe website and user criticism is welcomed by frequent GNUe maintainers. In the "geek" culture, truth is a core priority in developing open source software: "It should not be too surprising, then, that one of the key values for the community is truth. In a world where people are constantly exchanging ideas, evaluating concepts, and suggesting enhancements, it is vitally important that everyone speak the truth as he sees it. If someone fails to speak the truth, the process of creating software will be greatly impaired (Pavlicek, 2000, p. 53)."

In the GNUe culture and in the open source culture in general, the importance of speaking the truth in daily work practices is a key element of their culture. In the GNUe project, truth is apparent in the norm of open disclosure of software development processes. This is accomplished by the recording and public archiving of CMC via various mediums all recorded for archival purposes: IRC logs, email discussions, and digests (i.e. kernel cousins).

Each digest summarizes IRC logs and/or email messages for a period of from one to two weeks, includes direct quotes from participants, and includes hyperlinks to the original message sources. A digest sometimes reads more like a dramatized account with editorial remarks than like a simple summary of facts. These summaries serve as a resource and organizational memory of activities within the GNUe virtual organization (Ackerman and Halverson, 2000).

7.2.2 Informal Management

The entire GNUe virtual organization is informal. There is no lead organization or prime contractor that has brought together the alliance of individuals and sponsering firms as a network virtual organization. It is more of an emergent organizational form where participants have in a sense discovered each other, and have brought together their individual competencies and contributions in a way whereby they can be integrated or made to interoperate (Crowston and Scozzi, 2002). In GNUe no company has administrative authority or resource control to determine: (a) what work will be done; (b) what the schedule will be; (c) who will be assigned to perform specified tasks; (d) whether available resources for the project are adequate, viable, or extraneous; nor (e) who will be fired or reassigned for inadequate job performance. As such, there is comparatively little administrative overhead to sustain ongoing software development and community portal support activities. Instead, there is a group of core developers, secondary contributors, and casual volunteers who review and comment on what has been done. The participants come from different small companies or act as individuals that collectively act to move the GNUe software and the GNUe community forward. Thus, the participants self-organize in a manner more like a meritocracy (Fielding, 1999).

Due to this informal management, GNUe contributors volunteer their time to the project and their work is not subject to strict timelines or due dates. Core maintainers might occasionally chide someone who is falling behind in developing an agreed-upon module, but mainly there is a flow to the work determined by participants' availability. A frequent contributor describes it as:

<reinhard> the good thing about free software projects is that there is no strict timeline :)

Another frequent contributor explains the typical method of managing the GNUe software assignments as:

> "The number one rule in free software is 'never do timelines or roadmaps'. This is a problem in open source projects. We could use a better roadmap, not having one hinders us. The features we add come about by need during consulting implementations. We may need some kind of roadmap in the future as we expand with more people."
> (Derek, face-to-face interview, August 2002)

Along with this informal management comes the easy acceptance of outsider contributions.

7.2.3 Immediate Acceptance of Outsider Reviews of Code, Documentation, and Procedures

Outsiders are welcome to join the GNUe daily IRC at any time. Many lurkers[1] remain on the IRC channel for hours at a time. If an infrequent contributor or newcomer criticizes code, documentation, or procedure, their comments are recognized and respected, rather than ignored. The first GNUe case exemplifies this immediate acceptance of a newcomer's critical review of documentation.

7.3 Motivations

Motivations for joining free/open source communities include the joy of programming, freedom to choose work, beliefs in freedom, and establishing "geek" fame on the Internet (Pavlicek, 2000). Other cited motivations involve scratching a "personal itch" with regard to software development, and wishing to be part of a team of programmers (Raymond, 2000). Open source programming has also been likened to a "gift" culture where a developer's reputation depends on free contributions (Raymond, 2001). Other explanations for motivations to join open source projects have been from an economic perspective (Hahn *et al.*, 2002). In this study, we focus on the social motivations for joining a free software project and show how these motivations are linked to the resolution of conflict in everyday work.

7.3.1 Fun at Work

There is a sense of camaraderie and bantering that takes place among regulars to the GNUe IRC. Insiders on the IRC exchange "inside" jokes and generally discuss personal issues in a fun-loving manner interspersed with serious work.

In addition to this sense of camaraderie as motivation to join a free/open source project, many people who join free/open source projects do so for the joy of programming. The first author attended a seminar at a recent open source conference where a programmer talked about how he had spent most of the night into the early morning hours gleefully programming with a group of

[1] People who observe IRC sessions and browse the website without otherwise making their presence known to others participating in the project. Their login name appears on the IRC log, even though they do not contribute text to the discussion.

open source hackers over the Internet. He indicated that it was the sheer joy of programming that motivated him. Pavlicek (2000) describes the joy of programming as a motivation of open source programmers:

> "There is great joy in the pursuit of creating something that improves the world… Many people involved in Open Source software development have been at least partially motivated by the joy of programming. For many of these people, producing excellent software is like producing excellent art. The fact that they might never see any substantial remuneration from their work is not the issue. They had something inside of them that they wanted to put out into the world. They can look at what they have done and realize that they have made the world a little better than it was, and, in some sense, just a little more beautiful (Pavlicek, 2000, p. 13)."

7.3.2 Personal Reward

Personal rewards for participating in free/open source come in the form of monetary gains for some and professional advancement for others. In the GNUe community, there are twelve businesses as of 2001-2003 that support programmers to work on the GNUe project (usually as part of other non-GNUe routine work). Other GNUe programmers act as paid consultants to companies attempting to use GNUe software. In addition, those who are unpaid contributors build a reputation via the open disclosure norm as open source programmers. This recognition can in turn lead to a more lucrative professional position (Hahn *et al.* 2002).

7.3.3 Idealism – Beliefs in Free Software and Freedom of Choice

These beliefs were described above and are listed here to emphasize the connection between the idealistic beliefs of the GNUe contributors and the motivation to perform the work. Since only a few people are being reimbursed for their work, incentive to continue the contributions comes partly from idealistic beliefs in the work.

7.4 Work Practices

Two areas of GNUe software development are analyzed in this study: 1) impromptu/informal realtime code and documentation reviews, and 2) the mitigation and resolution of conflict over the IRC. The IRC medium is used by GNUe contributors to communicate on a daily basis.

7.4.1 Impromptu Realtime Code and Documentation Review

GNUe contributors participate in informal/impromptu code and documentation reviews. They take place on the IRC basically when someone logs on to the IRC with suggestions for improvement. Code and documentation reviews happen in two ways:

1) Contributors complete source code or documentation and ask for volunteers on the IRC to test and debug their modules prior to an official release.
2) Contributors find bugs and report bug fixes or design changes concerning official releases of the code or documentation (on the IRC or mailing list).

7.4.2 Mitigation and Resolution of Conflict over IRC

Individual and group problems are mitigated and resolved over the IRC. We present two

instances of this phenomenon in our case studies.

7.5 Intervening Conditions

Two intervening conditions influence the software development practices: time zone differences and unreliable network connections.

7.5.1 Time zone differences.

GNUe contributors come from various time zones across the world from Lithuania to Pittsburgh. These time differences often result in contributors eating breakfast in one country during an IRC conversation, while in another country people are eating dinner or getting ready for sleep. These varied connection times make it difficult for a lead maintainer to plan an online meeting.

7.5.2 Unreliable Network Connections.

The network connections for the GNUe chats are slow, unstable, and often prone to shutdowns. During one IRC session used as part of our data collection, nearly all participants experienced netsplits for almost two hours. The IRC logs are used to refresh and update people to the GNUe development efforts for the day so delays in connections can impact the progress of GNUe contributors.

7.6 Electronic Artifacts

The electronic artifacts for the GNUe project include the GNUe Website, IRC archives, kernel cousins archives, mailing list archives, GNUe documentation of code and user's guides, and downloadable software for various platforms.

8 Case One: Conflict and Debate over Use of Non-Free Graphics Tool

In this section we present the first case study that reveals a trajectory of a conflict and debate over the use of a non-free tool to create a graphic on the GNUe website. This exchange takes place on November 25, 2001 on the IRC channel and ends the next morning. This example illustrates the ease with which a newcomer comes onboard and criticizes the methods used to produce a graphical representation of a screenshot on the GNUe website. CyrilB, an outsider to GNUe, finds a graphic that was created using Adobe Photoshop, a non-free graphical tool. He begins the interchange with a challenge to anyone onboard stating that "it is quite shocking" to see the use of non-free software on a free software project. He exhibits a **strong belief in free software,** which causes a debate lasting a couple of days. Table 4 displays the total number of contributors and the number of days of the conflict. Eight of the nine regular GNUe contributors were software developers and one was working on documentation. The infrequent contributors drifted on and off throughout the day – sometimes lurking and other times involved in the discussion.

Table 4 – Contributors and Duration of Conflict in Case One

Total Contributors	Regular Contributors	Infrequent Contributors	Number of Days
17	9	8	1

The **strong belief in free software** of the outsider leads to conflict among those insiders who have a moderate view of the use of free software for GNUe software development. A daylong debate ensues among the Neilt, creator of the graphic, CyrilB, and other GNUe contributors regarding the use of a non-free software tool to create a graphic for a GNUe screenshot for Website documentation. This first excerpt shows how CyrilB gets on the IRC and expresses his concern for the "shocking" use of a non-free tool on a free software project[2]:

<CyrilB> Hello **(Outsider Critique-1**
<CyrilB> Several images on the GNUe website seems to be made with non-free Adobe softwares, I hope I'm wrong: it is quite shocking. Does anybody now more on the subject ?
<CyrilB> lynx -source http://www.GNUe/modules/NS-My_eGallery/gallery/GNUe/GNUePkgArchitecture.png | strings | head
<CyrilB> We should avoid using non-free software at all cost, am I wrong ? **(Strong belief in free software (BIFS)-1)**
<CyrilB> Anyone awake in here ? **Outsider Critique-1)**
…

Reinhard, a core maintainer, arrives and points out to CyrilB that the main goal of the project is to produce good free software and *how* it is produced is not a main concern. In this next passage, Reinhard explains his moderate view of the belief in free software and surprisingly, he accepts the criticism of CyrilB (note the **Immediate acceptance of outsider-1 passage)** engaging him in conversation to explain the reason for allowing such work on a GNU free software project:

<reinhard> CyrilB: our main goal is to produce good free software **(Moderate BIFS-1**
<reinhard> we accept contributions without regarding what tools were used to do the work
<reinhard> especially we accept documentation in nearly any form we can get because we are desperate for documentation
<reinhard> just like any other gnu project
<reinhard> just as long as the format itself isn't proprietary, and it can be viewed without proprietary programs
<reinhard> anything is ok for us
<reinhard> at least that is my understanding **Moderate BIFS-1)**
<Maniac> reinhard: good point of view **(Community support-1)**
<reinhard> but if you want to redo those pictures in dia (or whatever) we will _gladly_ take it
<reinhard> the contributor was not familiar with dia at all and felt that he would be more productive when he used his adobe
<reinhard> which he is used to because he used it before
<reinhard> and we were ok with that
<reinhard> well i think "contributor" is a bit of an understatement for neilt
<reinhard> please s/contributor/core team member/ :)
<CyrilB>I understand your point of view but if you accept contributions that can be viewed with free

[2] The IRC excerpts are presented verbatim with extraneous text eliminated for clarity. They are in a different font than regular text. The codes are shown in parentheses in bold type.

softwares, you also have to be able to modify the contributions.
<CyrilB> What if we need to add a component to this graphic ?
<CyrilB> Even ASCII art graphic would be better.
<reinhard> CyrilB: isn't png viewable with free software?
<CyrilB> reinhard: png is viewable with free software, you are right.
<CyrilB> You can consider this PNG as a binary distrubution of the contrib, not the source code.
<CyrilB> We NEED to be able to modify the code.
<reinhard> so we need someone that can do the graphic in dia?
<CyrilB> And we can't modify adobe files with free softwares.
<reinhard> so we need someone that can do the graphic in dia?
<CyrilB> We need people do be able to use free softwares. **(Strong BIFS-2**
<CyrilB> And produce free documents.
<reinhard> ok **(Immediate acceptance of outsider-1**
<reinhard> you know someone who would want to do this graphic
<reinhard> as well as maintain it for adding new modules etc over the next few years?
<reinhard> i think we have to seek such a person
<reinhard> till we found it i think we can live with what we have now as an intermediate solution
<CyrilB> If this solution is considered as an intermediate solution, it is ok for me. **Strong BIFS-2)**

After several interchanges with CyrilB, Reinhard recalls from CyrilB's name that he is a regular particpant in the fsfeurope (Free Software Foundation in Europe). As such, he asks him for any other comments CyrilB may have on GNUe.

<reinhard> any other comments on gnue?
<reinhard> iirc i saw you a few times in #fsfeurope
<reinhard> but never here on #gnuenterprise?
<CyrilB> This is the first time for me in #gnuenterprise. **Immediate acceptance of outsider-1)**
<CyrilB> Did the author of this graphic understood that this file has to be freed ?
<CyrilB> If think that if he is able to produce this kind of graphics with non-free softwares, he can easily do the same with free softwares.
<reinhard> from what i know about the author i think he is aware of the issue
<reinhard> but he works on mac mostly and afaik not so much free software is available for mac
<CyrilB> This discussion is interesting and I have to talk much with you later but I have to go outside now.
<CyrilB> See you later.
Action:[3] CyrilB is away:

Once CyrilB has pointed out the use of the non-free graphic, Neilt, the original creator of the GNUe diagram (using Adobe Photoshop), joins the IRC, reviews the previous discussion on the archived IRC, and returns to discuss the issue with Reinhard and CyrilB. A lively argument ensues between Neilt and others with onlookers contributing suggestions for the use of free tools to develop the Adobe graphic. This excerpt shows how the GNUe contributors work together as

[3] The term Action: is used occasionally by contributors to signify a physical movement of some sort often in humorous terms, or to bring attention to a statement.

a community to resolve the issue in a cooperative manner:

<reinhard> hello neilt
<reinhard> i just had a discussion about the graphics you did for the homepage **(Value in Community and cooperative work-1**
<reinhard> i hope what i said is ok for you
<neilt> hello
Action: neilt goes to look at log
<neilt> reinhard: it would be nice of we had free software that would do nice diagrams
<neilt> reinhard: it does not exist
<reinhard> IIRC i have never made such a diagram in my entire life
<reinhard> neither with proprietary nor with free software :)
<reinhard> so i can't tell but i can imagine very well you're right :)
<neilt> you do know that my graphics are the most viewed screenshots on the web site :)
<neilt> and it was not me sitting here hitting the reload button
<neilt> :)
<reinhard> lol
<neilt> i think i missed the point
<neilt> what is the problem with the graphics
<reinhard> his point was that only you have the "source" and only you can change the graphic
<reinhard> if i understand him correctly
<reinhard> as you are probably the only one among us that has this adobe software
<neilt> any program that reads png which is a standard format can edit the graphic
<neilt> if you can suggest another format, derek said this was the preferred for graphics, then I will convert to a better format
<reinhard> i think his point is that usually you create such a graphic with a vector drawing too
<reinhard> tool
<reinhard> however this is somewhat "silent post"
<neilt> png is a vector format graphic, so all vector information is in the version on the web
<reinhard> i thought png is a pixle graphic format
<reinhard> but i can be wrong i'm not sure at all
<neilt> portable network graphic
<neilt> http://www.libpng.org/pub/png/

Meanwhile Maniac, who has been “listening” to this debate, jumps in and gives technical details about a PNG image. Then Reinhard and Neilt agree that CyrilB had a valid point since a PNG has no vector information stored. These exchanges illustrate how the IRC medium enables the cooperative work needed to resolve this issue. It also conveys the community spirit and cooperative work ethic that is a value in the GNUe work culture. They both agree to wait until CyrilB comes back to give more suggestions for an alternative.

<Maniac> a PNG image saved in one app is readable in any other PNG-supporting application
<reinhard> from what i can read on that page png has no vector information stored
<reinhard> but is rather comparable to gif, jpeg etc.
<neilt> ok, you might be right
<neilt> i thought it was a vector format

<reinhard> nevermind
<reinhard> so actually a good thing that CyrilB brought that up
<neilt> but gimp and gnome both edit png files
<reinhard> sure gimp is a pixle drawing tool
<reinhard> not a vector drawing tool
<reinhard> gimp is like photoshop or so
<reinhard> and i'm sure you wouldn't want to edit this graphic in photoshop ;)
<neilt> http://www.libpng.org/pub/png/pngaped.html is a list of applications that can edit png files
<neilt> i hope not
<neilt> what would be a vector format we should use?
<reinhard> i can't tell
<reinhard> i know zero about graphics at all
<reinhard> neither proprietary nor free
<reinhard> you know i don't deal with things i can't view in vi ;)
<reinhard> CyrilB should be back in some time, maybe he can make a proposal
<reinhard> off for a bit **Value in community and cooperative work-1)**

Neilt has an idea to switch the graphics to the svg format but later changes his mind when CyrilB returns and suggests the use of the xfig tool. CyrilB and Neilt have an exchange about beliefs regarding free versus non-free software with Neilt finally agreeing to a change in the graphic to "free" software if there is an alternative. However, he also points out that his freedom of choice is being rescinded by this strict adherence to the use of free software to develop GNUe.

<neilt> i'll probably switch the graphics to svg **(Change to documentation-1)**
<neilt> http://www.w3.org/Graphics/SVG/SVG-Implementations#svgedit
<neilt> http://sodipodi.sourceforge.net/ is the editor from gnome
<neilt> i just need someone to let me know if http://sodipodi.sourceforge.net/ actually works?
<neilt> also http://www.levien.com/svg/ will edit svg
<neilt> interesting article is at http://xml.com/pub/a/2001/11/21/svgtools.html
…
<chillywilly> sodipodi works
Action: chillywilly is away: I'm busy
<nickr> sodipodi works but it needs lots of work, its not all that pleasent to use

Action: CyrilB is back (gone 00:00:01)
<CyrilB> I'm back.
<CyrilB> <-- reinhard
<reinhard> wb
<reinhard> please read the logs
<reinhard> the author of the drawings was here meanwhile
<CyrilB> I've read the logs.
<CyrilB> Did the author left ?
<CyrilB> (yes)
<CyrilB> 2 hours ago
<reinhard> yes he left

<reinhard> but i think he got your point
<CyrilB> The point is not vector drawing or pixel drawing, as you say I don't deal with non ascii files. **(Strong BIFS-3**
<CyrilB> The point is free software VS non-free software.
neilt (~neilt@dhcp64-134-54-175.chan.dca.wayport.net) joined #gnuenterprise.
<reinhard> oh
<CyrilB> neilt: welcome back.
<reinhard> neilt: you have a watchdog or something like that?
<neilt> yep
<neilt> :)
<CyrilB> <neilt> reinhard: it would be nice of we had free software that would do nice diagrams (...) it does not exist
<neilt> CyrilB: what do you suggest
<CyrilB> neilt: friends of mine are using dia and xfig for this kind of graphics.
<neilt> dia ****
<reinhard> i can second that
<neilt> what is xfig
<reinhard> dia can't even display the same font on screen as it does when printing out
<CyrilB> neilt: dia **** less than adobe non-free softwares
<CyrilB> neilt: but dia sucks, you are right
<neilt> CyrilB: i am sorry, but i am not a biggot about software
<CyrilB> http://freshmeat.net/projects/xfig/
<neilt> my time is valuable
<neilt> anything that wasts my time is not good, free or not
<CyrilB> What is 'biggot' ?
<neilt> biased
<CyrilB> neilt: I don't agree at all, we should use free software at all costs. **Strong BIFS-3)**
<reinhard> CyrilB: no i don't agree here
<reinhard> we should develop good free software at all costs **(Moderate BIFS-2**
<CyrilB> reinhard: using non-free software _do not_ 'develop good free software'.
<reinhard> it cn
<reinhard> can
<CyrilB> and promoting them is really a shame.
<CyrilB> reinhard: not in this case. **Moderate BIFS-2)**

Neilt installs xfig in realtime:

Action: neilt does install xfig
<neilt> installing now
<reinhard> i don't see that this is promoting
<reinhard> a "normal user" doesn't see where this png comes from
<reinhard> afaik
<CyrilB> reinhard: neilt just said Adobe non-free software make him avoid loosing time.
<CyrilB> reinhard: and free software DO make him loose time
<neilt> CyrilB: i agree to goal GNUe, that is to use free software for stuff in cvs
Action: CyrilB is shocked

<neilt> so if there is a free software alternative, i will support that **(Debate reinforces beliefs-1)**

During the debate, the beliefs in free software and freedom of choice are reinforced by the persistent recordation of the arguments. Through real-time testing and team discussion of free tool alternatives, the issue is resolved.

<CyrilB> neilt: I've used xfig a couple of time, I can help you.
<neilt> i did not know xfig existed
<neilt> i am installing now
<neilt> i'll let you kow
<neilt> know how it works
<neilt> otoh i see no reason to avoid non-free software either **(Belief in Freedom of Choice (BIFC)-1**
<neilt> if this is really a freedom thing then we should be free to use whatever we want
psu (psu@manorcon.demon.co.uk) joined #gnuenterprise.
<ajmitch> hi psu
<reinhard> neilt: i agree
<reinhard> neilt: as long as we don't take away freedom from others
<reinhard> for example
<neilt> in you case you are saying its not about freedom i guess, its about using what the free software movement tells you to use
<neilt> that is just another form of bias
<psu> hi aj
<reinhard> i think it's ok if i use vi, emacs or even windows notepad to write my source code
<ajmitch> psu: seems you've walked into a free vs non-free debate again :)
<reinhard> but if i used winword and stored as .doc i would take away freedom from those wanting to read the text
<neilt> again
<neilt> only because i am being told that using non-free software is bad :(
<neilt> i wish we could just leave the opinions about free vs non-free out **BIFC-1)**
<reinhard> the main point is that we want to achieve something
<reinhard> a very ambitious goal actually in this project
<reinhard> and everyone tries is best to reach that goal
<reinhard> his best
<reinhard> and we make compromises if it helps the overall goal
<reinhard> anyway
<reinhard> gotta get me some sleep
<reinhard> night all

<neilt> reinhard: later
<neilt> CyrilB: ok i installed xfig
<CyrilB> neilt: the interface is ugly but useful.
<neilt> CyrilB: sorry but i have to suffer through this, because you think its better why?
<neilt> CyrilB: what do you do professionally?

While Neilt questions the credentials of CyrilB who is causing him to redo a trivial diagram,

mdean makes a suggestion to use dia. Mdean has already tested dia on Neilt's diagram. Here is an example of the community spirit that ties the GNUe developers together. In order to preserve the "free" nature of GNUe, all work together to find a solution.

<mdean> dia is much more useful than xfig for these types of diagrams
<neilt> mdean: does dia do color
<neilt> ?
<mdean> yup
<nickr> definitely
<mdean> I was just messing around with it - converting your diagram
<mdean> it can definitely do it
<CyrilB> neilt: you are compromising our freedom by using non-free software: we can't modify and/or redistribute the source vector file. **(Strong View BIFS-4)**
<neilt> so we should change to *.xfig because its such a popular format?
<CyrilB> neilt: you don't need colors
<neilt> CyrilB: thats pure crap
<CyrilB> neilt: not popular but FREE
<neilt> CyrilB: what do you do professionally?
<nickr> dia outputs a free format, which is XML
<CyrilB> neilt: sysadmin in a european isp
<mdean> dia exports png, eps and other popular formats too ;-)
<nickr> which makxes it exceptionally useful for converting dia diagrams into things like SQL
<neilt> CyrilB: cool, thanks

More people get involved now and discuss the technical details of which format and tool to use. Finally, CyrilB exits and apologizes for any offensive behavior:

<CyrilB> neilt: excuse me if I said stupid things.
<neilt> CyrilB: no problem, i dont takes things badly
<CyrilB> neilt: do not hesitate to contact me if you have any questions with xfig: cyb@gnu.org
<neilt> i just feel free to express myself
<neilt> as everyone else should
<neilt> CyrilB: thank you

The next day, November 26, 2001, CyrilB is on the IRC again and derek discusses the solution:

<derek> CyrilB: re your earlier complaint about images being made in prop tools **(Moderate BIFS-3**
<derek> a. we use a modified phpNuke (gpl) so the images came from that, i.e. i would beat them up about it :)
<derek> b. i dont care what tools another developer uses as long as with no fuss i can use free tools
<derek> for example if i write something in C++ it doesnt matter to me if someone else wants to use Borland CBuilder
<derek> as long as prop extensions and files are not used to prevent me from using gcc
<derek> et

<derek> c
<derek> same goes for images, if someone wishes to use illustrator that is fine by me
<derek> as long as i can manipulate the file in gimp
<derek> (i.e. a large portion of gnue development has taken place on windows, we are not about to throw out that contribution because the developers of those pieces choose something that is not free
ToyMan (stuq@c5300-3-ip210.albany.thebiz.net) left irc: Ping timeout: 181 seconds
<CyrilB> derek: I understand what you mean.
<CyrilB> derek: but the fact is (as I've already written here) that the PNG are not the sources.
<derek> yes, agree, but i can edit the png if need be
<CyrilB> derek: the author used a non-free vector drawing tool that produce bitmap images.
<derek> IF it were gnue i woudl have a problem
<derek> BUT
<CyrilB> You can rename or move the nodes.
<derek> its phpNuke and frankly i dont have the energy to bug them about it :) **Moderate BIFS-3)**
<CyrilB> (easily)
<reinhard> derek: CyrilB is talking about the graphics that neilt posted on the screenshots site
<reinhard> just fyi
<CyrilB> As if you said you could change the code of a binary executable: yes you can but it's not easy.
<derek> ah
<derek> sorry
<derek> slap neilt then :)
<reinhard> but we resolved anyway

The issue ends with a clarification by derek (a core maintainer) that contributors can use any tools necessary to create free software even if some are non-free tools. In addition, he indicates to CyrilB that if the graphical object were part of the free software product, GNUe (not for a graphical diagram intended for documentation), then he would be more concerned. The discussion ends with reinhard reminding everyone that the issue is resolved (with neilt agreeing to redo it at a future date with free software). We now turn to a discussion of the data presented as case study one.

8.1 Discussion of Case One

In case one, we have demonstrated how the beliefs in free software and freedom of choice, and the values in community and cooperative work are manifested into GNUe software development processes. Figure 4 shows a summary schematic of case one that captures the sequence and configuration of relations among the observed variables identified in Section 7. All members believe in the value of free software but these beliefs vary in intensity from moderate to strong. These beliefs motivate people to contribute to GNUe and influence their activity as contributors. For moderate views, GNUe contributors believe in the ideal of using free software but temper this view with a practical attitude towards using whatever tools are appropriate to produce a free ERP system. A conflict arises when the outsider (CyrilB) with the strong belief in the exclusive use of free software criticizes the use of Adobe Photoshop to create a Website graphic.

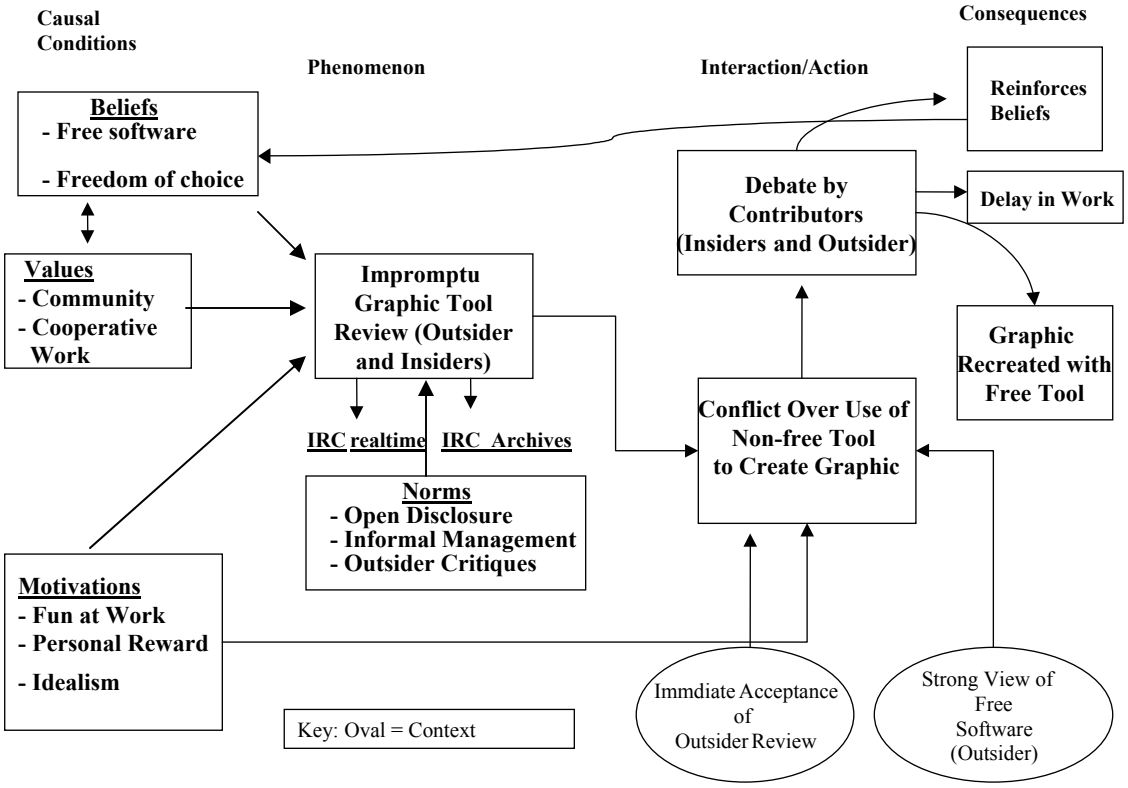

Figure 4 – A Schematic Summary of Relations Among Observed Variables in Case One

CyrilB with his **strong view of the use of free software** promotes the spirit of the free software movement by exclaiming that images on the gnuenterpise.org website seem to be made with non-free Adobe software. His reaction provokes strong reactions from GNUe contributors:

> "I hope I'm wrong: it is quite shocking…We should avoid using non-free software at all cost, am I wrong? **(Strong BIFS-1)**"

Reinhard responds with a **moderate view of free software:**

> "Our main goal is to produce good free software. We accept contributions without regarding what tools were used to do the work especially we accept documentation in nearly any form we can get because we are desparate for documentation." **(Moderate View BIFS-1).**

Outside critiques of software and procedures used during development are common to the GNUe project. One of the norms of the work culture is **immediate acceptance of outsider contributions.** Eventually, the creator of the non-free graphic questioned CyrilB's qualifications and was satisfied when he learned that CyrilB was a member of the European Free

Software Foundation.

The work practices of GNUe are strongly influenced by the beliefs, values, and norms of their organizational culture. Consequences of the debate are a reinforcement of the **belief in free software, value in community,** and **value in cooperative work;** and a recreation of a Website graphic with free software to replace the original created with Adobe Photoshop, a non-free software tool. While the discussion reinforces the beliefs and values of the work culture by persistent recordation on the IRC archives, concomitantly, it delays the work for those involved in the discussion. As part of daily communication on the IRC, GNUe contributors sometimes contemplate provocative statements regarding the use of non-free software and engage in detailed philosophical discussions on the merits of using free software. Although this may interrupt work, the reinforcement of the beliefs and values via the debates is also important toward maintaining the GNUe community. It takes one day to resolve the conflict resulting in an agreement for Neilt to change the graphic at a later date. Next we describe case two where an insider review of the use of non-free tools for documentation sparks a three day debate.

9 Case Two - GNUe Informal Documentation Tool Review

The second case study explores project insider review of the procedures and practices for developing GNUe documentation. Once again the debate revolves around polarized views of the use of non-free tools to develop GNUe documentation. In this case, Chillywilly, a frequent contributor, balks at the need to implement a non-free tool on his computer in order to edit the documentation associated with a current release. Even though his colleagues attempt to dissuade him from his concerns by suggesting that he can use any editor – free or non-free- to read the documentation in HTML or other formats, Chillywilly refuses to back down from his stance based on a **strong belief in free software**. This debate lasts three days. Table 5 displays the demographics of the IRC channel during the two day debate involved in this case study. This case exemplifies the fierce adherence to the belief in free software held by some purists in the free software movement and how it directs the work of the day. While the three day debate reinforces beliefs and values of the culture, at the same time, it ties up valuable time which could have been spent writing code or documentation.

Table 5 – Contributors and Duration of Conflict over Documentation

Total Contributors	Regular Contributors	Infrequent Contributors	Number of Days
24	9	15	3

In order to understand the following IRC and mailing list segments, some background information is needed. The GNUe core maintainers selected a free tool to use for all documentation called *docbook* (http://www.docbook.org). DocBook is based on an SGML document type definition which provides a system for writing structured documents using SGML or XML. It is becoming an increasingly popular tool for free software development environments and is used worldwide. It is particularly well-suited to books and papers about computer systems. However, several GNUe developers as of November 15, 2001 were having trouble with its installation. Consequently, they resorted to using lyx tool to create documentation. LyX

is advertised as the first WYSIWYM (What you see is what you make) document processor. "LyX is an advanced open source document processor that encourages an approach to writing based on the structure of your documents, not their appearance. LyX lets you concentrate on writing, leaving details of visual layout to the software." (http://www.lyx.org).

The problem with lyx is that even though it was developed as a free software tool, its graphical user interface (GUI) requires the installation of a non-free graphics package (*libxforms*). Chillywilly gets upset with the fact that he has to install non-free software in order to read and edit GNUe documentation. A lengthy discussion ensues with debates over which tool to use for GNUe documentation. This debate lasts for three days taking up much of the IRC time until Chillywilly finally gives up the argument. The strength in the **belief in free software** drives this discussion. The debate and its resolution also illustrate the tremendous effort by developers to collaborate and work cooperatively through the use of the IRC channel. Although the discussion is heated at moments, a sense of fun also pervades.

Chillywilly begins on the November 14, 2001 IRC with an observation that a fellow collaborator, jamest, has made documents with lyx and questions the appropriateness of using lyx which requires the installation of non-free software. Another developer, Maniac, adds humor to the problem by wondering if the creator of documentation also used a non-free computer.

Action: chillywilly trout whips jamest for making lyx docs **(Strong BIFS-5**
Action: jcater troutslaps chillywilly for troutslapping jamest for making easy to do docs
<chillywilly> lyx requires non-free software
<Maniac> lyx rules
<chillywilly> should that be acceptable for a GNU project? **Strong BIFS-5)**
<Maniac> chillywilly: did he type it on a non-free computer?
<Mr_You> heh
<chillywilly> Maniac: you make no *** sense
<Maniac> :)
<jcater> chillywilly: basically, given the time frame we are in, it's either LyX documentation with this release, or no documentation for a while (until we can get some other stinking system in place)
<jcater> pick one :)
<chillywilly> use docbook then

This discussion continues into November 15, 2001 and evolves into a discussion of the problems with docbook as well:

<chillywilly> sheesh
--- Thu Nov 15 2001
<jcater> sheesh
<jcater> I've only wasted the last few days trying to get the **** jade/openjade/docbook crap to install
<chillywilly> dude, I seriously get *** at hacing to install non-free software for a GNU projects docs

<jcater> days that could have been spent WRITING docs
<chillywilly> apt-get install blah onm ash

<Maniac> jcater: i've had that same problem which is why i resorted to lyx months ago
<chillywilly> s/onm/on
<jcater> chillywilly: I apt-got it and apt-got couldn't do it right EITHER!
<chillywilly> how can I build the docbook docs then?
<chillywilly> magic?
<jcater> I honestly don't know
<jcater> but apt-get docbook crap was broken as of last night

Maniac also jokingly asks everyone if they are NOT using windows for development.

<Maniac> mmm i hope you all are developing on GNU/Linux and not windows.....
Action: jcater whistles and ignores Maniac :) **(Moderate BIFS-4**
<chillywilly> jcater: what are you using?
<jcater> at the moment slackware and debian
<jcater> but I will be rebooting slack into Win98 shortly
<jcater> and I use Win98 + RedHat at work for development **Moderate BIFS-4)**
<chillywilly> cause you ****
<jcater> but that's out of necessity
<chillywilly> dude
<chillywilly> if I can build this docbook stuff I can then **** slap you all?
<jcater> um
<jcater> no
<jcater> btw, if LyX is such a big deal, we'll HAPPILY delegate this documentation stuff to someone else <hint, hint> as we'd rather be coding anyway :)

Maniac describes the main problem and chillywilly causes a stir by calling the developers KDE nazis. He is referring to what to him is the traitorous use of non-free software by KDE developers.

<Maniac> lyx's graphics library is non-gpl **(i.e. non-free software)**
<chillywilly> I'm not writing your docs for you
<Maniac> this is an issue the developers are aware of but do not, at this time, have the time to rectify

<chillywilly> Maniac: because they are **** KDE nazis
<chillywilly> that's who the original lyz authors are mathhias, et. al.
<chillywilly> lyx
<chillywilly> matthias
<Maniac> well, my understanding is, they are working toward UI independance, to make it able to use differnt toolkits ie. kde, gnome, xyz as time/coding permit
<chillywilly> jcater: I am supposed to document code I haven't hardly even looked at?
<jcater> chillywilly: we aren't documenting code
<chillywilly> Maniac: yes they are but until then it is not even functional without libxforms
<jcater> the code is already documented
<Maniac> chillywilly: such is the cruel reality of life
<chillywilly> jcater: I got afew warning but html docs built

<chillywilly> for the module guide
<chillywilly> Maniac: shouldn't be the reality of GNU....use **** texinfo then :P

Maniac questions chillywilly's incessant reminders about using non-free software as though this myopic view of free software development is unnecessary. Chillywilly continues his debate showing his **strong view of free software**.

<Maniac> chillywilly: so GNU projects cannot use non-GNU software in any portion of their project?

<chillywilly> no, they shouldn't use non-free software **(Strong BIFS-6**
<chillywilly> libxforms would reqwuire me to add non-free section to sources.list
<chillywilly> thus I will not do it and cannot read the **** docs
<ajmitch> can use non-GNU tho
<chillywilly> sure
<chillywilly> that's not my point though
<ajmitch> chillywilly: your KDE hatred is showing, btw
<chillywilly> I know
<chillywilly> KDE is fine actually
<ajmitch> wow
<chillywilly> but then again they did use QT before it was actually free and then the original foundere goees and writes lyx with aq non-free GUI toolkit....so I have not much love for him **Strong BIFS-6)**
<ajmitch> shoudl get psu to put that in the KC, chillywilly admitting KDE is fine ;)
<jcater> cool
…

<chillywilly> death to LyX
<chillywilly> grrrrrr
<ajmitch> lyx rocks ;)
<chillywilly> dude
<Maniac> as soon as some GNUtastic individual comes up with a GNUified Lyx i will stop using it
….
Reinhard comes onboard and chillywilly tries to enlist his support. Reinhard and Ajmitch agree with chillywilly that using non-free software is wrong for a GNU project:

<reinhard>morning **(Strong BIFS-7**
<chillywilly> reinhard: dude, should somone using GNU project software be required to install non-free software in order to read and edit the documentation?
<chillywilly> i.e., install LyX which require non-free xforms in order to read the and edit the forms documentation
<chillywilly> am I craxy for thinking this is **&^?
<ajmitch> nope
<ajmitch> i think having to use lyx is wrong for a GNU project **Strong BIFS-7)**

Reinhard agrees with chillywilly as do others, but in order to complete the documentation, they

agree to use an interim solution. Chillywilly is so adamantly opposed to the use of non-free software that he references Richard Stallman as part of his reasoning – "I will NOT install lyx and make vrms unhappy". This passage shows how RMS is considered the "guru" of the free software movement.

Action: reinhard thinks that the only valid documentation format for a gnu project is texinfo
<reinhard> but i
<reinhard> 've had this discussion before
<reinhard> and i lost
<chillywilly> well I agree with you
<chillywilly> I will NOT install lyx and make vrms unhappy **(Support for FSF leader - RMS-1)**
<Mr_You> cool.. this savannah software is a good thing
<reinhard> oh sorry
<reinhard> there's another valid documentation format i forgot
<reinhard> text/plain :)
<Mr_You> "We've finished our beer, it's time to win our freedom. **(Belief in Freedom-1)**
<chillywilly> :)

At this point, there are a lot of net-splits with people going on and off. When they convene, chillywilly has sent an email to the mailing list condemning the use of lyx by Jcater. GNUe developers communicate using IRC for instant meand using the mailing list for more serious problems but chillywilly feels so strongly about this issue and post to the blasts Jcater in email:

From Chillywilly Nov 15 IRC log

(Strong BIFS-8 OK, I saw on the commit list that you guys made some LyX documents. I think it is extremely **** that a GNU project would require me to install non-free software in order to read and modify the documentation. I mean if I cannot make vrms happy on my debian system them what good am I as a Free Software developer? **(Support for FSF leader – RMS-2)**
Is docbook really this much of a pain?
I can build html versions of stuff on my box if this is what we have to do. This just irks me beyond anything. I really shouldn't have to be harping on this issue for a GNU project, but some ppl like to take convience over freedom and this should not be tolerated. I mean I would love to read the forms technical reference, but there's no way in hell I am going to install lyX unless I can have a fully free version with the toolkit of my choice (which is supposed to happen eventually). I mean the official GNU format is texinfo anyway, at least docbook can be exported to this format, but LyX docbook won't even friggin parse without doing black magic or some shit. Is it really that unreasonable to request that we not use something that requires ppl to install non-free software? Please let me know. **Strong BIFS-8)**

Here is Jcater's response on the IRC. Notice that the document was also available in html and

text format so chillywilly could easily have read the documentation. Other developers with a more moderate view of the sole use of free software criticize this argument regarding lyx. Even though they agree that chillywilly is being unreasonable, several reflect on how they agree with him philosophically:

(Debate over choice of non-free tool
<jcater> chillywilly: what the **** were you thinking sending an email like that to our public list???
<chillywilly> what the **** were you thinkin making me have to install non-free software?
(Strong BIFS-9
<jcater> no on e made you install crap
<chillywilly> I still cannot read the forms tech reference
<jcater> sigh!
<chillywilly> well that's what I would have to do in order to read it
<jamest> it was in the last tarball release in text format!
<jamest> and IIRC html!
<chillywilly> install libxform0.88
<jcater> and on the web in HTML!
<chillywilly> and if I wanna change something? I mean are all GNUe sources gonna be in **** lyx?
<jamest> vi works real good on .lyx files **(Moderate BIFS-5)**
<chillywilly> oh that is crap
<jcater> um
<jamest> and docbook has a better editor!
<jcater> no more so than docbook
<chillywilly> lyx uses it's own little commands
<chillywilly> not like i can just look the **** up
<chillywilly> we shouls use texinfo then if docbook is such a hassle **Strong BIFS-9)**
<jamest> i wrote all the ****docs in texinfo and that wasn't good enough
<jamest> so it was docbook
<chillywilly> why would it not be good enough?
<jamest> which is great if you can build the tools
<jamest> however i can't
<jamest> and it's not from lack of trying
<jcater> nor can I
<chillywilly> what was wrong with texinfo?
<jamest> don't ask me as I don't know
<jamest> people wanted docbook
<jamest> we did docbook
<jamest> and the docs went from being a few large files
<chillywilly> you can go texinfo->docbook
<jamest> to lots and lots of files
<jamest> and a **** of utilities that you can build on any system as long as it's some sub-derivative of GNU/Linux

Chillywilly goes back to arguing that the installation of lyx does not match his philosophical orientation toward free software development. The discussion starts to deteriorate with jcater

suggesting that the argument not show up on the kc. Chillywilly ends this "meeting" with an exclamation that the lyx is evil software.

> <chillywilly> look I am not tryint to be a jerk I just cannot stand the fact of how contradictory it is to have to use lyx when I cannot get a free gui for it yet...it **** **(Strong BIFS-10**
> <jamest> it really comes down to this
> <jamest> if people want docbook that is fine with me
> <chillywilly> and what if you wanna hack docs on the serve?
> <chillywilly> **** docbook I want texinfo :P
> <jamest> however the people that are willing to put the effort into the user_guide and tech_ref (jcater and myself)
> <jcater> **** it all... read the source code!
> <jamest> are sick of fighting docbook
> <jamest> so it's either docbook and no docs from us
> <chillywilly> yes, so dump it then it is not worth it...but lyx has issues for others too
> <jamest> yes it does
> <jamest> we don't have a good solution
> <jamest> but we're trying to get docs made
> <jcater> psu: I don't think this discussion should be in the KC
> ToyMan (~stuq@c5300-2-ip42.albany.thebiz.net) left #gnuenterprise ("Client Exiting").
> <chillywilly> how about plain ol' text then for now until you decide to format it
> <chillywilly> :P
> <chillywilly> I casn read that wihout being installing evil software **Strong BIFS-10)**
> <chillywilly> s/being//
> Nick change: reinhard -> rm-away **Debate over non-free software-1)**

A lengthy discussion of technical issues unrelated to the documentation problem ensues. Meanwhile Jcater has sent a reply to Chillywilly's message to the mailing list. Jcater comes on the IRC the next day worrying about Chillywilly's response to the message. Here is Jcater's email message on the mailing list:

> I would like to personally apologize to the discussion list for the childish email you recently received. It stemmed from a conversation in IRC that quickly got out of hand. **(IRC facilatates debate-1)** It was never our intention to alienate users by using a non-standard documentation format such as LyX.
>
> Writing documentation is a tedious chore few programmers enjoy. The developers of the GNUe client tools are no exception. Yes, we are a GNU project -- but, MORE IMPORTANTLY, we are a group of VOLUNTEERS spending our ever-fleeting time -- time that, perhaps, should be spent with our families -- writing this hopefully valuable suite of software.
>
> The upcoming release was originally planned for this past weekend. James and I decided to postpone the release by at least a week to generate useful (both technical and non-technical) documentation -- and that is just what we have been doing! Our sole intention was to generate as much useful

documentation as quickly and painlessly as possible.

LyX was chosen because it is usable and, more importantly, installable. After many failed attempts at installing the requirements for docbook, James and I made the decision that LyX-based documentation with the upcoming 0.1.0 releases was better than no documentation at all. **(Informal Management-1)**

Also, please note that the LyX format is internal to the project. Both Text and HTML (and hopefully PDF) snapshots of the documentation will be shipped with the software and also made available online. We will also accept any additions/corrections to the documentation in any format people will send. (Hint, hint :)

Sincerely,

JCater

PS, I hope all our American friends have a wonderful Thanksgiving. And to everyone else, well, you have a wonderful Thanksgiving as well.

PPS, By the way, Daniel, using/writing Free software is NOT about making RMS happy or unhappy. He's a great guy and all, but not the center of the free universe, nor the motivating factor in many (most?) of our lives. For me, my motivation to be here is a free future for my son. **(Belief in Freedom-2) (Motivation – Belief in Freedom)** So, please, don't question our convictions just because of some non-essential piece of the puzzle ... we are here giving freely of ourselves. Better yet, don't question our convictions at all -- they are, after all, a personal thing.

Next jcater discusses the problem with derek before chillywilly comes onboard.

<jcater> chilly's gonna be **** in the morming
<derek> ?
<jcater> I replied to his email
<jcater> that's what the
<derek> uh oh
Action: derek still never read the original
<jcater> derek, please forgive me
<jcater> was about
<jcater> don't get me wrong... I see where he's coming from
<derek> no problem
<derek> btw: you missed the key point :) **(Moderate BIFS-6**
<derek> “think it is Strongly **** that a GNU project would require me to
<derek> install non-free software in order to read and modify the documentation.”
<derek> is a NON TRUE statement
<derek> since the docs are distributed in html, ps, pdf etc they are READABLE by MANY non free

tools
<derek> since the output is latex
<derek> the modification is a lie too :)
<jcater> yes
<derek> one could edit the files in emacs just fine and produce diffs etc
<jcater> well, the last paragraph alluded to your first point
<jcater> actuall jamest and I told him that this morning
<derek> now its a bit more DIFFICULT than if they had lyx but not impossible :)
<jcater> that email just really, really, REALLY **** jamest and I off
<jcater> it's one thing to **** and moan in IRC
<jcater> but that was unnecessary
<derek> agree, we all have our bad days **Moderate BIFS-6)**

Later chillywilly comes back and more discussion takes place regarding lyx usage. Chillywilly first broaches the subject of jcater's response to chillywilly's email regarding lyx.

Action: chillywilly sees jcat **** and moaning about my mail.
<chillywilly> jcater
<ajmitch> no, it doesn't look like much of a **** & moan to me
<jcater> huh?
<chillywilly> irc logs shows otherwise

Chillywilly sticks to his diehard view without any interest in switching to non-free software.

<chillywilly> editing lyx files with a text editor is not acceptable **(Debate over non-free software-2**
<jcater> it's no different
<jcater> than editing docbook
<jcater> with a text editor
<chillywilly> docbook is a standard format
<chillywilly> lyx uses its own markup
<jcater> no
<chillywilly> which is not documented anywhere
<jcater> lyx is LaTeX
<chillywilly> no!
<jcater> have you looked at a LyX file??????
<chillywilly> irtfm
<jcater> um
<chillywilly> of course
<jcater> I did too :)
<chillywilly> it is not 100% LaTeX
<chillywilly> it is psuedo lyx tags
<jcater> oh, you're right
<jcater> the first line is
<jcater> \lyxformat 218
<jcater> damn
<jcater> that made it hard to understand

<jcater> the rest of the latex
Action: jcater is not gonna get into this discussion.. I think I made my point well in the email
<ajmitch> sigh
<chillywilly> well I haven't read my mail yet
<chillywilly> I still think it is ****
<chillywilly> if you want LaTeX then why not texinfo?
<jcater> drop it man
<chillywilly> dude, if someone wants to go with the standard packages distro stuff and be able to modify the docs easily then they must use non-free libxforms to get lyx working
<Mr_You> temporarily?
<Mr_You> I realize thats a problem, but the plan is that its temporary... AFAIK
<Mr_You> can we agree on atleast that chilly?
<Mr_You> I think everyone is on your side, its just a matter of timing as a result of relying on other Free Software
<chillywilly> can I view the current forms tech refernce?
<chillywilly> is it on the web?
<Mr_You> can we not provide translated documents?
<Mr_You> ie. one method for author (which is temporary) but translated to a more universal method of reading?
<Mr_You> s/author/authoring/
<chillywilly> this is what is mist pissiing me off....and that crack about RMS was really **** jcater as all I said was that I like to not run non-free software and I use vrms to make sure I am not using it
<chillywilly> last time I had to install lyx to even read the current forms tech ref
<chillywilly> that was not cool
<Mr_You> so what will it take to make it translated?
<jcater> Mr_You: that's what we do... we don't actually distribute the Lyx... we convert to text, ps, pdf, and html for the distros
<Mr_You> so chilly, just create your docs how ever you want, if thats what you want to do
<chillywilly> guess I should just downgrade myself to a user? **(Strong BIFS-11**
<chillywilly> Mr_You: that would require installing non-free software to do that easily
<chillywilly> that's the whole point
<chillywilly> I don;t see you guys committing any formats that would be useful for those of us wishing to keeo our system free of non-free software **Strong BIFS-11)**
<Mr_You> uggghh
<Mr_You> lag
<chillywilly> I know that you don't distribute it as lyx
<chillywilly> duh
<Mr_You> chilly, I think you're making this too big a deal, AT THIS TIME...
<Mr_You> you haven't been able to convince anyone that another software package is as useful AT THIS TIME **Debate over non-free software-2)**

Finally, chillywilly is convinced to drop the issue for now.

(Conflict Resolution
<chillywilly> well because I wanted to look at the forms form a higher level without diggin into code and this wa s abarrier

<chillywilly> even I had problems with exporting the docbook
<Mr_You> as time goes on, we can move to another solution
<Mr_You> why is that such a problem?
<Mr_You> I realize it goes against your philosophy.. but philosophy shouldn't get in the way of progress if it is a temporary issue **(Moderate BIFS-7**
<Mr_You> its just a minor temporary issue in the huge scheme of things
<Mr_You> I don't think it threatens our integrity
<chillywilly> why not just do it in text and then mark it up later then everyone csn read development docs without the B.S.
<Mr_You> go for it chilly.. you have your valid reasons
<chillywilly> Mr_You: the easy way to do that would be to run lyx and copy and paste I will not install it again until I can run it easily with a Free GUI
<Mr_You> you just haven't been successfull in convincing others at this time, I have no doubt you may be able to in the future, as everyone agrees with you ideally but technically its a minor issue
<chillywilly> it is not *minor*...it makes us look bad **Moderate BIFS-7)**
<jcater> chillywilly: emails like what you sent make us look bad

Mr_You appeals to chillywilly to not let his philosophy impede progress and jcater suggests that constant bickering looks bad for the GNUe project. Chillywilly still insists that needing to install non-free software is a huge impediment to developers, yet finally he drops the issue.

....
<Maniac> lyx is GPL
<Maniac> says so on the website
<chillywilly> yes they are, but I am not going to us eit until then
<mdean> mcb: yeah - the Qt port is nearly complete it seems
<Maniac> http://www.devel.lyx.org/guii.php3
<Isomer_> I found bugs in LyX
<Maniac> looks liek their getting close
<Mr_You> mdean: chilly doesn't like QT tho ;-)
<Isomer_> which I think lies in xforms
<Maniac> Isomer_: just incomplete features
<chillywilly> but anyway **** I don't wanna talk about this **** anymore...I hate QT it is **** ugly

Mdean appeals to the freedom of choice here to appease chillywilly.

<mdean> chillywilly: you have a choice - which is what is *really* important
<Isomer_> it doesn't help me if I can't look at the code to figure out why and to fix it
Action: Maniac is back (gone 04:46:27)
<chillywilly> mdean: I choose GNU whenever I can
<chillywilly> that is *my* choice
<Mr_You> sorry your choice is a frustrating one
<chillywilly> bah, I need a break from this **** ...time for some GNUe hacking....muuwahahahaaha
<chillywilly> Mr_You: eh?
<Isomer_> :)
<chillywilly> Mr_You: that is totally your opinion

<Mr_You> chillywilly: was sympathizing with you
<chillywilly> uh?
<Mr_You> empathizing?
<chillywilly> oh duh
<Mr_You> your choice appears to have created frustration
<chillywilly> whatever man I am burnt to a crisp
<chillywilly> me? them? who?
<chillywilly> stop conduzzzilating me :P
<Mr_You> for you mainly
<mdean> well, I for one am happy that I can run KDE or GNOME or Windowmaker or whatever - same for any productivity software **(Moderate BIFS-8)**
<mdean> it is all excellent software IMO **Conflict Resolution)**

This concludes the presentation of data for case study two. As chillywilly reflects on his frustration with losing the battle over the use of a non-free tool for documentation ("whatever man I am burnt to a crisp"), mdean exclaims that unlike chillywilly, he is happy to run KDE (even if it occasionally requires the use of non-free software) because it is "all excellent software IMO". This case shows that GNUe contributors vary in the strength of their **belief in free software.** The next section presents a discussion of the data from case two.

9.1 Discussion of Case Two

This case explores a GNUe insider's review of the procedures and practices for developing GNUe documentation. Figure 5 captures the sequence and configuration of relations among the observed variables identified in Section 7 and referenced in the data presentation.

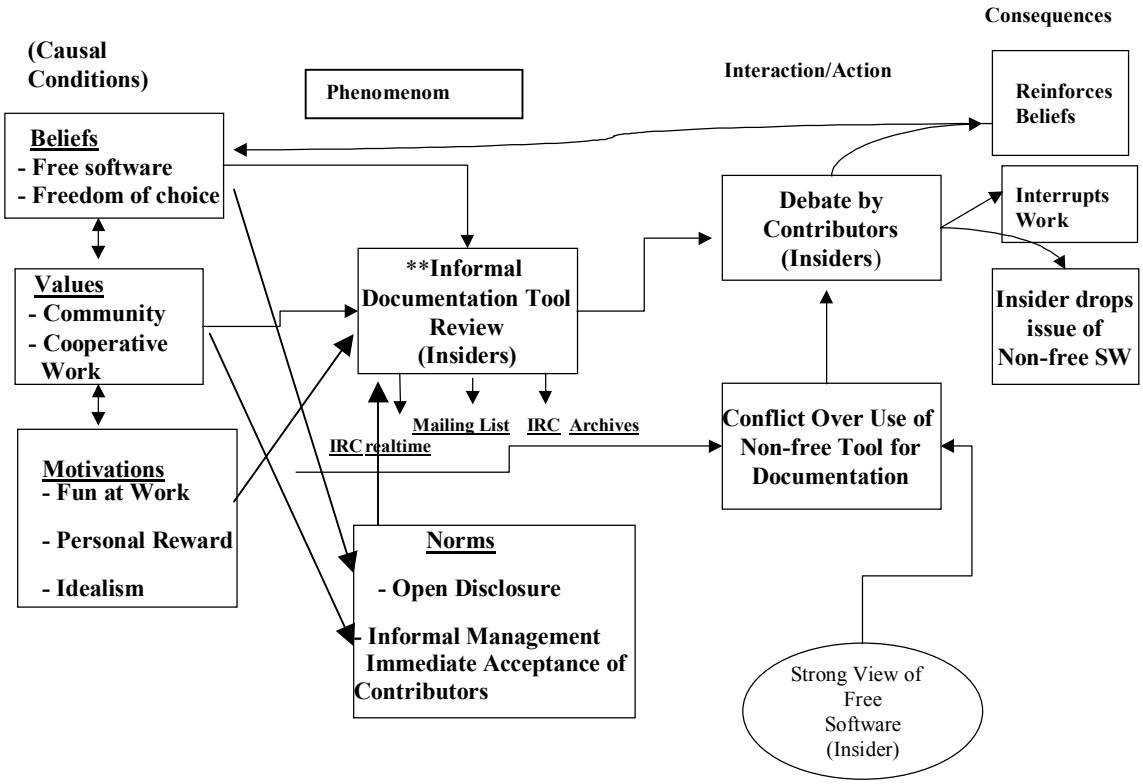

Figure 5 – A Schematic Summary of Relations Among Observed Variables in Case Two

In this case, chillywilly, an insider to the GNUe community, causes a stir by demanding an explanation for the use of lyx, which requires a download of non-free software to be operable. Chillywilly initiates the debate by asking fellow contributors on the IRC:

> "lyx requires non-free software... should that be acceptable for a GNU project? **(Strong BIFS-5).**

His colleagues respond with comments regarding their development infrastructure that for some includes Windows 98. Chillywilly continues his tirade over the course of three days eliciting support to replace the documentation tool with something that is based solely on free software. On the IRC and later on the mailing list, chillywilly refers to RMS, the founder of the Free Software Movement, as a reason for refusing to install lyx with its required non-free software. His IRC comment shows the reverence placed on RMS as the cultural leader:

> "I will NOT install lyx and make vrms unhappy **(Support for FSF leader – RMS-1)**."

He more explicitly states this in the email:

> "I mean if I cannot make vrms happy on my debian system then what good am I as a Free Software developer? **(Support for FSF leader – RMS-2)**"

The debate becomes quite heated between chillywilly and Jcater, a core maintainer. Jcater responds to chillywilly's email with an apology to contributors for how the IRC conversation got out of hand:

> "I would like to personally apologize to the discussion list for the childish email you recently received. It stemmed from a conversation in IRC that quickly got out of hand. **(IRC facilitates debate-1."**

Later Jcater mentions what his motivation is in creating free software:

> "For me, my motivation to be here is a free future for my son **(Belief in Freedom-2)."**

This persistent recording and recitation of motivations and beliefs reinforces the organizational cultural beliefs and values. While not all agree with chillywilly's strong position, some do side with him:

> "<amitch> I think having to use lyx is wrong for a GNU project **(Strong BIFS-7)."**

However, people like amitch are willing to use non-free software to continue to make progress on the GNUe "free" software development.

The issue is resolved by chillywilly dropping it after many contributors harangue him for wasting valuable time on what they consider to be a relevant, but temporary issue. Once free software is available for documentation, the GNUe team plans to use it. So chillywilly is finally mollified and no longer brings it up on the IRC or mailing list. As in case one, an outcome of the discussion is a reinforcement of the **belief in free software**. The amount of time to resolve the conflict in case two is longer than in case one - three days instead of one. There is no change to the documentation procedure as a result of the conflict (in contrast to case one where the Website graphic was changed). The **value in cooperative work** and **value in community** is evident in these exchanges. The conflict is mitigated and resolved over the IRC while building community at the same time.

10 Comparison of Two Case Studies

We have presented two cases of GNUe software development processes – both involving conflict which is mitigated and resolved over the IRC. Next we compare these cases across the causal conditions listed in Figure 6 and highlight the differences in consequences. Both examples show the power of the **belief in free software** and **belief in freedom of choice** that influences the statements, actions, and decisions of GNUe contributors. In both cases, a conflict arises due to strong beliefs in the use of free software for GNUe software development and documentation and the conflict is resolved via interaction among GNUe contributors on IRC and mailing lists.

However, the **consequences** in both cases differ significantly in **length of time to resolve** and in **modifications to code or procedures**. In both cases the debate results in **reinforcement of cultural beliefs in free software.** In the first case when an outsider instigates the debate, the

discussion lasts one day with an agreed-upon resolution that the graphic will be altered. In the second case, when the discussion involves a group of insiders being criticized by a fellow frequent contributor, the debate roars on for three days until chillywilly, the instigator, backs down over peer pressure. While other work is accomplished on the IRC during that period, the work of chillywilly and others is delayed to some degree as they argue about the use of a free versus non-free tool for documentation. During the discussions, several contributors become angry with chillywilly for causing such commotion over the use of lyx to produce documents and complained that they could be doing useful work instead. There appears to be less resistance to change regarding free versus non-free tools when the change involves an impact to one person's work and is not of a procedural nature.

In both cases, the **values of community building** and **cooperative work** are evident. In the first case, the online contributors (even those lurking on the IRC but not actively participating) contributed suggestions for free tool choices to create the graphic. In the second case, cooperative work was impaired by the incessant complaints of chillywilly regarding the installation of non-free software to reproduce the documentation. However, several frequent contributors banned together to convince chillywilly of the futility and impractical nature of his demand for exclusive use of non-free software to create documentation.

The **norms of open disclosure** and **informal management** combine to create an atmosphere that facilitates debate and conflict resolution. The **informal management** in both cases permits the interrupted work on the IRC to debate the issue of whether it is acceptable to use non-free software on a GNU-type project. Discussions on the IRC are monitored to some degree (like not allowing excess foul language) but there is no manager attempting to resolve the conflict so in the second case, the discussion led to some vituperative comments on a mailing list. There is basically no one there to quiet the ruckus and demand that people go back to work. The norm of **immediate acceptance of outsiders** influences the course of case one and its resolution of conflict. The outsider's opinion regarding the non-free use of a graphic tool is respected by all contributors.

There are two **intervening conditions** that influence the flow of the IRC debates. One is the **difference in time zones.** The core contributors are located across the world and establishing meetings on IRC can be difficult. The other is **the problem with netsplits** that occur throughout the day. Netsplits cause some deterioration of discussion depending on the length of time and amount of people involved.

In both cases, the **open disclosure** of the work encourages constant critiques of code and documentation. The similar beliefs and value in community bind the contributors together so that working in a non-collocated environment is not an impediment to completing the work. Surprisingly, only a few of the GNUe core maintainers are being reimbursed for their work. Currently, there are six companies listed on the GNUe website (http://www.GNUe) who pay consultants to support GNUe. Yet the website and list of companies using GNUe keeps

growing. Motivations are based on **idealism** regarding free software and **fun at work** due to the

sheer pleasure in programming and the comradarie formed with other GNUe contributors. In addition, some are motivated by **personal reward** in the form of potential opportunities for career advancement by providing consulting services to those companies who download and implement GNUe software.

This research indicates the importance of recorded logs of instant messages for resolving conflicts in cooperative work environments such as virtual work communities. Results have implications for CSCW software developers described in the next section.

11 Implications for CSCW

Conflict is an integral part of cooperative work in many work settings (Easterbrook, 1993). In a non-collocated virtual organization whose purpose is software development, the development work may be globally distributed with no tangible handle on the exact status of the project at any particular point in time (Scacchi, 2002). Conflict and its management in such a collaborative environment is an important aspect of open software projects. We broaden the discussion of conflict in CSCW presented in (Easterbrook, 1993) in terms of the two categories: a)Causes of Conflict and b)Resolution and Management of Conflict. Finally, we present our own assertion regarding Resolution and Management of Conflict based on this research.

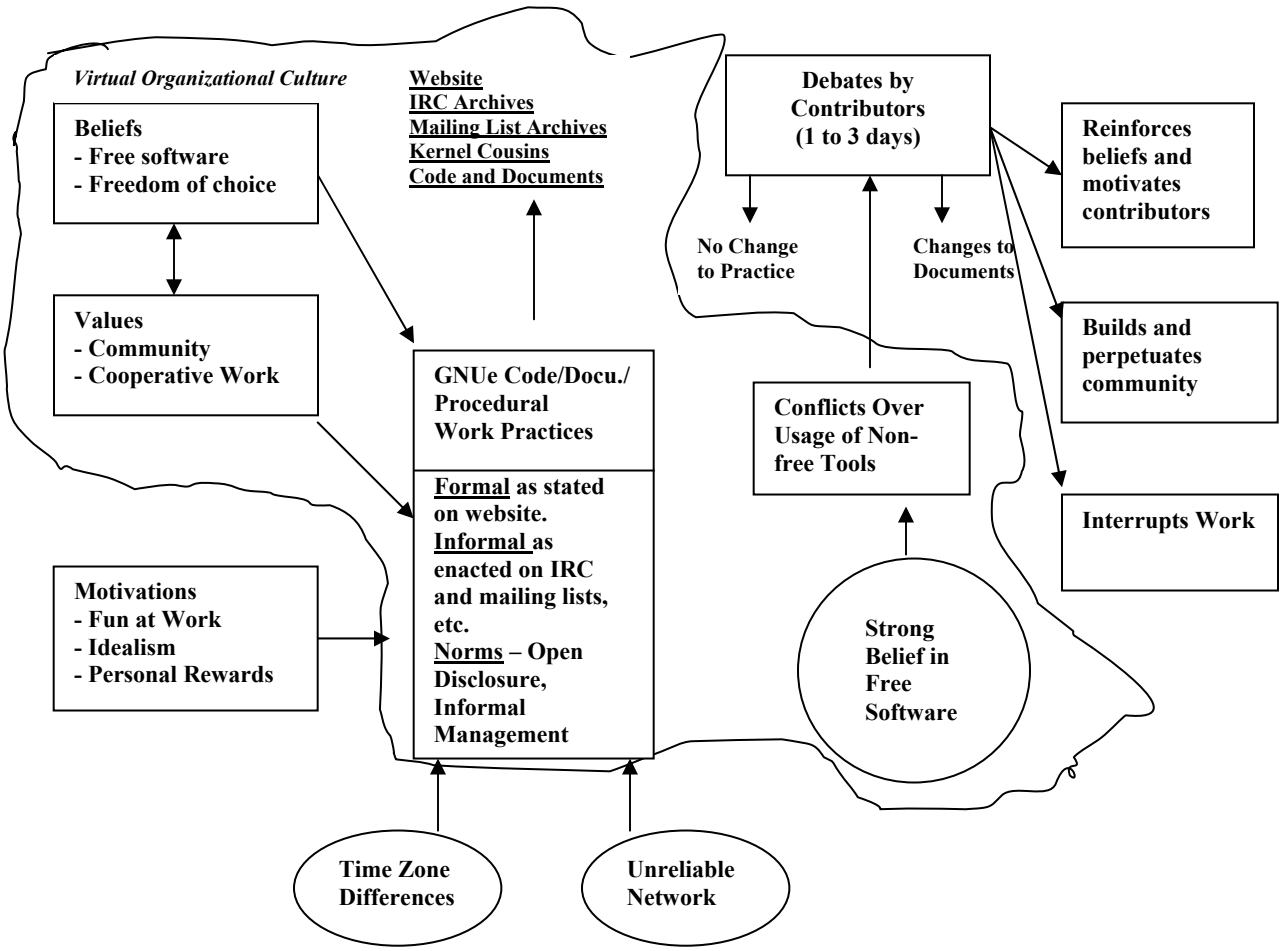

Figure 6 – A Schematic Summary of Relations Among Observed Variables in Both Cases

11.1 Causes of Conflict

This category refers to particular causes of conflict that arise in group work. Our research *refutes* the assertion related to the fact that anonymity and physical separation contribute to conflict.

11.2 Assertion A (*Refuted by This Research:* Anonymity and physical separation contribute to conflict.

Easterbrook *et al.*(1993) discuss several studies focusing on the effects of anonymity on interactions between people in CMC. These studies indicate that de-individuation occurs when social cues that distinguish individuals are missing. The effect on people is less of a sense of individuality among group participants, detaching individuals from his or her comments which can lead to a reduction in normal restraints on behavior (Jessup *et al.*, 1990). Another study showed that e-mail reduces social context cues encouraging people to behave irresponsibly more often with a focus on themselves not others (Sproull and Kiesler, 1986). Lea and Spears (1991)

criticized the Sproull and Kiesler study suggesting an alternative explanation for the de-individuation effect of CMC. They suggest that the social context influences the occurrence of de-individuation. If de-individuation occurs with immersion in a group, then this enhances the salience of the group and strengthens its norms. However, if the group identity is not already established, then de-individuation only serves to strengthen one's sense of individuality, and weakens group norms.

The research in our study supports the conclusion of Lea and Spears in that de-individuation occurs with immersion of GNUe contributors in a group effort creating strong community ties and strengthening its beliefs, values and norms. Results of this research also indicate that the value in community and cooperative work assist in maintaining the salience of the group. Although there is conflict that occurs in their daily work practices, their strong cultural ties diminish the effects of anonymity and distance separation. In addition, the frequent contributors are not really anonymous since their names and addresses are listed on the GNUe website.

11.3 Resolution and Management of Conflict

We find that certain conditions are necessasry for conflict to be easily managed and resolved at a distance – articulating conflict on the IRC and recording debates as archives accessible from a Website. Our research *refutes* one of Easterbrook's assertions in this category: Assertion B) ***Conflicts are unlikely to be resolved if particpants argue from entrenched positions,*** and *supports* another: Assertion C) ***Articulating conflict helps in its resolution***. Each assertion is described below.

11.3.1 Assertion B (Refuted by this research): Conflicts are unlikely to be resolved if particpants argue from entrenched positions

Easterbrook *et al.* (1993) argue that if participants become entrenched in their opinions, it becomes difficult to explore the middle ground. This may happen when participants have opposing basic beliefs, values or principles that they believe must be mutually exclusive. Pace (1990) referred to this type of conflict as "competitive conflict" characterized by defensiveness, hostility and escalation. This is in contrast to "cooperative conflict" which is positive and characterized by peace-keeping efforts. The recommendation for handling competitive conflict is to separate people from their positions to help support creativity in resolving the conflict. However, this is not always possible or desirable especially in a virtual community.

Our research indicates that the arguments made for the exclusive use of free software come from deeply entrenched positions in the cultural beliefs, values and norms of free software development. However, in debating appropriate choices for action, the GNUe participants were able to resolve their conflicts - in case one, by redoing a graphic in free software, and in case two, by dropping the issue and accepting the group norm. In a free software project like GNUe, there are no managers who monitor or attempt to resolve conflicts, so contributors with strong views use the IRC medium along with mailing lists to debate issues with fellow contributors. The **values in community** and **cooperative work** motivates people to try and resolve differences without a third party/manager. In the case of an outsider, others listened and the creator of the non-free graphic changed it using free software. In the case of an insider criticizing a standard procedure, the entrenched position of chillywilly *did* delay the resolution of the conflict.

However, beliefs in cooperative work were evident in the persuasion of fellow GNUe contributors and finally, in the interests of resuming GNUe work, chillywilly dropped the issue.

11.3.2 Assertion C (Support by our research): Articulating conflict helps in its resolution.

Research has shown that groups who talk about a task will perform significantly better than those that do not in the areas of group cohesiveness, commitment to task, and productivity (Easterbrook *et al.*, 1993). Pace (1990) uses the term "differentiation" to refer to the group process of identifying and understanding the dimensions of a conflict. This involves making the conflict explicit, identifying issues involved, and recognition of individual views by other members of the group. Pace concluded that differentiation of depersonalized conflict was important towards group consensus and cohesion and that on the other hand, a thorough differentiation does not ensure that consensus will be attained. He also found that a prolonged differentiation can damage personal relations in the group. Our two examples of conflict in GNUe support this assertion. In example one, the conflict was depersonalized since the criticizer and the creator of the graphic did not "know" each other in a virtual or real way. Their conflict was articulated over a day-long period and then finally, resolved by the agreement to use non-free software. Case two is an example of a prolonged differentiation where relations in the group can suffer (i.e. the vituperative nature of chillywilly's complaints and email). However, although no change was made to the choice of documentation tool, chillywilly dropped his complaints and the issue was resolved by no procedural change.

11.4 Assertions from the GNUe Research

Previous CSCW research has not addressed how the collection of IRC messaging, IRC transcript logs, and email lists, and periodic digests (kernel cousins) can be collectively mobilized and routinely used to create a virtual organization that embodies, transmits, and reaffirms the cultural beliefs, values, and norms such as those found in free software projects like GNUe. We assert the following conclusions derived from our research supporting this issue:

11.4.1 Text-based CMC in the form of IRC instant messaging streams, persistent IRC logs, digests (kernel cousins), and mailing lists enhances management and resolution of conflict in a virtual work community.

Results from this study indicate that when someone like a newcomer or frequent contributor generates a debate on the use of free versus non-free software for GNUe development, the articulation of strong (somewhat polarized) views of free software strengthens the community. Anyone can come onboard in the middle of a debate, review the previous day(s) worth of IRC and/or mailing list, and fairly quickly come up to speed on the issues. This fast access to archived information perpetuates the cultural beliefs that have been articulated and assists in the management and resolution of the conflict.

11.4.2 *Strong organizational cultural beliefs in a virtual community tie a group together so that conflict is more easily managed and resolved.*

The beliefs in freedom, free software, and freedom of choice create a special bond for the people working on free software projects. These beliefs foster the values of cooperative work and community-building. Schein's (1985) theory of organizational culture includes the revelation of

underlying assumptions of cultural members that are on a mostly unconscious level. In the GNUe world, the underlying assumptions of cooperative work and community-building become engrained in the everyday work practices in their pursuit of a ERP system implemented as free software. These beliefs and values enhance and motivate management and resolution of conflict despite the distance separation and amorphous state of the contributor population.

12 Conclusions

We have shown how the organizational cultural beliefs and values of a free software virtual organization influence software development processes. Our examples illustrate the importance of personal motivation and a sense of working as a team in the perpetuation of a virtual work community. Cultural beliefs and values combined with motivations directly influence the processes of free software development. Table 6 illustrates the summary of the Integration perspective applied to the GNUe Virtual Organizational Culture. As with most integration studies, the matrix shows evidence of consensus-building and consistency across the practices and artifacts columns. In addition, the beliefs and values are consistent with each other – all working in concert to form the ideology that promotes and perpetuates the free software movement and its many communities. For example, the belief in free software supports the values in community and cooperative work since contributors to free software need to work cooperatively in order to maintain the community of developers working at a distance. Each content theme is discussed below in terms of its cultural manifestations.

Belief in Free Software. Both formal and informal practices support the belief in free software but slight variations occur in the instantiation of this belief in everyday work practices. As a formal work practice, the GNUe software is being developed under the GPL and developers who join the community as a contributor must agree to abide by its premises. The general policy established by the FSF is for free software developers to use free software whenever possible (i.e. use a Linux/GNU operating system for other free software development). Informally, as evidenced in the case studies, some GNUe developers combine free and non-free tools in order to complete the software development. Similarly, GNUe is being developed for both Linux/GNU and Windows PC platforms. The GNUe project promotes the norm of open disclosure of all code, IRC discussions, digests (kernel cousins), mailing list discussions, and current software development assignments. The standard methods of communication in the GNUe project are GNUe mailing lists - used mainly for technical issues, and the IRC - dedicated to technical and social issues. Often people post code to the IRC asking for help with a bug or design issue. The IRC is also used for socializing (lots of joking around) and repeated contacts with fellow contributors. As in most free software project, the majority of GNUe contributors are volunteers but a few are being supported by consulting firms or become consultants to businesses implementing GNUe software.

Belief in Freedom of Choice. The belief in freedom of choice is manifested in the flat structure to the GNUe organization. There are several core maintainers who take on major tasks and work on much of the organizational and promotional issues related to the GNUe software development but there is no "top dog" making decisions for the rest of the organization. Formally, people download the latest GNUe releases for upgrades via CVS. Informally, people might take on GNUe assignments from "talking" on the IRC when GNUe developers ask people to test their

software modules prior to official release. Electronic artifacts that support this freedom of choice are the DCL software which enables people to select available assignments and report results via a Website. By the nature of a virtual organization, people who volunteer to do the GNUe work use self-selected computer equipment and software.

Value in Community. Formally, the GNUe project encourages volunteers of any ability level. Our research revealed that when newcomers log on to the GNUe IRC and point out a bug or offer a bug fix, they are easily welcomed into the community. Newcomers appear to become a member of the community fairly quickly, especially if they show advanced programming skills. Realtime impromptu code or documentation reviews on the IRC are acceptable by GNUe participants even if whatever they are working on at the moment is interrupted.

Value in Cooperative Work. The value in cooperative work is instantiated in the flat management structure with organic assignments from a non-profit organization. The mitigation and resolution of conflict over the IRC by GNUe participants show their willingness to cooperate over distances to complete the project. GNUe contributors work very hard as a team to solve technical and social issues using the archived IRC logs and kernel cousins to keep abreast of incidents requiring cooperative work.

Table 6 GNUe Virtual Organizational Culture

Content Themes		Practices		Artifacts
Espoused	**Tacit**	**Formal**	**Informal and Norms**	**Electronic Artifacts**
Belief in Free Software		GPL license required for all free software. Policy to use free software tools used for free software development. Use of CVS for code modifications and releases. Volunteer work (no pay) to produce software.	Combined use of free and non-free software for free software development. ***Open disclosure*** of all code and documents. Mailing list for work (technical issues) and IRC for social (***Having fun***) and work issues. Consulting work for profit to businesses using GNUe.	Free downloadable source code and documentation Archived IRC logs, Kernel cousins.
Belief in Freedom of Choice		Flat structure to management (no top person in charge). Assignments selected from website list.	***Informal management*** of people completing assignments. People inquire and take tasks from "talking" on IRC.	Use of home or other self-obtained computer and self-selected software for development. Double Choco-Latte (in-house project management)
	Value in Community	***Immediate acceptance of new contributors*** without a resume or reference check.	Contributors motivated to build and preserve free software community **Informal/Impromptu realtime code and documentation reviews.**	Sign-up on website. Archived IRC logs. Kernel cousins
	Value in Cooperative Work	Flat Management structure; organic assignments; non-profit organization	**Mitigation and resolution of conflict over IRC;** Working together to find real-time solutions	Archived IRC logs, Kernel cousins

This research indicates the importance of recorded logs of instant messages for resolving conflicts in virtual work communities. Results have implications for software developers and managers who plan to start an open source project with similar global temporal collocation and virtual communication characteristics.

The conclusions from this study include:

- The recording, publication, archiving, and subsequent referencing of IRC transcript logs assists in conflict resolution. Debates over non-free versus free software are recorded permanently for constant review. The recording and public distribution of the IRC logs contributes to the persistence and reiteration of cultural beliefs and values.
- The free software movement and the FSF, both with RMS as the founder, have generated and continue to be a source of inspiration and cultural beliefs, values, and norms for free software development projects like GNUe. GNUe can be considered a subculture of the organizational culture of the FSF and as such, they have cultural beliefs in common with the FSF and some that have developed as unique to the GNUe work culture.
- There is a tension between believing in free software and completing the work in a timely fashion. As a result of this tension, we see contributors who are so adament about the exclusive use of free software that their own work on the project as well as the work of those who are interrupted is delayed.
- Outsiders (lurkers or first time contributors) or occasional contributors can instigate and mitigate organizational conflicts. Source code, documentation, and procedural reviews are welcomed and respected as valuable contributions without minimal identification of an outsider's credentials. In a typical organization, a newcomer might take some time to become acclimated to the organizational culture and "the way to do things around here" such that he or she feels accepted in the community. However, in this world of free software development projects like GNUe, a newcomer's review of code and documentation is immediately considered worthy of a response.
- The beliefs in free software and freedom of choice in work combined with the values of community and cooperative work bind the GNUe together to influence decisions regarding tool choice and management of contributors' work.
- Despite the work without pay, the GNUe community of free software developers thrives and produces reliable, useable business software that is being actively used by about 15 companies.

We have shown how the community spirit in the forum of collective beliefs, values, and norms of the GNUe project fosters collaboration and resolution of conflict. To further the cause of free software, many contributors work with no monetary compensation to develop a free software package. The informal management and open disclosure facilitate the impromptu reviews and debates, yet result in a delay in actual completion of the software development. At the same time the longevity of the project is improved by the persistent recording and distribution of GNUe debates over free versus non-free software and by the camaraderie established by becoming a member of the GNUe virtual culture.

It is important for future managers of open source projects to pay particular attention to the beliefs and values of open software developers. Those with strong beliefs in free software may react differently to management directives involving the use of non-free software than others with moderate beliefs. We also discovered that while most open source communities use mailing lists for intergroup communication, the use of IRC can be a rich medium for software development in non-collocated environments. In future work, we will continue the analysis of GNUe by comparing the two cases involving conflict to a no-conflict software review. Future work also involves comparing the theory derived in this research with other open software projects in the area of the game world and academic computing research.

Researchers have referred to the development of open source software as gift-giving of a public good (Raymond, 2001; Kollock, 1999). To preserve the presence of such an online community, several structural features are important (Kollock, 1996): ongoing interaction, identity persistence, and knowledge of previous interactions. For the perpetuation of a free software development community, we would add to that list, the fruition and persistence of rich cultural beliefs and values in the work itself.

References

Ackerman, M., & Halverson, C. (2000). Reexaming Organizational Memory. *Communications of the ACM, 43*(1), 59-64.

Ackerman, M., & Scacchi, W. (1999). ITR/SOC: Understanding Open Software Communities: Processes and Practices: A Socio-Technical Perspective: National Science Foundation Proposal.

Avison, D. E., & Myers, M. D. (1995). Information Systems and Anthropology: An Anthropological Perspective on IT and Organizational Culture. *Information Technology and People, 8*, (43-56).

Bergquist, M., & Ljungberg, J. (2001). The Power of Gifts: Organizing Social Relationships in Open Source Communities. *Information Systems Journal, 11*(4), 305-320.

Bowker, G., Star, E., & Turner, W. (1997). *Social Science, Technical Systems, and Cooperative Work: Beyond the Great Divide*: Lawrence Erlbaum.

Brown, J., & Duguid, P. (1991). Organizational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning, and Innovation. *Organizational Science, 2*, 40-57.

Cooper, R. B. (1994). The inertial impact of culture on IT implementation. *Information Management, 27*, (17-31).

Crowston, K., & Scozzi, B. (2002). *Exploring Strengths and Limits on Open Source Software Engineering Processes: A Research Agenda.* Paper presented at the 2nd Workshop on Open Source Software Engineering, Orlando, Florida.

Davis, S. (1984). *Managing Corporate Culture*. Cambridge, MA: Ballinger.

Deltor, B. (2000). The Corporate Portal as Information Infrastructure: Towards a Framework for Portal Design. *International Journal of Information Management, 20*(2), 91-101.

Dempsey, B. J., Weiss, D., Jones, P., & Greenberg, J. (2002). Who is An Open Source Developer? *Communications of the ACM, 45*(2), 67-72.

DiBona, C., Ockman, S., & Stone, M. (1999). *Open Sources: Voices from the Open Source Revolution*. Sebastol, CA: O'Reilly & Associates Inc.

Dube, L., & Robey, D. (1999). Software Stories: Three Cultural Perspectives on the Organizational Practices of Software Development. *Accounting, Management and Information Technologies, 9*(4), 223-259.

Easterbrook, S. (Ed.). (1993). *CSCW: Cooperation or Conflict*. New York: Springer-Verlag.

Elliott, M. (2000). *Organizational Culture and Computer-Supported Cooperative Work in a Common Information Space: Case Processing in the Criminal Courts*. (Vol. Unpublished Dissertation). Irvine: University of California, Irvine.

Elliott, M. (2003, May, 2003). *The Virtual Organizational Culture of a Free Software Development Community.* Paper presented at the 3rd Workshop on Open Source Software, Portland, Oregon.

Elliott, M., & Scacchi, W. (2002). Communicating and Mitigating Conflict in Open Source Software Development Projects. http://www.ics.uci.edu/~melliott/commossd.htm: Institute for Software Research, University of California, Irvine.

Elliott, M., & Scacchi, W. (2004). Free Software Development: Cooperation and Conflict in a Virtual Organizational Culture. In S. Koch (Ed.), *Free/Open Source Software Development*: Idea Press.

Feller, J., & Fitzgerald, B. (2002). *Understanding Open Source Software Development*. N.Y.: Addison-Wesley.

Fielding, R. T. (1999). Shared Leadership in the Apache Project. *Communications of the ACM, 42*(4), 42-43.

Fielding, R. T., Whitehead, E. J., Anderson, K. M., Bolcer, G. A., Oreizy, P., & Taylor, R. N. (1998). Web-Based Design of Complex Information Products. *Communications of the ACM, 41*(8), 84-92.

Freericks, C. (2001). Open Source Standards on Software Process: A Practical Appproval. *IEEE Communications Magazine* (April).

Geetz, C. (1973). *The Interpretation of Cultures*. New York, NY: Basic Books.

Glaser, B., & Strauss, A. L. (1967). *The Discovery of Grounded Theory: Strategies for Qualitative Research*. New York: Aldine.

Gregory, K. (1983). Native-view Paradigms: Multiple Cultures and Culture Conflicts in Organizations. *Administrative Science Quarterly, 28*, 359-376.

Hahn, I. H., Roberts, J., Slaughter, S., & Fielding, R. (2002). *Why Do Developers Contribute to Open Source Projects? First Evidence of Economic Incentives.* Paper presented at the 2nd Workshop on Open Source Software Engineering, Orlando, FL.

Herbsleb, J. D., & Grinter, R. (1999). Splitting the Organization and Integrating the Code: Conway's Law Revisited. *21st International Conference on Software Engineering*.

Hine, C. (2000). *Virtual Ethnography*. London: Sage.

Jessup, L. M., Connolly, T., & Tansik, D. (1990). Toward a Theory of Automated Group Work: The Deindividuating Effects of Anonymity. *Small Group Research, 21*(3), 333-348.

Kling, R., & Iacono, S. (1984). Computing as An Occasion for Social Control. *Journal of Social Issues, 40*(3), 77-96.

Koch, S., & Schneider, G. (2000). *Results from Software Engineering Research into Open Source Development Projects Using Public Data*, Wirtschaftsuniversitat Wien.

Kogut, B., & Metiu. (2001). Open Source Software Development and Distributed Innovation. *Oxford Review of Economic Policy, 17*(2), 248-264.

Kollock, P. (1996). Design Principles for Online Communities, *The Internet and Society: Harvard Conference Proceedings also available at http;//sscnet.ucla.edu/soc/faculty/kollock/papers/design.htm*.

Kollock, P. (1999). The Economies of Online Cooperation: Gifts and Public Goods in Cyberspace. In M. A. Smith & P. Kollock (Eds.), *Communities in Cyberspace* (pp. 220-239). New York, NY: Routledge.

Kollock, P., & Smith, M. A. (1999). Communities in Cyberspace. In M. A. Smith & P. Kollock (Eds.), *Communities in Cyberspace* (pp. 3-25). New York, NY: Routledge.

Kotoyna, G., & Sommerville, I. (1998). *Requirements Engineering: Processes and Techniques*. New York, NY: John Wiley and Sons, Inc.

Lea, M., & Spears, R. (1991). Computer-mediated Communication, De-individuation and Group Decision-making. *International Journal of Man-Machine Studies, 34*, 282-301.

Mackenzie, A., Rouchy, P., & Rouncefield, M. (2002, February 25-26, 2002). *Rebel Code? The Open Source 'Code' of Work.* Paper presented at the Open Source Software Development Workshop, Newcastle-upon-tyne, UK.

Martin, J. (1992). *Cultures in Organizations: Three Perspectives*. Oxford UK: Oxford University Press.

Martin, J. (2002). *Organizational Culture: Mapping the Terrain*. Thousand Oaks: Sage Publications.

Meyerson, D., & Martin, J. (1987). Cultural Change: an Integration of Three Different Views. *Journal of Management Studies, 24*, (623-647).

Mockus, A., Fielding, R., & Herbsleb, J. (2002). Two Case Studies on Open Source Software Development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology* (To appear).

Mockus, A., Fielding, R. T., & Herbsleb, J. (2000). *A Case Study of Open Source Software Development: The Apache Server.* Paper presented at the ICSE '2000.

Monk, A., & Howard, S. (1998). The Rich Picture: A Tool for Reasoning About Work Context. *Interactions, March/April*, 21-30.

Nakakoji, K., & Yamamoto, Y. (2001). *Taxonomy of Open Source Software Development.* Paper presented at the 1st Workshop on Open Source Software Engineering at ICSE 2001.

Nardi, B., Whittaker, S., & Bradner, E. (2000). *Interaction and Outeraction: Instant Messaging in Action.* Paper presented at the Computer Supported Cooperative Work '00, Philadelphia, PA.

Noll, J., & Scacchi, W. (1999). Supporting Software Development in Virtual Enterprises. *Journal of Digital Informaion, 1*(4).

Olsen, M. (1971). *The Logic of Collective Action*. Cambridge, MA: Harvard University Press.

Olsen, M. H. (1983). New Information Technology and Organizational Culture. *MIS Quarterly, 6*(5), 71-92.

Olsson, S. (2000). *Ethnography and Internet: Differences in Doing Ethnography in Real and Virtual Environments.* Paper presented at the IRIS 23, Laboratorium for Interaction Technology, University of Trollhattan Uddevalla.

Orlikowski, W., & Robey, D. (1991). Information Technology and the Structuring of Organizations. *Information Systems Research, 2*, 143-169.

Ott, J. (1989). *The Organizational Culture Perspective*. Pacific Grove, CA: Brooks/Cole.

Pace, R. C. (1990). Personalized and Depersonalized Conflict in Small Group Discussions: An Examination of Differentiation. *Small Group Research, 21*(1), 79-96.

Pavlicek, R. G. (2000). *Embracing Insanity: Open Source Software Development*. Indianapolis, IN: SAMS Publishing.

Putnam, L., & Poole, M. (Eds.). (1987). *Conflict and Negotiation*. Beverly Hills: Sage.

Raymond, E. S. (2001). *The Cathedral & The Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. Sebastopol, CA: O'Reilly & Associates.

Robbins, J. E., Medvidovic, N., Redmiles, D. F., & Rosenblum, D. S. (2001). *Integrating Architecture Description Languages with a Standard Design Method* (Working Paper). Irvine, CA: Dept. of Information and Computer Science, University of California.

Robey, D., & Azevedo, A. (1994). Cultural Analysis of the Organizational Consequences of Information Technology. *Accounting, Management, and Information Technology, 4*(1), 23-37.

Samuelson, P. (1954). The Pure Theory of Public Expenditure. *Review of Economics and Statistics, 36*, 387-390.

Sawyer, S. (2000). Packaged Software: Implications of the Differences from Custom Approaches to Software Development. *Operational Research Society Ltd., 9*, 47-58.

Sawyer, S. (2001). Effects of Intra-Group Conflict on Packaged Software Development Team Performance. *Information Systems Journal, 11*, (155-178).

Scacchi, W. (2002). *Is Open Source Software Development Faster, Better, and Cheaper than Software Engineering?* Paper presented at the 2nd Workshop on Open Source Software Engineering, Orlando, Florida, May, 2002.

Scacchi, W. (2002). *Open EC/B: A Case Study in Electronic Commerce and Open Source Software Development* (Technical Report). Institute for Software Research, Irvine, CA: University of California, Irvine.

Scacchi, W. (2002). Understanding Requirements for Developing Open Source Software Systems. *IEE Proceedings - Software, 149*(2), 24-39.

Scacchi, W. (2002). *Understanding the Social, Technological, and Policy Implications of Open Source Software Development.* Paper presented at the NSF Workshop on Open Source Software, Arlington, VA, January 2002.

Schein, E. (1990). Organizational Culture. *American Psychologist, 45*, 109-119.

Schein, E. H. (1984). Coming to a New Awareness of Organizational Culture. *Sloan Management Review, 25*(2), 3-16.

Schein, E. H. (1991). The Role of the Founder in the Creation of Organizational Culture. In P. J. Frost, L. F. Moore, M. R. Louis, C. C. Lundberg, & J. Martin (Eds.), *Reframing Organizational Culture* (pp. 14-25). Newbury Park, CA: SAGE Publications, Inc.

Schein, E. H. (1992). *Organizational Culture and Leadership*. San Francisco: Jossy-Bass.

Schmidt, D. C., & Porter, A. (2001). *Leveraging Open-Source Communities to Improve the Quality & Performance of Open-Source Software.* Paper presented at the First Workshop on Open-Source Software Engineering, Toronto, Canada.

Schultze, U. (2000). A Confessional Account of An Ethnography About Knowledge Work. *MIS Quarterly, 24*(1), 43-79.

Sharman, S., Sugurmaran, V., & Rajagopalan, B. (2002). A Framework for Creating Hybrid-Open Source Software Communities. *Information Systems Journal, 12*(1), 7-25.

Site, C. S. W., & http://asc.harvard.edu/. (2000). Chandra Software Tools Information Site. http://asc.harvard.edu/udocs/docs/docs.html.

Site, S. A. R. W., & http://www.ics.uci.edu/pub/arch/. (2000). http://www.isr.uci.edu/events/wesas2000.

Smircich, L. (1983). Concepts of Culture and Organizational Analysis. *Administrative Science Quarterly, 28*, 339-358.

Smith, A. D. (1999). Problems of Conflict Management in Virtual Communities. In M. A. Smith & P. Kollock (Eds.), *Communities in Cyberspace* (pp. 134-163). New York, NY: Routledge.

Sproull, L., & Kiesler, S. (1986). Reducing Social Cues: Electronic Mail in Organizational Communication. *Management Science, 32*, 1492-1512.

Sproull, L., & Kiesler, S. (1991). *Connections: New Ways of Working in the Networked Organization*: The MIT Press.

Stallman, R. (1999). Free Software Foundation. Cambridge, MA: Free Software Foundation.

Stallman, R. (1999). The GNU Operating System and the Free Software Movement. In C. DiBona, S. Ockman, & M. Stone (Eds.), *Open Sources: Voices from the Open Source Revolution* (pp. 53-70). Sebastopol, CA: O'Reilly & Associates, Inc.

Star, S. L. (Ed.). (1996). *The Cultures of Computing*. Cambridge, MA: Blackwell Publishers.

Strauss, A. L., & Corbin, J. (1990). *Basics of Qualitative Research: Grounded Theory Procedures and Techniques*. Newbury Park, CA: Sage Publications.

Trice, H. M., & Beyer, J. M. (1993). *The Cultures of Work Organizations*. Englewood Cliffs, NJ: Prentice Hall.

Van Maanen, J. V., & Barley, S. R. (1984). Occupational Communities: Culture and Control in Organizations. *Research in Organizational Behavior, 6*, (287-365).

Watson, T. (1987). *Sociology, Work & Industry*. (2nd ed.). London: Routledge & Kegan Paul.

Wenger, E. (1998). *Communities of Practice: Learning, Meaning, and Identity*. Cambridge, Massachusetts: Cambridge University Press.

Williams, S. (2002). *Free as in Freedom: Richard Stallman's Crusade for Free Software*. Sebastopol, CA: O'Reilly & Associates.