Common Threads: Hierarchy and Communication in Open-Source Software

Colleen McKenzie

"Provided the development coordinator has a communications medium at least as good as the Internet, and knows how to lead without coercion, many heads are inevitably better than one." (Raymond, 2000).

Penned by Eric Raymond, founder of the public-benefit corporation that defines and licenses nonproprietary software, the above principle is the driving force behind the current success of community-based models of software design. These communities employ an all-inclusive, unrestricted, and collaborative organizational strategy to maximize the number of participants involved, and with no small success: the Apache web server platform supports over half of all websites on the internet, and the Mozilla Firefox browser has steadily risen in popularity to its current position of 46% usage share. Both are examples of open-source, community-based, free software (W3Schools, 2010).

To marshal the potential of such a diverse and and delocalized group of contributors, these communities must employ novel means of communicating, delegating tasks, and sharing their work; and because community-based software design emerged simultaneously with the internet, most of these groups' organizational strategies rely on virtual modes of communication. Further, some of these projects require a governing body to manage legal and macro-organizational matters due to the abundance of contributions. This paper will discuss how the structure and communication of the Ubuntu project compare to general strategies discussed by current literature on the open-source projects, and I argue that it is both the project's hierarchical decision-making model and its facilitation of mass communication that make it such a successful model of production.

Ubuntu's particular model of community-managed software is by no means representative of other open-source software, or even of projects governed by similar ideals. The term Open-Source can be used to refer to any piece of software for which users can download the source code free of charge; the source code for a program differs from the program itself in that it is readily understood by anyone proficient in the programming language it was written, while the program itself has already been modified into the much more complicated directions a computer would need to run it. The Open Source Initiative narrows the category of software it recognizes to open-source software that conforms to specific licensing criteria. These recognized open-source projects must be governed by licenses that make them freely redistributable, open to modification, and protected from more exclusive relicensing (OSI, 2010). The Ubuntu package includes both the operating system platform and other software that it supports, and while this supplemental software may or may not conform to standards in the Open-Source Definition, the Ubuntu system platform does. Yet a smaller category of recognized open-source software, and the one on which this paper will focus, are those projects that are developed, modified, and distributed by a community of independent individuals contributing their skills without any reimbursement from the governing body. Ubuntu is such a community-managed project, relying on input from a vast and scattered group of contributors, and governed by both boards of community members and a parent corporation, Canonical Ltd (Canonical Ltd, 2010).As I will subsequently discuss, this variety of inputs at even the higher levels of organization lets the project benefit from the suggestions of experts from the project's multiple departments, or "teams," of which there are many.

Although open-source projects and project structures are continually morphing as a result of their novelty, several studies have observed the organizational strategies these groups use to delegate and cumulate work on the project. These inquiries emphasize four aspects in particular of development and distribution: governance structure and hierarchy (where it exists), distribution, legal issues pertaining to licensing and dissemination, and production mechanics and dynamics. The structures of governance that have been reviewed seem to follow a basic model in which an elite governing group has a final say in what code will be used for the final project, but the code itself comes from a scattered but close community of developers, who are often divided into subgroups. In Kogut and Metiu's review of two community-managed software projects, both adhered to this model of governance, with slight idiosyncrasies: decision-making power over the Linux operating system is ultimately wielded by one man, Linus Torvalds, who founded the project, and "development for the Apache model is *federal*, based upon a meritocratic selection process" (Kogut and Metiu, 2001). Many of these projects are also funded and supported by a parent corporation, as in the case of Ubuntu. In O'Mahoney's study of six different community-managed software projects, five were incorporated as non-profit organizations, which allowed the projects to "protect volunteer contributors from individual liability, enter into agreements collectively, and protect their code, trademarks, licenses, and brand" (O'Mahoney, 2003). While these corporations aid more in the interactions between a project and the outside world, rather than within the project itself, they constitute a framework—and, often an ideology—around which the rest of the community can organize itself.

Product distribution among community-managed software projects seems to be easily divided into costless and priced models, but many if not most projects employ a combination of both models. Rosenberg calls this a "double-breasted" approach to distribution, and cites a number of companies it describes: Ajuba Solutions, which distributes a scripting language and tools supporting it; Larry Wall, creator of the scripting language Perl; and Digital Creations, which created the open-source server software Zope all distribute both free and priced versions of their software. Usually the buyer is paying primarily for ongoing support for the purchased product, as is the case with the Red Hat distribution of the Linux operating system, which "offer[s] support contracts for those that need long-term security" (Moody, 2001). Closely related to distribution decisions are concerns about licensing open-source products. O'Mahoney's study of a variety of such projects found that all but one project used the GNU Public License, or GPL, which is the de facto standard license for free software projects and was developed by the GNU project in 1986 to protect the operating system it created (Moody, 2001). The GPL is marked by its pervasive quality, which lies predominately in the following clause:

> "You [the user of the licensed program] must cause any work that you distribute or publish, that in whole or in part contains or is derived from the [licensed] Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License" (Rosenberg, 2000).

If a work is covered by the GPL, then, no derivative works can be licensed under more proprietary conditions.

Perhaps the most interesting aspect of community-managed software is the process of actually assembling the code for a program. Kogut and Metiu suggest that "the rapid growth of open-source development suggests that the traditional methods of software development are often inefficient," partially as a result of the benefits a large participating community can contribute (Kogut and Metiu, 2001). Both they and Pavlicek note that when source code is publicly available, "the community can see precisely how things are implemented as they are being implemented. It gives the community the opportunity to discuss the merits of using one type of implementation rather than another before anything has been cast in stone" (Pavlicek, 2000). Raymond's less formal version of this concept—"Given enough eyeballs, all bugs are shallow"—has become one of the central tenets of open-source enthusiasts, and is termed "Linus's Law," after the creator of the Linux desktop project (Raymond, 2000). This structure offers two benefits. First, it fosters discussion of the goals and directions for the project, encouraging what Lester and Piore would call a more "interpretive" problem-solving process resulting from the ability to "overcome barriers between producers and consumers" (Lester and Piore, 2004). Further, it allows for simultaneous execution of multiple production processes at once. This second benefit stands in marked contrast to the conventional, proprietary methods of production organization that Kogut and Metiu term the "factory model," in which production is carried out sequentially and by groups of workers assigned by merit to separate tasks. Such a model is a "weak link" chain, in which the entire process can be delayed by one unproductive worker or group, but the simultaneous model visible in community-managed projects allows a project to profit from productive members' input without being hindered by unproductive contributors' impedance. Implementing this collaborative process, however, is much more complicated than simply assigning workers to specific tasks.

The Ubuntu project tackles the problem of organizing the production, distribution, and improvement of its products by employing a variety of organizational structures both in keeping and in conflict with the general trends observed in other open-source software communities. In keeping with the model proposed earlier, the project has a governing group overseeing the project and a large community of contributors. This presiding body is further divided into three sections. The Technical Board is responsible for not only technical decisions about the product but also "key policy documents and standards" while the Community Council makes decisions regarding interpersonal communication and conflicts. The third branch is composed solely of Mark Shuttleworth, founder of Ubuntu's parent corporation, Canonical Ltd., who is referred to in official documentation as the "SABDFL," an acronym of "Self-Appointed Benevolent Dictator For Life," and who serves on both the Technical Board and the Community Council (Canonical Ltd, 2010). Both bodies comprise many of the most prolific, experienced, and involved members of the Ubuntu community, most of whom continue to contribute code to the project in addition to fulfilling their administrative functions, and member appointments (by Shuttleworth) are elected for both groups. While the Board and the Council have final decision-making power in conflicts within their respective areas, both have issued official statements which speak to a reluctance to exercise this power: the Technical Board states that "individual developers and teams are best placed to make day-to-day decisions, but when technical issues cannot be resolved between individual developers...these may be brought to the Technical Board for resolution," and the Community Council looks into interpersonal matters only "if a member of the community has asked the Community Council to review the behaviour of another" (Canonical Ltd, 2010).

Canonical Ltd, the public-benefit corporation founded to support and protect the Ubuntu project, among other open-source enterprises, also follows a governance strategy more hands-off that traditional proprietary software firms'. Shuttleworth has described the company as "entirely based on services around our software," and sees its role in the project's organization as follows:

> "Canonical plays a significant role in [Ubuntu governance], and we are the largest underwriter of all the work that gets done. We make sure that it releases on time; that it's available globally; that it meets

> criteria....But we don't take credit for all of the smart thinking that happens in Ubuntu." (Moody, 2008).

Further testament to Canonical's supportive, non-authoritarian role is its unprofitability. While not officially a non-profit corporation, Canonical's annal revenue of $30 million is just enough to cover the costs of supporting and publicizing its projects, supporting its slightly over 300 employees, and facilitating developer interaction through non-virtual conferences for project teams, and the company's CEO has confirmed in a March 2010 that the company is "not profitable now" (Vance, 2009; VAR Guy, 2010).

While there is a strong sense of hierarchy surrounding the separation of governing groups, there seems to be very little bureaucracy involved in communications between different levels of the project. According to Ubuntu contributor Jonathan Thomas, the governing bodies are anything but lofty: Community Council meetings are open to observation by anyone on the internet, and Jonathan has "seen a couple council meetings on [the online chat network] and they're actually pretty receptive to other people being there as long as you don't talk out of turn all the time" (Thomas, 2010). The Technical Board is "a little harder to get a hold of but from what I've heard they still cover all the stuff on the agenda, I don't know anyone who hasn't had something covered if they put it on there" (Thomas, 2010) Fellow contributor Rob Sobczak says of the technical board that "it's much more of a respect setup. They're more like really focused developers than anyone's boss" (Sobczak, 2010). This lack of detachment between different levels of involvement in the project—both Thomas and Sobczak are enthusiastic but peripheral contributors—allows the project to benefit from the expertise of participants at lower, more specialized positions. Shuttleworth confirms this ability with the evidence that "in almost every release there's been an idea that came from volunteer participants that turned into a profoundly important feature in that release" (Moody, 2008). Eric Raymond praises this type of open model in another pillar of open-source community management, "The next best thing to having good ideas is recognizing good ideas from your users. Sometimes the latter is better" (Raymond, 2000). Community-managed software projects have the added benefit of a significant overlap between contributors and users of the product: all five of the developers I interviewed were running the Ubuntu operating system, three as their primary operating system and two as an alternative to a commercial operating system. This overlap allows for a more recombinant approach to the feedback-revision model in which feedback is augmented by user understanding. Dave Thacker, who helps lead development on a subproject called Kubuntu that differs from Ubuntu only in graphical features, says of this input-focused paradigm that it "is definitely true [that it works]. It's the same idea as playing a song my band wrote for some friends of mine and getting their feedback, except that here most of your friends will end up contributing to the end product in some way beyond just feedback" (Thacker, 2010).

The divisions of the larger Ubuntu project into smaller teams supports this incorporation of feedback. The teams fall into eleven larger groups of between one and six teams, and each team has a specific area of the project to focus on: the Artwork Team, for example, "work on artwork and graphics for many different parts of the Ubuntu project," while the six Distribution Teams focus on unique operating systems based on Ubuntu, such as the Kubuntu distribution that Thacker works on. In addition to these field-oriented teams, there are many location-specific groups, called "LoCo teams" for Location Community, which function as both project-oriented and social groups. LoCo team density varies by country, but in general the more actively participating countries (such as the U.S.) have one or two LoCo teams per state, and less active countries have only one or two LoCo teams total (Canonical Ltd, 2010). According to California contributor Chris Brislaine, the LoCo teams often organize informal meet-ups that connect contributors in a less formal setting. Despite their casual nature, these events contribute to the development of the Ubuntu project by allowing developers to "get to meet a bunch of really awesome folks who you'd probably have things in common with even without free software but then you end up with this network of people you can swap ideas with about whatever feature [of the Ubuntu project] you're working on" (Brislaine, 2010). These friendly networks of contributors can include developers from multiple contrasting areas of the project and seem to provide similar benefits to those that Polodny and Page discuss is their review of inter-firm networks: developers can learn about different aspects of the project from each other and can attain higher reputations by associating with more skilled or connected members.

While these informal, non-virtual gatherings may connect developers in a more lasting fashion, most of the communication between contributors takes place in heavily mediated spaces. Ubuntu uses several different mediums of communication to collect and disseminate information, the most-used of which are its message boards, its mailing lists, its wiki, and the Launchpad collaboration site developed by Canonical specifically for use with Ubuntu (Canonical Ltd, 2010). Developers seem to use most, if not all, of these modes of communication, though some prefer certain modes over others. Most of the developers with whom Sobczak discusses the project rely heavily on the message boards and mailing lists and post to the wiki often, but many do not have an individual account set up on the Launchpad site, which Sobczak and Brislaine both feel is more common among the more committed and prolific developers (Sobczak, 2010; Brislaine, 2010). The three most commonly-used methods of communication allow the community to choose the mode best-suited to the information it relays and the impact that information makes. The forums stand out as the most versatile domain for correspondence. Because they are highly organized first by topic and then by discussion, they can be easily viewed and followed by people at any level of the organization and are thus an effective means of communicating developers' ideas not only within teams, but also both laterally across teams working on different aspects of the project and vertically across different levels of management. Even nonparticipants are encouraged to read community forums to learn about the project's development (Canonical Ltd, 2010). Mailing lists, on the other hand, constitute discussions not entirely aggregated in any specific place, and thus not as easily perused; unlike forums, however, they are sent directly to all users on the list, and thus the person sending the information can be more confident that it was received. According to Thomas, the mailing list is the prevailing forum for official statements from governing boards, though mailing list messages often spawn very specific threads about particular aspects of the project (Thomas, 2010). The Wiki differs from both the forums and the mailing lists in that information is not tied to the person disseminating it, a fact which allows the Wiki to function as a more formal and official mode of communication, since readers can assume whatever information is there has been verified by a majority of the project's community (Thomas, 2010).

The Ubuntu project is tied together by a complex, interacting network of hierarchical structures and lateral communication networks allowing a large number of developers to work efficiently and effectively on a large project. But one particular kind of unity, without clear virtual form, is the ideology shared not just among the project's participants but also with the wider community of community-managed software projects. As Moody puts it, "open source projects are not about software code only....they are also about freedom, sharing, and community; they are about creation, beauty, and...the code within that is at the root of all that is best in us" (Moody, 2001). This sense of trust works in tandem with the structure the Ubuntu community has enacted to foster a collaborative, noncompetitive environment that every developer I spoke to described as not just successful as a means of production, but fulfilling as a means of bringing the world more in line with one's values.

Appendix

The research supporting my conclusions in this paper cam from many different sources—books, official websites, news sites, articles, and individuals participating in open-source development. While I initially expected the paper to be entirely based on individual experience, I quickly realized that I could not find out specific details about governing bodies from contributors who spent most of their time interfacing with other contributors. In deciding to focus on both the vertical and the lateral organization of the organization, I ended up using more books, articles and interviews by reporters in my paper.

The developers I interviewed were mostly hobbyists: Zachary Edwards, Jonathan Thomas, and Chris Brislaine each put in about 5-10 hours a week of work on the project, and Rob Sobczak about 10, although David Thacker guessed he put in more than that as a result of his leadership position. All interviews were done by email: I sent them each a list of questions I had, they responded, and then I asked follow-up questions as needed. (All interview dates in the Works Cited are the dates of the first email sent out.) Most of them were very responsive and willing to describe their experiences working on the Ubuntu project.

Works Cited

Brislaine, Chris. E-mail interview. 14 April 2010.

"Browser Statistics." W3Schools. 2010. 30 March 2010. <http://www.w3schools.com/browsers/browsers_stats.asp>.

Canonical, Ltd. "About Us | Ubuntu." Ubuntu. 2010. 30 March 2010. <http://www.ubuntu.com/aboutus>.

Edwards, Zachary. E-mail interview. 30 March 2010

Kogut, Bruce and Anca Metiu. 2001. "Open-Source Software Development and Distributed Innovation." *Oxford Review of Economic Policy* 17:248-64.

Lester, Richard K. and Michael J. Piore. Innovation: The Hidden Dimension. Cambridge, MA: Harvard University Press, 2004.

Moody, Glyn. "Linux is a platform for people, not just specialists." The Guardian Online. 22 May 2008. 21 April 2010. <http://www.guardian.co.uk/technology/2008/may/22/internet.software>.

Moody, Glyn. Rebel Code: Inside Linux and the Open Source Revolution. Cambridge, MA: Perseus Publishing, 2001.

O'Mahoney, Siobhán. 2003. "Guarding the commons: how community-managed software projects protect their work." *Research Policy* 32:1179-1198.

"Open Source Definition, The." Open Source Initiative. 21 April 2010. <http://www.opensource.org/docs/osd>.

Pavlicek, Russell C. Embracing Insanity: Open Source Software Development. Indianapolis, IN: Sams Publishing, 2000.

Polodny, Joel M. and Karen L. Page. 1998. "Network Forms of Organization." *Annual Review of Sociology* 24:57-76.

Raymond, Eric S. The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol, CA: O'Reilly, 2001.

Rosenberg, Donald K. Open Source: The Unauthorized White Papers. New York, NY: IDG Books Worldwide, Inc., 2000.

Sobczak, Rob. E-mail interview. 14 April 2010.

Thacker, David. E-mail interview. 20 April 2010.

Thomas, Jonathan. E-mail interview. 14 April 2010.

Vance, Ashlee. "Ubuntu and Its Leader Set Sights on the Mainstream." NYTimes.com. 10 Jan 2009. 20 April 2010. <http://www.nytimes.com/2009/01/11/business/11ubuntu.html>.

VAR Guy, The. "Ubuntu: Canonical's New CEO Discusses Top Priorities." The Var Guy. 4 March 2010. 30 April 2010. <http://www.thevarguy.com/2010/03/04/ubuntu-canonicals-new-ceo-discloses-top-priorities>.