22 High Noon at OS Corral: Duels and Shoot-Outs in Open Source Discourse

Anna Maria Szczepanska, Magnus Bergquist, and Jan Ljungberg

The open source software (OSS) movement can be related to the societal changes that started in the late 1960s: the rise of a network society supported by new information and communication technologies and the rise of new forms of collective actions, which also have been referred to as "new social movements" (see Touraine 1981; Melucci 1996; Thörn 1997). The emergence of these movements has been explained by changes in the relations between the economic, political, and cultural powers and institutions that have supported a transition from a modern to a late modern or postindustrial society and lately been linked to an understanding of the importance of the rise of a global network society and new forms of communication (Castells 1996; Giddens 1990). The emergence of a network society, or information society, has involved new conflicts regarding the control over information, knowledge, symbolic capital, and social relationships. These are conflicts closely connected to unequal or transformed power relations between different social positions in the new global society. According to both Thörn (1997) and Melucci (1996), it is in this social climate of opposition, ambivalence, and conflict that new forms of collective actions have emerged.

American contemporary movements of the 1960s gave rise, as Castells (1996) has shown, to an intellectual climate that contributed to and inspired the march of technical innovations and the "culture of IT". However, the rise of information technology also gave birth to conflicting ideas about how the tools to create and bring forth information should be developed, and what they should look like. When large hardware and software companies—such as IBM and later, Microsoft—slowly gained dominance over the growing IT market, this had serious consequences for how the power relations between different actors in the field of software development were organized. A resistance to this development has been growing around different advocates of open source and free software, raising

questions on the freedom of speech and information. Advocates of open source and free software have also in recent years been noticeable in the discussions on how to bridge the "digital divide," thus trying to offer an alternative IT infrastructure that avoids expensive licensing structures. In order to achieve a position in the world of systems development as a counter-movement to proprietary software actors, the members of the open source movement have to be able to create a shared identity. As Thörn (1997) points out, collective identity is one of the most important traits of any social movement. Collective identity is created by, or related to, a movement culture comprised of a relatively autonomous network of interactions between different individuals, institutions, and organizations. In order to be effective, movements have to be goal-oriented, and act as strategic collectives that always strive toward social change. Collective identity is a powerful force that has to be fully recognized when understanding how social movements are assembled and constituted. Collective identity provides an important context for the creation of meaning, social integration, and action. The symbolic dimension of collective action is to manifest, and thereby constitute, the unity of the group. This is done through a multidimensional process of communication—that is, rituals and demonstrations—or with the help of texts. Collective identity thereby incorporates different forms of narrative that create an overall meaning for the individual, for his or her everyday practices, and for the symbolic manifestations that are communicated by members. Accordingly, the production and use of texts give new social movements a discursive form (Thörn 1997).

Viewing Open Source from a Discourse Perspective

The concept of discourse builds on a social constructionist perspective where language is seen as constitutive of social reality; this means that an important access to reality is through language and its manifestation in discourses. A discourse can be understood as a group of statements that produce and define objects of knowledge, but also inform us in how to conduct our social and cultural practices (Foucault 1972). Discourse is tied to practice, and is articulated not only through text, but also through metaphors, symbols, cultural codes, stories, pictures, and other forms of representation.

Discourse constructs the subject and present reality in a certain way, thus creating limits between true and false, relevance and irrelevance, right and wrong. It also limits other ways in which a topic can be defined (Hall 1992).

Discourses thereby create webs of meaning that cluster around certain topics. However, a discourse is not a closed entity, but is continuously reconstructed in contact and struggle with other discourses. Different discourses that represent certain ways of speaking about the world then struggle for domination. This struggle concerns taking command in defining the world from a certain point of view; that is, to become the normative discourse defining social order and making sense of the world according to that view. Discourses possess different powers, which in some sense make us more secure, feeling safer in a world that is somewhat predictable, but they also operate in disciplinary and authoritarian ways.

Analyzing discourses in the open source movement is not about reducing the movement to texts. It is, rather, a way to understand how collective identity is created, communicated, and managed in the form of "webs of meanings." Understanding discursive practices becomes especially important because of the movement's decentralized and networked character. In this chapter, we will analyze different discourses taking place both within and outside the open source movement. First we present discourses that are related to how a sense of "us" is created within the movement. Then we discuss how authority and leadership is created, or how discourses are "managed." We then elaborate on how the enemy is constructed, and how the enemy's discourse fights back. Finally we address internal dynamics between different actors in the open source community.

Constructing the Hacker

Developing discourses is vital for providing the members of the movement with a meaningful context that enables creative software development activities across organizational and geographical boundaries. People feel a bond with others not because they share the same interest, but because they need that bond in order to make sense of what they are doing. Discourses in the form of texts and symbols enable members of a community to affirm themselves as subjects of their action and parts of a collective action. Actors in the open source movement handle a specific set of discursive practices. They gain status as they learn to master the rhetoric of the community.

This process of socialization often starts with being a "newbie" and ends with achieving the status of being a "hacker" (Bergquist and Ljungberg 2001). In order to be able to socialize new members into the community, symbolic representations of the core values in the community must be created and communicated. This is done in several ways: through virtual

objects, symbols, strings of statements and messages, and with the help of certain ways of associating with symbolic tokens that sometimes, from an objective point of view, are only vaguely related to the core business of open source. In this section, some examples will be given on how the relationship between discourse, practice, and identity is constructed and communicated in the open source movement.

One way to understand the collectiveness within the open source community is through the concept "hacker." Even if there seems to be no unambiguous definition of the concept, certain characteristics like creativity and a genuine interest for troubleshooting most commonly describes the work of a hacker (see for example, http://wikipedia.org). The online document "The New Hackers Dictionary" including the Jargon File (Jargon File 4.3.1), which is well known within the hacker community, gives a comprehensive insight in the tradition, folklore, and humor of the hacker community. The Jargon File includes the Jargon Lexicon a compilation of the specific vocabulary used by hackers. The Jargon File is one instantiation of a discourse that is continuously replicated in different forms in the community. Implicitly the documentation contains general social and cultural aspects of what it means to be a hacker, what they do, what their preferences are, and, in that sense, what defines them as a collective.

The Jargon File is a continuing work-in-progress. The earliest version of the Jargon File was created in university milieus, especially the MIT AI Lab and the Stanford AI Lab and other communities that grew out of the cultures around ARPANET, LISP, and PDP-10 in the 1970s. It was also in these technical milieus that the concept of *hacker* came into use. Eric Raymond, one of the leading figures within the community, has done the main work, with the latest version updated and presented with the characteristic version numbers: 4.3.1, 29 June 2001. It has been revised so that it complements the cultures that have emerged due to new programming languages, hardware and software applications: most prominently the C language and Unix communities, but also, for example, IBM PC programmers and Amiga fans. The fact that Raymond has updated the original Jargon File to include different hacker cultures indicates the importance of defining the hacker community so that it represents different understandings of what it means to be a hacker.

Eric Raymond divided the content in the Jargon Lexicon into three categories: (1) slang, or informal language from mainstream English or nontechnical subcultures; (2) jargon, "slangy" language peculiar to or predominantly found among hackers, and finally, (3) techspeak, the formal

technical vocabulary of programming, computer science, electronics, and other fields connected to hacking. The understanding of hacker culture thus must be seen as a multidimensional process of communication between different cultural manifestations deriving from both a social (real-life) and a technical (virtual) level. The hacker culture takes inspiration from and merges different cultural expressions. It also expresses different distinct fields that stand in opposition to each other. In this process, we can see several discourses take form. These involve hackers as a distinct field that excludes other fields as nonhackers, but also "internal" fields constituted by different hacker cultures as defined by their interests in and use of different technologies.

Looking up the word *hacker* in the Jargon File, the affinity toward the "intellectual challenge of creatively overcoming or circumventing limitations" is considered the primary characteristic, but a hacker is also defined as connoting "membership in the global community defined by the Net" and as "a person capable of appreciating hack value" and one possessing "hacker ethics." Since the hacker community is described as a meritocracy based on ability, it is also stated that hackers consider themselves something of an elite (Jargon File 4.3.1). The problem-solving aspects of participation in the open source movement—to be able to fix a bug right away and participate in rapid software development through knowledge sharing—are characteristics commonly associated with the strengths of the movement, and the skills of the programmers.

However, a common characteristic of movements is internal conflicting relations between different groups of actors. In the open source movement, some conflicts are due to different hacker cultures and traditions. In an enclosure to the *New Hackers Dictionary*, Eric Raymond has incorporated a section called "Updating JARGONG.TXT Is Not Bogus: An Apologia" (Jargon File 4.3.1). In this text, Raymond responds to criticism from hackers representing the PDP-10 culture. The criticism pointed out that the Jargon File was Unix-centric, that Raymond lacked insider knowledge of their culture, and that blending Unix and PDP-10 cultures distorted the original jargon file. This illustrates an example of the importance of experiences and meaning connected to a certain field of interest and practices with its own historical process. It also points to the significance of being accepted as a separate collective with its own cultural identity. In Raymond's answers to the criticism, he verified the fact that the Unix and the PDP-10 community did have their own identities but that they also belong together as being hackers. The "us" is thereby expanded to include various groups and networks creating an even greater

potential developer and user base. The tension between different (groups of) actors within the open source movement, and between the open source movement and the free software movement, is an interesting topic leading to a deeper understanding of how collective identity is created and managed in the open source movement. This topic is elaborated upon later in this chapter.

Managing the Production of Discourses—The Leaders

Eric Raymond has become a leading figure within the open source movement, and he is the president of the movement's formal body, the Open Source Initiative. He attained this position as a recognized contributor to the Linux project, and as the author of a much-cited anthropological analysis of the open source movement (Raymond 2001). He has also drawn considerable attention to the open source community with, for example, the publication of the "Halloween documents," a set of confidential Microsoft memos that discussed the potential threat from open source software to Microsoft products. These memos were important for the open source movement, in the sense that their competitors acknowledged the potential of the movement.

Through these different types of contributions and through widespread reputation, Raymond has become a key person in the open source movement. Such leaders or "movement intellectuals" are interesting, because they possess great power through the advantage of interpreting the orientation, goal, and work of the movement. They have the power to formulate and set standards of what should be done, how it should be done, and why it should be done. They also have the power to define and manipulate texts and symbols and arrange them within the discursive context. As in the example of the Jargon File, we can see how Raymond tries to formulate a "true hacker nature" as a strong symbol for a collective identity throughout his vivid descriptions of how hackers are, and how they think and code.

In the sense that movement intellectuals play a central role for identifying the collective, they enjoy the ability to include and exclude (Thörn 1997). They create cultural forms that set the agenda for the movement, but also create new forms of cultural and social understanding of the community. This can be exemplified by the rhetoric of different movement intellectuals, as with the polemic between Eric Raymond and Richard Stallman, founder and leader of the free software movement from which open source grew.

Stallman, a hacker formerly at MIT, has positioned himself as one of the most prominent activist within the programming community, with large contributions to free software, his GNU/Linux project, for example, and as the founder of the Free Software Foundation (FSF). Stallman always adopted a more ideological line in his work with free software, promoting the freedom of hacking and information, than that found in the open source movement. Taking a look at his personal website, his political interest in different "freedom of speech"—and in the civil rights movements—are evident. However, Raymond has a more pragmatic outlook, which maybe is best explained via a well-known dispute between Raymond and Stallman, initiated by a posting from Stallman to an Internet bulletin board:

> People have been speaking of me in the context of the Open Source movement. That's misleading, because I am not a member of it. I belong to the Free Software movement. In this movement we talk about freedom, about principle, about the rights that computer users are entitled to. The Open Source movement avoids talking about those issues, and that is why I am not joining it. The two movements can work together on software. . . . But we disagree on the basic issues. (Stallman 1999b)

Raymond responded with an essay where he stated that he could agree on Stallman's ideas about freedom and rights, but that it was ineffective and bad tactics for the free software community to engage in those issues:

> OSI's tactics work [. . .] FSF's tactics don't work, and never did. [. . .] RMS's [Richard Matthew Stallman] best propaganda has always been his hacking. So it is for all of us; to the rest of the world outside our little tribe, the excellence of our software is a far more persuasive argument for openness and freedom than any amount of high-falutin appeal to abstract principles. So the next time RMS, or anybody else, urges you to "talk about freedom," I urge you to reply "Shut up and show them the code." (Raymond 1999b)

Though the open source and free software movements share an ambition to create free software of high quality, and share mutual cultural expressions through the hacker community, this is an example of a power struggle where the leaders take different roles as symbolic organizers. It is a battle that originates from different perspectives on how work should be done and what the strategies are to be. Most importantly, this inspires members to act on different forces and purposes when contributing to the movement.

Last but not least, Linus Torvalds should be mentioned, as he is considered an icon of the open source movement. Because of his extraordinary

contribution in leading the development of the Linux kernel and his further work making decisions about contributions to the Linux kernel project, he is considered one of the father figures of the open source movement. Torvalds seldom comments on political issues concerning the work of the movement. Torvalds keeps a low profile, which can be seen as a symbolic incarnation of the hacker ethic. At the same time he is seen as the charismatic leader of the movement. The charismatic leader is recognized as being such by virtue of the extraordinary qualities that ensure him/her a mass following (Melucci 1996). In this case, Torvalds has become a symbol for the movement as such; he stands for the kind of values and practices that are associated with the hacker. It might seem contradictory to define a person who keeps low profile as charismatic, but then it is considered good manners amongst hackers not to brag about the exploits made. A movement leader receives his/her position as a leader of the community if he/she is publicly acknowledged by the collective. In the hacker context, this is therefore mainly a matter of exceptional contributions and reputation. This is clearly expressed in the Jargon File by the definition of *demigod*:

> **demigod** *n.* A hacker with years of experience, a world-wide reputation, and a major role in the development of at least one design, tool, or game used by or known to more than half of the hacker community. To qualify as a genuine demigod, the person must recognizably identify with the hacker community and have helped shape it. Major demigods include Ken Thompson and Dennis Ritchie (co-inventors of Unix and C), Richard M. Stallman (inventor of emacs), Larry Wall (inventor of Perl), Linus Torvalds (inventor of Linux), and most recently James Gosling (inventor of Java, NeWS, and GOSMACS) and Guido van Rossum (inventor of Python). In their hearts of hearts, most hackers dream of someday becoming demigods themselves, and more than one major software project has been driven to completion by the author's veiled hopes of apotheosis. See also **net.god**, **true-hacker**. (Jargon File 4.3.1)

Us and Them—Constructing the Enemy

In order to be able to construct an "us" that can be successfully communicated within the movement, a "them" also has to be constructed in order to sharpen the edges and help the movement build their collective identity. A "them" is a part of all movements. It has the function of strengthening the movement as well as legitimating its norms, values, and actions. The following examples taken from the Jargon Dictionary show how hackers define themselves when articulating what they are not. In the

humorous example of *suit*, a certain "hackish" lifestyle is expressed by an ironic description of lifestyle tokens that belong to the other:

suit 1. Ugly and uncomfortable "business clothing" often worn by nonhackers. Invariably worn with a "tie," a strangulation device that partially cuts off the blood supply to the brain. It is thought that this explains much about the behavior of suit-wearers. (Jargon File 4.3.1)

The open source movement seems to have created a notion of open source software being superior to proprietary software. A movement needs an "enemy" to strengthen the community from the inside. In the open source movement the enemy part is played by the proprietary software industry, most often represented by Microsoft, which has played an important role as the "evil empire," as in this example taken from the Jargon File:

Evil Empire [from Ronald Reagan's famous characterization of the communist Soviet Union] Formerly IBM, now Microsoft. Functionally, the company most hackers love to hate at any given time. Hackers like to see themselves as romantic rebels against the Evil Empire, and frequently adopt this role to the point of ascribing rather more power and malice to the Empire than it actually has. (Jargon File 4.3.1)

The world of proprietary software is not only the evil enemy; it is also portrayed as a world of less intelligence. In an example of "adbusting" (figure 22.1), found at the "Micro$oft HatePage," a certain "collective excellence" constituting the open source movement is implicitly stated in contrast to what are considered the weaknesses of the "enemies."

The argument is that even though the enemy has resources in the form of people and money, the software developed is of low quality. The open source movement has a superior organization, smarter developers, and a culture that enables the movement to create software of higher quality in less time. Open source software is based on real needs, addressing real problems and developed in a fashion that secures the best quality compared to proprietary software development, which is driven only by the desire to make money and protect its intellectual property rights. The "Evil Empire" and "Borgs" symbolize this oppression by monopolistic software companies that try to limit the freedom of using and modifying software.

Borg [. . .] In *Star Trek: The Next Generation*, the Borg is a species of cyborg that ruthlessly seeks to incorporate all sentient life into itself; their slogan is "You will be assimilated. Resistance is futile." In hacker parlance, the Borg is usually Microsoft, which is thought to be trying just as ruthlessly to assimilate all computers and the

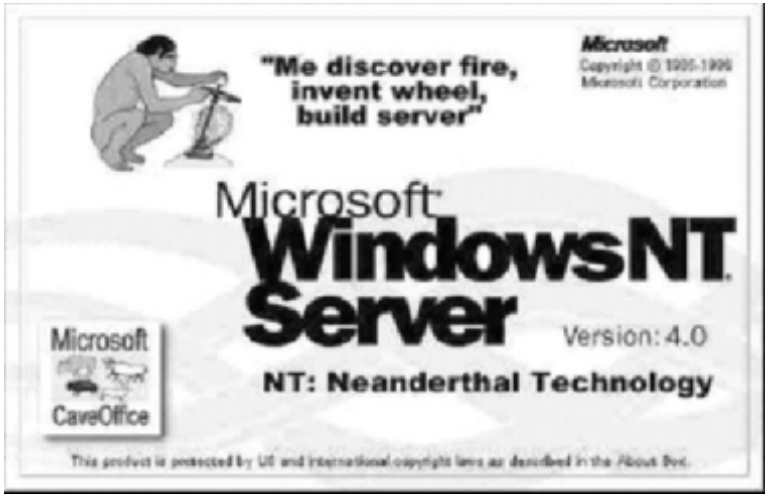

Figure 22.1
An example of "adbusting"

entire Internet to itself (there is a widely circulated image of Bill Gates as a Borg). Being forced to use Windows or NT is often referred to as being "Borged."

An important part of constructing the enemy is the enemies' own construction of open source. First Microsoft appeared to ignore open source as irrelevant and not a real threat. When the "Halloween documents" were leaked to Eric Raymond, it became obvious that Microsoft saw it as a threat and took it seriously. By annotating the memorandum with explanations and ironic comments and releasing it to the national press, Raymond made the memorandum part of the open source movement's strategy of constructing the proprietary software industry as an enemy. From being an internal Microsoft affair, the Halloween documents developed into an important story in open source folklore, embodying both the evilness and clumsiness of the enemy. The Halloween documents confirmed Microsoft's status as enemy; Raymond even felt obliged to thank the original authors:

> This page originally continued with an anti-Microsoft jeremiad. On reflection, however, I think I'd prefer to finish by thanking the principal authors, Vinod Valloppillil and Josh Cohen, for authoring such remarkable and effective testimonials to the excellence of Linux and open-source software in general. I suspect that historians may someday regard the Halloween memoranda as your finest hour, and the Internet community certainly owes you a vote of thanks. (Raymond 2003a)

When the "enemies" get back at open source in the media, they attack aspects of the software like quality and security, but also (mostly) its core values, such as the free distribution of software code, particularly through the GPL. The arguments draw on metaphors from cancer to communism, and the GPL has even been likened to Pac-Man, the little monster eating all in its way. It all serves to deconstruct the distribution model, arguing that it will stifle innovation and eat the healthy part of the software industry: According to Microsoft executive Jim Allchin, "Open-source is an intellectual property destroyer. I can't imagine something that could be worse than this for the software business and the intellectual property business (cited on CNET News.com Feb. 14, 2001). And Steve Ballmer, CEO of Microsoft, added, "Linux is a cancer that attaches itself in an intellectual property sense to everything it touches", (cited in *The Register*, June 2, 2001).

From the open source movement's point of view, these attacks are characterized as FUD (Fear, Uncertainty, and Doubt), spread with the purpose of maintaining Microsoft's monopoly. FUD is defined as a marketing technique used when a competitor launches a product that is both better and costs less than yours; that is, when your product is no longer competitive. Unable to respond with hard facts, scaremongering is used via "gossip channels" to cast a shadow of doubt over the competitors' offerings and make people think twice before using it. FUD has been alleged to have been first used on a large scale by IBM in 1970s, and the technique is now argued to have been applied by Microsoft.

Another part of this discourse attacks the free distribution model by accusing it of being unAmerican: "I am an American; I believe in the American Way, I worry if the government encourages open source, and I don't think we've done enough education of policymakers to understand the threat" (Jim Allchin, of Microsoft, cited on CNET News.com, Feb. 14, 2001).

Microsoft's rhetoric of the American way and of open source being unAmerican is used by open source supporters, to associate Microsoft with the McCarthy era and another way of being unAmerican. "If this continues I may one day be sitting in front of the House of Un-American Activities Committee with a modern McCarthy asking: Are you or have you ever been an advocate of open source software" (Reed 2001).

Both proponents and opponents of free software and open source software use the discourse about the "true American Way" to support their claims. Discourses attached to open source thereby become related to webs of meaning that are of profound significance to the people who make the

claims—in this case, Americans—regardless of whether they are for or against the movements. Microsoft refers to the software industry as the American Way, because it is an expression of the free market and civil rights. Free software and open source advocacy groups use the idea of the American Way to argue that the "freedom" in free software and the "openness" in open source connote American core values from the pioneering era and the constitution. Symbolic objects can be found that support the idea that the movements are an incarnation of the American Way; for example, when Richard Stallman puts the American flag on his home page together with the text "America Means Civil Liberties, Patriotism Is Protecting Them."

By citing Gandhi's famous words, this author describes the state of the process in the open source versus Microsoft combat: "'First they ignore you, then they laugh at you, then they fight you, then you win' (Gandhi). This is the exact path Microsoft has taken with open source software. When I first got involved with Linux eight years ago, we were in the ignore stage. We are definitely in the fighting stage now" (Reed 2001).

Dynamics within the Open Source Community

We have pointed out the persistent voices of proponents claiming the superiority of the social production of technological achievements presented by the open source movement. Openness is regarded as the key to technological excellence, profit making, different freedoms, and might even be the way to world domination or a new sociotechnical order. This discourse of supremacy, pride and self-assurance must be understood in relation to the meritocracy of hacker culture proposing that ideas, innovation, and problem solving form the fundamental basis of openness and freedom. The hacker culture, with its roots in academia and science, expresses a "belief in the inherent good of scientific and technological development as a key component in the progress of humankind" (Castells 2001, 39). Linked to this discourse is also the rhetoric about the different models used for the "success" of open source: the bazaar approach, the principles of gift culture and the peer-review process. Even if these aspects refer to what might seem uncontested and coherent communitarian subcultural hacker values, as well as the work for common good, these models include more or less institutionalized discursive practices that involve specific rules, authorities, competencies (in other words, powers). These powers manage and coordinate and therefore control the social organization of open source communities and work, but also define the relevance and rank the individual

performances of the contributors. As highlighted by an open source developer, "If you look closely, there really isn't a bazaar. At the top, it's always a one-person cathedral. It's either Linus, Stallman, or someone else. That is, the myth of a bazaar as a wide-open, free-for-all of competition isn't exactly true. Sure, everyone can download the source code, diddle with it, and make suggestions, but at the end of the day it matters what Torvalds, Stallman, or someone else says" (Wayner 2000, 115).

Due to the transnational feature and Internet-based networking aspects of the extensive open source movement, the best strategy for a person willing to contribute to the movement is either to join a project group or start a project on his/her own. The keystones of the development process within the project group are the open communication, cooperation, and sharing of resources (ideas, knowledge, and information). The peer review system is a way to ensure the quality of the development processes within different projects. Yet, as suggested in the previous quote, the openness of the development process is not always in the end the common cause of the community, but rather determined by the hacker elite. Raymond (2001) has characterized the sharing of personal assets as a gift culture. There is often no monetary compensation to be expected for efforts conducted, which is in line with the informal hacker ethic not to use common resources for personal benefit. Contributions to the movement therefore have to be explained in terms other than being based on the traditional cost-benefit rationality (Bergquist and Ljungberg 2001). Rather, the benefits are those of being part of and learning from the community, and the inner satisfaction of being able to contribute and being acknowledged for the efforts made, as well as the impact the community has on the surrounding world. Reputation and status due to intellectual and technological skills as a driving force is of course comparable with the aspirations within the academic tradition, as noted by Castells, for example, who also reminds us that the Internet was born out of academic circles where the academic values and habits—such as academic excellence, peer review, openness in research findings, credit to the authors of discovery, as well as academic ranks—"diffused into the hacker culture" (Castells 2001, 40).

It is evident that open source software development can be understood as some kind of gift culture. However, exchanging gifts does not automatically create a society without borders, driven only by individuals' inner satisfaction, where everyone steps back for the sake of the common good. An important dimension is also how power structures the exchange of gifts. This element in gift giving has been analyzed by the anthropologist Marcel Mauss in *The Gift* (1950/1990). Mauss understands gift giving

as the transaction of objects coordinated by a system of rules. The rules are in fact symbolic translations of the social structure in a society or a group of community members. He argues that giving a gift brings forth a demand for returning a gift, either another object or, in a more symbolic fashion, forces of power connected to the objects. Gift giving therefore creates social interdependencies and becomes a web upon which the social structure is organized. To give away something is to express an advantageous position in relation to the recipient.

Gifts therefore express, but also create, power relations between people. A gift culture, in this sense, has the power to discipline loosely coupled networks of individuals with no organizational forces in terms of economy or management that can otherwise force individuals to behave in a certain way. The gift culture in the open source setting is thus often, in contrast to traditional gift giving between two persons, rather an exchange between a contributor and the whole community of developers and users. Therefore, the obligation of returning a "favor" lies more, as already suggested, in the symbolic powers of acknowledgment, status and gratitude (Bergquist and Ljungberg 2001). Bergquist and Ljungberg (2001) have pointed out several aspects of how power and social stratification are expressed within the open source gift culture. For example knowing how and when to show respect and appreciation is a commonsense "netiquette" within the OS community. The norms and values though are not obvious for a first time visitor or a "newbie" who has to be socialized into the outspoken as well as the nonverbal practices of the community. Even if a common trait of the open source discursive rhetoric is the warm welcome of an expansion and growing strength of the movement, a more unspoken discursive practice built upon a memory from the tribal days of the techno-elite and a strong meritocracy where talent and advanced skills are all that matter, are quite evident. The frequency of FAQs for beginners to learn the most basic issues—from the "flamewars" against bad contribution or behavior and, as Bergquist and Ljungberg have pointed out, the humiliation of oneself before a more advanced community as a survival strategy of a newbie—are some aspects that point to elitism and a strong meritocracy. Norms and values are not always accepted without protest, though, and countermovements are being developed. In the following example from a newsgroup posting, the practice of flaming is questioned:

> More than once I have had the urge to begin contributing to the community. I have written code, documented it, and gained authorization for its release. But at the last minute I always hesitate and then stop. Why? I think I fear the fangs of the com-

munity. At this point, everywhere I turn, it's a big flamefest and getting quite tiresome. Its gotten to the point where it seems one has to be some sort of Jedi Master–level coder to contribute. On more then a few mailing lists, I have seen contributors flamed for their contributions! Flaming someone for *giving* something away. It's incredible. (In Bergquist and Ljungberg 2001, 315)

As the posting reveals, the practice of flaming is due not only to inappropriate behavior and the overruling of core communitarian values, but might also be used by the inner circle of a project team as a means to reject contributions. The system of peer review might in this way be a powerful tool to manifest authority and maintain the hierarchy of "excellence." The open source peer review model has no objective criteria for what counts as relevant or impressive contributions. On the other hand, project leaders try to be reasonable when making their selections. But what seems reasonable for one person is not always reasonable for another. Peer review is thus a social mechanism, through which a discipline's experts or the core members of a community maintain control over new knowledge entering the field (Merton and Zuckerman 1973; Chubin and Hackett 1990). Peer review can thus be seen as a way of organizing power relationships within a given community, but as we have also seen, in relation to other fields of social practices.

Conclusion

A discourse perspective gives important access to an understanding of how networked movements like open source are organized and managed. It also brings insight to the power struggles going on between different actors. One interesting point is that the values created within the movement's discourses are now spreading to larger circles outside the movement, expanding its territory. Thus, it can be concluded that the force of open source discourses is very powerful.

First, we illustrated the internal discourses forming open source identity. Then we described the struggle between open source and proprietary software (particularly Microsoft) discourses. The discourses about "the American Way" is an example of how both the open source movement and the proprietary software industry try to legitimate activities by relating to an external discourse about "good" American society.

In conclusion, we note that external actors in public sector and governments, representing a particular user category, have also linked to several open source discourses in order to change their policies and practices regarding information systems. These new discourses have the potential

enormous for an impact. Governments exist in every country and represent large user groups. The public sector discourse relates to several of the themes originating in open source discourses, some of them previously discussed in this chapter. Concepts like "freedom" and "openness" easily relate to democratic values, and in some countries (for example, Sweden) it is a constitutional right to have access to large parts of governmental information. Thus some would ask: Why should the source code of public sector information systems therefore be excluded from this openness? Because governments invest in information technology using taxpayers' money, there is a related demand to increase the value for money." For these reasons, it seems likely that the next major shoot-out will take place in the governmental and public sector arena.