Ethics Inf Technol (2010) 12:327–341
DOI 10.1007/s10676-010-9230-x

ORIGINAL PAPER

How can contributors to open-source communities be trusted? On the assumption, inference, and substitution of trust

Paul B. de Laat

Published online: 11 June 2010

Abstract Open-source communities that focus on content rely squarely on the contributions of invisible strangers in cyberspace. How do such communities handle the problem of trusting that strangers have good intentions and adequate competence? This question is explored in relation to communities in which such trust is a vital issue: peer production of software (FreeBSD and Mozilla in particular) and encyclopaedia entries (Wikipedia in particular). In the context of open-source software, it is argued that trust was inferred from an underlying 'hacker ethic', which already existed. The Wikipedian project, by contrast, had to create an appropriate ethic along the way. In the interim, the assumption simply had to be that potential contributors were trustworthy; they were granted 'substantial trust'. Subsequently, projects from both communities introduced rules and regulations which partly substituted for the need to perceive contributors as trustworthy. They faced a design choice in the continuum between a high-discretion design (granting a large amount of trust to contributors) and a low-discretion design (leaving only a small amount of trust to contributors). It is found that open-source designs for software and encyclopaedias are likely to converge in the future towards a mid-level of discretion. In such a design the anonymous user is no longer invested with unquestioning trust.

Keywords Design · Discretion · FreeBSD · Hacker ethic · Mozilla · Open source software · Trust · Wikipedia

P. B. de Laat (✉)
Faculty of Philosophy, University of Groningen, Oude Boteringestraat 52, 9712 GL Groningen, The Netherlands
e-mail: P.B.de.Laat@rug.nl

Introduction

From the 1980s on, the concept of trust has attracted renewed interest in disciplines like philosophy, the social sciences and economics. Especially important contributions were made by Luhmann (1968), Baier (1986, 1992), Gambetta (1988) and Pettit (1995). Conceptually, trust was understood as the phenomenon of reliance on the good intentions and/or adequate competences of others in situations characterized by dependence, vulnerability and risk. The particular example of a trusting situation, introduced by Dale Zand decades ago, is still a valid exemplar: entrusting one's child to a babysitter.

The usual mechanism by which one decides to trust others on whom one may perhaps rely is *inference*: their reputation based on past performance, certain perceived individual characteristics like sex or ethnicity, or institutions to which they can be linked inspire confidence in the transaction under consideration (Zucker 1986). Trustworthiness is inferred. Likewise, a shared culture and the specific context of the transaction—with actors involved calculating the costs and benefits of their actions—may be supposed to justify the inference of trust. More rarely, people may simply *assume* trustworthiness absent any clues to that effect, and just decide to act *as if* trust were present; one may actually produce trust, precisely by acting as if it is present (Gambetta 1988). The mechanisms underlying such an assumption of trust, in virtual life in particular, have been elucidated by Pettit (2004) and de Laat (2005).

If people continuously interact in 'trust situations', rules and regulations are likely to be introduced. Hierarchy, procedures, and contracts are relevant instances. This is when a third mechanism for handling the trust problem emerges (whether applied wittingly to that end or not).

Structure regulates mutual interactions by introducing restrictions on behaviour; degrees of freedom are circumscribed or even eliminated. As a result, participants in the system are, to varying degrees, no longer relied on; their possible actions are restricted. The amount of trust that needs to be invested in people is reduced to the extent that rules restrain them. In other words, structural arrangements may partly *substitute* for the need to perceive—or assume—people as trustworthy (cf. Granovetter 1985; Sitkin and Roth 1993). Note that this argument must not be taken to imply that rules can mechanically 'solve' the trust problem; some degree of actual trust will always be needed.

In all a three-fold dynamics of trust emerges: of assumption, inference, and substitution of trust. After a brief survey of open-source communities in general, arguments are presented, explaining why trust is a central issue in the functioning of two particular examples: software and encyclopaedias. The remainder of the paper is then devoted to examining how this dynamics of trust unfolds within the two communities.

Open-source communities and trust

In analogy with open-source software (OSS) 'open-source communities' in general can be defined as peers producing content together on a voluntary basis, without direction from markets or managerial hierarchy, and posting their created content in a virtual commons accessible to all. This, of course, is the definition coined by Benkler (2006). Social movements develop content collectively, both 'by the people' and 'for the people'. A new mode of production is born. It all started with OSS, which attracted increasing numbers of participants with the advent of the Internet. Subsequently the movement spread from software to other kinds of content: encyclopaedias, journals, books, movies and more came to be produced in an open-source fashion. Note, though, that the epithet 'open-source' in these instances just refers to the circumstance that contributed content is readily made available for distribution, refinement and modification—the typical software distinction between source code and object code no longer applies.

A mode of production like this opens the gates to the outside world for everybody. Anybody is invited to contribute inputs that are relevant to the project. But to what extent can these suggestions be relied onto make a valuable contribution, and be taken into account or ultimately integrated into the official, up-to-date version of the project concerned? Put the other way round: to what extent can one be sure that the incoming uploads are not disruptive or undermine the collective cause? Some communities are characterized by little interaction between contributors: the project simply amasses all inputs together, like photographs, journal entries, or music samples. In such cases, dubious quality inputs do little harm. However, when contributors do interact continuously about interconnected content that is ever-evolving, the quality problem becomes more acute. Exemplary domains of content with such dense interaction are software and encyclopaedias. With software, one is invited to submit comments, bug reports, code patches or new features in source code; with encyclopaedias, one is invited to submit comments or suggest changes to existing entries, or suggest new entries altogether. In both cases, the contents are in perpetual flux.

As soon as a project grows in size—and monitoring by a single person becomes unfeasible—those in charge have to ask themselves who can be trusted to provide valuable comments and/or content, in a spirit of loyal cooperation and proportional to their competences. Those—and only those—worthy of one's trust then can be given permission to introduce their changes directly into the official version as presented to the public. The defining body of content (either the source code repository or the body of textual entries as a whole) is *entrusted*, as it were, to a collective of dedicated contributors to take care of. In contrast to one's child in the babysitter example, though, the goods being entrusted are ever-expanding.

That trust is an issue can be shown quite specifically. When OSS hackers create a tree of source code together, inappropriate code may be a nuisance (cf. de Laat 2007: p. 171). For one thing, code may be sloppy, of mediocre quality, or contain bugs. For another, in a more malicious vein, code may contain viruses that have the potential to spread throughout the tree ('malware'). If the official repository is made accessible to a multitude of trusted persons, considerable damage may result. The risk taken is not insignificant, while cleaning a spoiled tree is a nuisance that can take many hours of painstaking work. One has to roll back to earlier versions of the repository and start anew without the contested code. Subsequent source code changes have to be reintroduced one after the other. Although in practice larger OSS projects are split up into modules that run in parallel, thereby reducing this risk, some element of risk still remains.

Project leaders in OSS are often acutely aware of the problem. From an online survey it transpired that sourceforge developers do consider interpersonal trust important for the effectiveness of OSS communities (Lane et al. 2004). In particular they identified obtaining write access to the repository as a matter of trust that has to be gained by adequate performance. Compare the following quotations:

> Once a potential project participant has proved his/her interest by submitting relevant code changes and expressing an interest to write more code, this is

> normally enough for them to gain the trust of existing project members. Once trusted a participant is typically given commit rights to the source repository, and can thus freely change the code base.
>
> Free Software is generally a trusting community. However, it is generally accepted that a new guy is not trusted. This means that a new guy can't just write an email to the developer's list and get write access to the project's CVS.[1] A new guy has to build trust with the project by submitting patches, useful criticism, help, and testing, and so forth. Before someone can have write access to CVS, they generally have to demonstrate programming skills, an ability to take criticism and use it constructively, work with a team, and show that they are willing to work to resolve problems.

The arch-father of Linux, the largest OSS project ever, is also aware that trust is at stake. By nature not a trusting kind of person, Linus Torvalds only extends his trust to a few chosen lieutenants:

> (…) I'm afraid that I don't like the idea of having developers do their own updates in my kernel source tree. (…) I know that's how others do it, and maybe I'm paranoid, but there really aren't that many people that I trust enough to give write permissions to the kernel tree. (retrieved from http://lkml.indiana.edu/hypermail/linux/kernel/9602/1096.html)

A similar analysis applies to open-source encyclopaedias. The entries that collectively make up an encyclopaedia can obviously be spoiled by contributors with mala fide intentions and/or poor capabilities. Wikipedia in particular has by now accumulated ample experience on this point and developed an amusing typology of such participants (http://en.wikipedia.org/wiki/Wikipedia: RCO; henceforth for all English Wikipedia references the prefix http://en.wikipedia.org/wiki will be omitted but presumed as the default). 'Cranks' insert nonsense, 'trolls' and 'flamers' stir up trouble, 'amateurs' disturb entries with their fake knowledge, 'partisans' smuggle their point of view into entries, and 'advertizers' find subtle ways to promote their products. As a result, entries become unbalanced at best, unreliable at worst. Having to monitor and redress such disturbances is obviously a major task (more on this below).

So here, too, providing write access to the project's official body of content is a matter of trust: dependence, vulnerability and risk are all involved. Inside Wikipedia, its volunteer 'officials' are well aware that trust is involved. This is most clearly shown by growing concerns about two issues. On the one hand, the reliability of *entries* has become a source of concern. In response, various quality measures and procedures for verifying them have been introduced ('good' article, 'featured' article). Moreover, software schemes that colour chunks of text according to their reliability are in the making. On the other hand, the trustworthiness of *contributors* themselves is becoming a critical issue. Should levels of 'trusted users' be distinguished—as opposed to ordinary users? Should such 'trusted users' specifically be charged to carry out quality inspection of entries? Should such users police vandalizing contributors? Obviously, the two initiatives, directed towards both entries and the contributors behind them, are seen as interrelated.

This issue of trust is explored below by a close analysis of developments in both OSS communities (FreeBSD and Mozilla in particular) and encyclopaedic communities (Wikipedia in particular). This selection of cases is meant to cover some typical open-source communities that currently exist. The central argument about the handling of trust—whether by inference, assumption, or substitution—can be briefly summarized as follows. In an initial phase when projects are still small they usually rely exclusively on the first two mechanisms ('informal phase').[2] In this respect OSS could rely on a culture common to the community as a whole: the 'hacker ethic'. Contributors could be supposed to adhere to this ethic and therefore be considered trustworthy enough. Wikipedia faced a much harder problem. When it started, no relevant common culture was in existence. As a result, trustworthiness could not be inferred directly in any plausible way; the only option was simply to go ahead and assume contributors were trustworthy. Was this assumption based on any rational underpinnings? The answer is found not so much in the mechanism of seeking esteem (as proposed by Pettit), but rather in the mechanism of substantial hope (as proposed by McGeer). Potential contributors were called onto develop and apply their encyclopaedic skills. To be sure, in order to fill the cultural vacuum, a 'wikiquette' soon enough came to be developed inside Wikipedia as an analogue of the hacker ethic.

In time rules and regulations were introduced, relating in particular to a division of roles and decision making (de Laat 2007). This is a common development as soon as projects grow, both in terms of the number of participants and the size of content created. In order to manage the complexities involved, project leaders experience the need

[1] CVS denotes Concurrent Versioning System: a software tool that allows distributed contributors to work efficiently together on the same tree of source code.

[2] It has to be borne in mind though that most projects starting *at present* will immediate adopt roles and procedures, simply because standard models for them are available for the taking. As a result, all three mechanisms for handling trust come to the fore simultaneously.

to structure their projects. The link with trust is that rules may substitute for trust—and so reduce the trust needed. It is important to emphasize, though, that such governance by rules and regulations may vary across projects. On the one hand, rules may be designed starting from the premise that participants can be fully trusted. A maximum amount of participant discretion will be designed in, so to speak. On the other hand, the leading presumption may be the opposite: participants cannot be trusted to deliver reliable content of their own accord, so as little discretion as possible—without stifling voluntary contribution altogether—is granted by the structural design. A low-discretion design signalling low trust is the outcome. In between these extremes, a continuum ranges from high to low discretionary design. It is argued that the design of open-source software and encyclopaedia production seems to be converging to a medium level of discretion.

Open-source software: hacker ethic

The origins of the OSS movement go back to the 1980s. Hackers—as they prefer to call themselves—used to freely exchange pieces of source code they had written. Large companies then started to enforce some of their alleged intellectual property rights over software. In particular, AT&T sought to protect UNIX. In response, hackers rallied together in an effort to keep the source code free (i.e., freely available). As a result, famous packages like FreeBSD (with a BSD licence) and the GNU Emacs editor (with a GPL) were developed. In the early 1990s, the Internet—itself largely the fruit of such open-source practices—boosted participation in open-source projects. People from anywhere around the globe could join with one click of the mouse. As a result, the number of OSS projects and participants rose sharply. Estimates of the total number of OSS projects currently underway amount to over 100,000 (on platforms like Freshmeat and Sourceforge).

Precisely this Internet-boosted era is central to my investigation. In the initial stages at least, regulating rules were few and far between. Someone usually initiated a project by putting a source code proposal on the web and inviting comments, patches and new features. This initiator—usually male—then operated as project leader, trying to manage the whole undertaking. The number of people responding could assume astonishing proportions. Larger projects easily attracted the attention of thousands of people out there (as evidenced by their downloads); among those, hundreds might actually send input back to the project, whether comments or code.

An astonishing feature of this open-source process is the near-total trust invested in strangers—outside a core of close friends, which often also existed. After having made a few useful patches, contributors were easily welcomed as developer, with permission to upload code into the official tree of the project. This was a big 'bazaar' indeed, virtually without rank and distinction, all babbling together and hacking away at the code tree. But what about the quality of these return gifts of code? Might some of these possibly be misguided, poorly formulated, misleading, outright irrelevant, or even poisonous? Might the code tree become corrupted as a result? All these objections notwithstanding, leaders—at least in this initial phase—practiced near-total trust towards strangers and did not attempt to delineate trust more carefully. This only happened later.

On what was this trust based? In 'real life', when we meet people, we are used to being able to infer some degree of trustworthiness from their characteristics. People's family background, ethnicity or sex may be interpreted as providing trust in the prospective transaction (characteristic-based trust; cf. Zucker 1986). Put otherwise, these ascribed characteristics serve as flags that *signal* trustworthiness to observers. On the Internet, though—at least initially—no such inferences about contributors are usually available. Characteristics that might give clues about trustworthiness are hidden from view. All one has are IP addresses that present themselves, hopefully with useful comments and code contributions.

Of course, these initial contributions could breed some trust. To some extent leaders could infer trustworthiness from past performance. In addition, however, the open source hackers of the Internet era who were appealing to unknown others to show their hacking capabilities did not operate in a complete cultural void. When the Internet opened up avenues for massive participation, they had the cooperative experiences of decades behind them, albeit on a much smaller scale. These were imbued with what has come to be labelled the 'hacker ethic'. This conception was first coined by Steven Levy (1984) to describe computer wizards from the 1960s to the 1990s. True hacking as a way of life revolves around spectacular and novel ways of using the available capabilities of computing. Throughout, the emphasis is on constructive cooperation and sharing. Bureaucracy, security, passwords, and copyrights are detested as just so many bureaucratic impediments to fruitful exchange. Pekka Himanen (2001) suggested even grander dimensions for the ethic of the 1990s hackers, closely tied to OSS development. In his vision, such hacking is a creative passion that is embedded in a new work ethic for the information age, which focuses on sharing information and keeping the Internet open for all, in a spirit of caring for all.

In the 1990s hackers of such persuasion took the Internet route to developing OSS together, a move that came to address a potentially much *broader* audience. Could it be presupposed that the audience involved would be bound by

the same ethical standards as 'true' hackers? My answer is in the affirmative. The broader audience was on the one hand composed of 'true' hackers living all over on the globe, and on the other hand of members of the computer underground, the 'new' hackers so to speak. The writings of the latter bloc have been analyzed by Steven Mizrach (1997). After distinguishing several categories of this underground movement (such as system intruders, phone phreaks and virus writers), he surveyed the do's and don'ts in their ethical self-conception. He finds a considerable continuity between 'old' and 'new' hacker ethic. The following principles in the new hacker ethic are particularly relevant for my purposes: share and exchange information with other people; do not just take information and software from other people (no hoarding, no freeloading); do not damage anything when entering other computers or data systems; do not crash others' systems by destroying hardware or data, by unleashing viruses, Trojans, or logical bombs.

So new hackers also predominantly cooperate and share, while avoiding damage and harm—in spite of the bad press reports about the few that deviate from this moral baseline ('crackers'). This suggests that opening up source code proposals to the world at large was not so irrational after all. Amid the potentially thousands of downloaders, only a small fraction could be expected to be knowledgeable enough to be able to reciprocate. And that fraction would seem to be bound by some kind of hacker ethic, whether of the old or the new variety. The inference that their contributions can be trusted seemed to be warranted. It was only much later that such inferences could no longer be upheld, a point that is explored below.

My conclusion therefore is that some kind of hacker ethic was shared by the audience at large—at least by those who were returning comments or code. As a result, this hacker ethic was instrumental in making the OSS experiment successful. Obviously, the question that arises is whether and to what extent my analysis is shared by the originators? That is, did they form an estimate of the constructive attitude of global audiences, was that estimate the same as mine, and did they *consciously* interpret it as support for their open-source experiment on the scale of the Internet? If the answer is positive, it implies that they took a reasoned step when deciding to open up source code on the Internet. If the answer is in the negative, however, it implies that OSS originators embarked on a considerable gamble by assuming trust worldwide. Someone like Eric Raymond, who is steadfast in his portrayal of the phone-phreaks involved as detestable and delinquent crackers, is a case in point. In any case, there is to my knowledge no more systematic or wider scale answer available.

Note that in my analysis culture plays a modest role in creating trust. Being a true hacker cannot be reliably signalled, due to the nature of virtual communications. A potential trustor can thus never be sure that a specific potential trustee indeed *is* a member of the hacker tribe. (S)he only may obtain assurances of a statistical nature, namely that the hacker ethic obtains *in general*. It is only a climate of trust that is gauged (cf. Baier 1992, Pettit 1995). A strong form of inference is replaced by a much weaker one.

Wikipedia: encyclopaedic ideals

The movement to produce encyclopaedic entries in open-source fashion is more recent in origin. It all started in 2000 with the American Nupedia, written and reviewed by experts. While that undertaking was slow to take off, Wikipedia, under the leadership of Jimmy Wales, was launched as a kind of experiment. Everybody, unregistered and anonymous, became entitled to read *and* 'edit' entries in the online encyclopaedia ('editing' meaning changing, deleting, or adding content). Three 'pillars' have to be observed in the process. 'Neutral point of view' means that articles should represent all significant viewpoints to an issue fairly, proportionately and without bias (Wikipedia: NPOV); 'no original research' means there is no room for original research of one's own that has not yet been published elsewhere (Wikipedia: NOR); and 'verifiability' means that all content that is likely to be challenged should be traceable back to a reliable source (Wikipedia: V). Soon enough this was a great success, at least numerically. Local Wikipedias were created, in languages other than English, numbering over 250 at the time of writing. The largest Wikipedia (in English) contains over three million articles, while a small one, such as the experimental one in Kirundi, as yet contains only seven.

This movement is to a large extent modelled on OSS experiments. The basic software tool, a wiki, allows distributed participants to work on the same body of text simultaneously (just as versioning systems do for source code). Wikipedia also admits everybody as contributor (denominated as 'editor'), and makes entries available to everybody (with a GNU Public Documentation Licence tailored to texts, the equivalent of the GPL for software). So here again, just as with OSS, we find almost unlimited trust in strangers from all over the globe. The numbers involved are impressive. For the English Wikipedia alone, apart from anyone being allowed to edit entries, 9.2 million people have registered, and, as a result, may additionally start a new article of their own. Roughly 1/3 of edits are made anonymously, while 2/3 originate with registered users. A similar ratio applies to other language versions.

Again we may pose the question: what mechanism of trust is involved? What grounds can be advanced for

trusting outsiders not to damage entries, introduce minor or major mistakes, or edit the details of their own biographies? Again we encounter the problem that signalling trustworthiness is difficult—most of the time IP-addresses are the only signs available. This time, unlike the case of OSS, no cultural supports for their initiative were to be found; no relevant ethic had been developed as yet. It has to be concluded, therefore, that we are dealing with an absolute *assumption* of trust—the point of departure for dealing with outside contributors being that they can be fully trusted to contribute to the worthy cause of an encyclopaedia by all and for all. But then, we may continue to ask, can any good reasons be provided for this assumption?

More than a decade ago, Philip Pettit (1995) proposed the following underlying mechanism (dubbed 'secondary trust'): since people are sensitive to the esteem of others, they will reply favourably to acts of trust in order to actually reap this esteem. The chance to be admired cannot be forfeited. Alternatively, people cherish some amount of self-esteem and will therefore behave in a trustworthy fashion in order to avoid feelings of shame. In our encyclopaedic context this reads as follows: outsiders who stumble on this Web 2.0 experiment may decide to contribute since they are seduced by the prospect of being admired by core Wikipedians. Alternatively, they will refrain from vandalizing contributions since they want to avoid embarrassing themselves. This imputation of 'normative pressure' (in the Luhmannian sense) to the gesture of entrusting a large body of text to the public at large might have some plausibility.

Recently, though, this mechanism of normative pressure has been reformulated by Victoria McGeer (2008). She tries to move away from the calculating and cynical conception of as-if trust in Pettit's formulation. Instead, she focuses on moves inspired by the kind of trust that does not rely on coldly weighing the evidence available but is prepared to go beyond ('substantial trust'). This is based on a vision of and hope in the capabilities of the other. The trusting move hopefully energizes the other to realize his capabilities to the full. Such trust is empowering the other, not—à la Pettit—seducing or manipulating the other. As the prototype of this hopeful trust she presents the example of parents who at appropriate times have to let their offspring go and engage in risky adventures. Such trust is a hopeful wager on a future in which their children will be able to take care of themselves.

To me, her theorizing about the assumption of trust seems to be the more plausible avenue in the case of Wikipedia. By opening up their entries to immediate modification, contributors appeal to the encyclopaedic capabilities of unknown others 'out there'. They are expressing *substantial* trust in unknown others based on a vision of the ultimate attainment of the encyclopaedic ideal of knowledge accessible to all and developed by all. These anonymous visitors are challenged to show what they are worth as commentators and/or composers of text. It is not so much esteem or self-esteem that is at stake; it is the prospect of exercising and developing one's encyclopaedic capabilities that spurs fellow participants into action. A continuing cycle of high-quality contributions may ensue.

Inside Wikipedia there was full awareness that trust was mainly an assumption based on such hopes. The assumption of good faith in one's co-editors is stressed repeatedly:

> Assuming good faith is a fundamental principle on Wikipedia: it is the assumption that editors' edits and comments are made in good faith. Most people try to help the project, not hurt it. If this were false, a project like Wikipedia would be doomed from the beginning. (Wikipedia: AGF)

Even faced with evidence to the contrary ('vandalism'), one is urged to remain calm and cling to this assumption as long as possible. Moreover the theme of 'empowerment' of newcomers emerges clearly.

> Remember, our motto and our invitation to the newcomer is **be bold** [emphasis in original]. (…) Understand that newcomers are both necessary for and valuable to the community. By empowering newcomers, we can improve the diversity of knowledge, perspectives, and ideals on Wikipedia, thereby preserving its neutrality and integrity as a resource and ultimately increasing its value. In fact, it has been found that newcomers are responsible for adding the majority of lasting content to Wikipedia (*i.e.*, substantive edits) (…). (Wikipedia: BITE)

A final quotation serves to show a clear awareness of the fact that the fog of cyberspace causes a lack of clues and almost forces an attitude of blanket trust without apparent justification:

> Attempting to believe the best of your fellow Wikipedians, and they of you, helps to eliminate some of the problems that arise when we communicate only in text, and cannot use all the verbal and visual cues used in talking face-to-face. (Wikipedia: CIV)

Wikiquette

So the Wikipedia adventure was founded on hope and vision, but *not* on a full-blown underlying ethic among potential contributors (as in the hackers' case). Such an ethic simply did not exist. Therefore, at the time Wikipedia started, no one could forecast how people 'out there' would

react. An encyclopaedic community was a novelty. In that sense Wikipedia was more of an adventure than OSS, with its decades of experience before the Internet boosted collaboration.

Problems surfaced soon enough, though. Several varieties of 'disruptive' behaviour emerged (Wikipedia: BP): on the one hand, vandalism of entries (like changing small details, inserting nonsense, adding obscenities or crude humour, blanking pages, and changing details of one's own life; cf. Wikipedia: VAN); on the other hand, gross incivilities, persistent harassment, and threats or attacks against editors personally (on discussion forums, on talk pages, by e-mail, etc.). A specific term was coined: 'edit warring', referring to contributors fighting over the contents of an entry from their own point of view by repeatedly deleting each other's changes (in the Wikipedian jargon these are called 'revert edits', returning an article to an earlier version). Notice that the reverts need not be unjustified as such; rather it is the lack of any explicit comment or justification that may make the act, to many a participant, rude and insulting. All kinds of rules were devised to deal with the phenomenon *after* the fact and they are analyzed below. What matters here is that simultaneously a kind of cultural offensive was launched to *develop* a kind of 'Wikipedian ethic' and fill the ethical vacuum that existed at the outset of the experiment. The true Wikipedian had to be constructed de novo. To that effect, the original guideline at the start of assuming good faith (Wikipedia: AGF) was expanded into a coherent set of ethical rules for conduct.

Several entries in Wikipedia testify to these efforts. Starting from the assumption of good faith, help other editors gently to correct their mistakes if any (like introducing 'original research' or attacking someone personally) (Wikipedia: AGF). Do not accuse anyone lightly of bad faith; and above all, do not forget to show your own good faith. The focus is on constructive argument in order to be able to reach consensus over articles. Proposals for textual changes or deletions should always be accompanied by arguments (in the 'talk pages', with one's name and date attached). In the process, one should be civil and avoid incivility (Wikipedia: CIV). Civility means: a considerate, polite and respectful attitude to others (remember the Golden Rule) in discussing differences of opinion. Incivility—to be avoided—means: being rude, uttering insults or profanities, personally attacking or harassing other editors, and the like. The atmosphere, moreover, should be open and warm. Turn the other cheek if necessary, give praise, and forgive! (Wikipedia: EQ).

This 'Wikiquette'—as it is aptly called—is repeated in guidelines for newcomers. As regards content, newcomers are urged not to contribute an article about themselves or their company, to add or delete content with caution and with arguments only, and to avoid chatting or flaming (Wikipedia: ACM). As for treating newcomers, established Wikipedians are advised not to bite them and so scare them away with hostility (Wikipedia: BITE). Be respectful and constructive in correcting them (=civility); make them feel welcome (=warm atmosphere); assume good faith on their part (=good faith). Give them a chance! *Ignorantia juris* (i.e., of Wikipedia law) and inexperience may be excused. One can have faith in this approach, it is asserted, while 'many new users who lack an intuitive grasp of Wikipedia customs are gradually brought around once the logic behind them becomes more clear' (Wikipedia: AGF). This approach is carried as far as making available pre-fabricated templates cordially welcoming newbies into a project (Wikipedia: WT).

So here we clearly find the articulation and fostering of a 'Wikipedian ethic' that is to underpin the basic premise of trust towards fellow-Wikipedians. It is a civilization campaign to keep the open approach to editing viable and alive. The true Wikipedian had to be co-constructed with the organizational design of Wikipedia (which is explored below). While in OSS culture preceded structure, with Wikipedia these have been evolving *simultaneously*. As a result, confidence in fellow-Wikipedians has gradually become more warranted. The trust involved is no longer purely an assumption, but has partly become based on inference; in particular on weak inference based on statistical reasoning. Note, though, that this 'civilization effect' becomes weaker and weaker as we move from more experienced Wikipedians towards less experiences ones. By definition, newcomers to the project are not even aware of—let alone bound by—any Wikipedian ethic.

Open-source software: rules and regulations

In OSS, the 'bazaar' soon came to be regulated. The 'simple structure' of project leaders-cum-followers no longer sufficed, especially in projects that were growing in size. Efficient management necessitated the introduction of rules, regulations, and prescriptions. The most important governance tools introduced in this formal phase are the following (as described in de Laat 2007):

- *Modularization*. In many larger projects, the code tree is divided into several subtrees. In this fashion, up to dozens of modules may be carried out in parallel.
- *Formalization.* Technical tools and standardized procedures have been introduced to streamline virtual cooperation. This applies to mutual discussion, reporting of bugs, working on the code tree simultaneously, and testing.
- *Division of roles*. In almost any project nowadays roles are distinguished that define what the occupant is

allowed—and expected—to do inside the project. A common role division consists of observer, developer and project owner.

- *Decision making.* In every project decisions have to be made about a range of matters; for example, the methods to be used, acceptance of code in the main tree, and preparing new releases. Decision-making powers in such matters are formalized and in some way distributed among participants.

As a result of the introduction of these governance tools OSS projects assume a design. Differences in size, technology, and phase of maturity will impinge on the shape of a proper design; assumed designs will therefore vary among projects. Moreover, for any individual OSS project there will be a variety of possible designs; not one but many options are open.

Can the introduction of governance in a project be related to our issue of trusting virtual hackers in cyberspace? As argued in the introduction, the argument in general is that structuring may substitute the need for establishing trustworthiness by restricting freedoms. Would this mechanism by any chance apply to the specific context of OSS? I intend to show in detail that it does: design parameters do indeed substitute the need for trust, but the extent to which members of the institution appreciate and welcome this substitution varies.

In the following argument I start from considerations that have been developed for 'real life' industrial organisations by Fox (1974) (to return later to the focus of my concern, open-source communities). The argument proceeds as follows. Rules from a design impinge on the amount of *discretion* allowed to participants, discretion referring to the extent to which the exercise of activities calls for one's own wisdom, judgment, expertise—as opposed to following prescribed rules (Fox 1974: ch. 1). Discretion, in turn, can be interpreted as the amount of trust granted to participants by the institution involved. It represents the extent to which the institution consciously regulates participants' behaviour (Fox 1974: ch. 2). So discretion is conceived of as the lynch-pin between rule formation and trust. The usual effects of rule formation are thus as follows: rules reduce discretion by increasing prescription, thereby reducing dependence on the whims of organizational members. As a result, the amount of trust granted to them is reduced, and less trustworthiness is needed on their part. Rules indeed substitute in part for the need to rely on trust. This is thus the emergence of the third way of handling trust.

This relation between design rules, discretion and trust may, however, be accorded quite different valuations. Fox (1974) conceives of design parameters as exclusively *coercive*: they are the means by which leaders obtain control over the workings of their members and can enforce compliance. This is the usual function associated with bureaucratic rules. In this instance, rules reducing degrees of freedom—and thereby reducing discretion and granted trust—are just exercises in coercion. Two decades later, however, Adler and Borys (1996) argued that there can be more to rules than that. Organizational parameters may also have an *enabling* function: they are designed to enable employees to master their task in more professional ways. In my view this gives the following twist to the reduction of discretion and trust involved. The parameters in question focus on particular elements of discretion: those that are seen as inducing waste of time and inefficiencies. Curtailing these elements and no longer incorporating them in the 'trust package' is then easily perceived as empowering. In addition, as a rule recipients have more adequate tools placed at their disposal, which allows them to perform their tasks in a novel and more professional fashion. Substitution of trust is inspired by intentions to empower organizational members—not to control them.

So far, this has been a classic organizational analysis. As the next step in the argument I seek to maintain that a hierarchy is also to be obeyed in virtual communities like OSS. Project leaders or owners have powers of regulation that are embedded in the virtual design of their projects. Contributors may choose their tasks and leave whenever they want, but in the meantime access to files has to be granted, write access given, certain tools are obligatory, and procedures have to be respected. Volunteers have to play by these rules in order to be accepted by the community; otherwise they are just ignored. In view of this hierarchy nexus there is a close correspondence between organizations on the one hand and open-source communities on the other. As a result, the above organizational analysis concerning rules, discretion and trust is *mutatis mutandis* also applicable to such communities. In particular, design parameters in OSS communities—as analogues of similar organizational parameters—may be interpreted as being either of the coercive or the enabling kind.

Returning now to the four parameters for OSS distinguished above, how are they to be interpreted along these lines of inquiry? Some would clearly seem to be enabling in the sense just coined. Both modularization and formalization streamline an otherwise chaotic process. This is structuring as a minimal condition for fruitful collaboration to occur. Formalization tools in particular may function as tools that enhance programming capabilities (cf. Andrews et al. 2005) and streamline software development (cf. examples in Adler and Borys 1996). As a result, although fewer freedoms are actually granted to contributors, this is welcomed as leveraging members' skills, not detested as encroaching control. Robbins (2005) calls our attention to the remarkable fact that the *whole* OSS community has

standardized on the use of such formalization tools. I would suggest this is so precisely because they are experienced as empowering by all hackers concerned.

The introduction of role divisions and powers of decision making in OSS, on the other hand, is a more delicate affair. These parameters may easily be considered controlling, as they touch directly on the amount of essential discretion that outside collaborators may enjoy. The choices for project leadership lie on a continuum. To make this clear, I refer to the famous distinction between high-discretion and low-discretion work role patterns in organizations, developed in Fox (1974), and adapt it to my own purposes in relation to OSS communities. At one end of the scale, the role division employed in OSS may be minimal and decision-making decentralized. The discretion granted remains high—as high as the particular project allows. In such a *high-discretion design*, trust in virtual strangers remains high. As a result, outside contributors may be expected to remain committed to the project and continue contributing code or comments. At the opposite end of the scale, a *low-discretion design* can be introduced in OSS: an elaborate division of roles is carved out, with minimal discretion for the lower echelons and decisional powers highly centralized. Such a design effectively awards little trust to contributors. While hackers may be supposed to be attached to autonomy in their voluntary activities, they might well interpret the design as a manoeuvre of control, a way of expressing distrust in their very capacities and/or intentions. As a result, volunteers might be chased away from continued participation and enlist elsewhere.

Take Tigris, for example, a well-known platform that hosts many OSS projects. The site carries an explanation that a role consists of a set of permissions granted; a permission allows specific activities (like reading or editing) to be performed on specific resources on the project's site (like project documents or source code files) (http://www.tigris.org/scdocs/DomAdminRoles.html.en). A common threefold division of roles is the following (cf. http://www.tigris.org/scdocs/ProjectRoles): an observer has read-only access to most of the project's documentation and source code files, and may return comments and/or code proposals and patches. A developer obtains more permissions: (s)he also obtains write access to the official source code tree and project text files. A project owner is someone at the top who manages the project as a whole (and part of the job is precisely to grant membership roles as just discussed). Note the pivotal role of developers: they are the ones who are empowered to incorporate changes in the code tree. To gain role permissions, candidates have to qualify—though standards seem quite relaxed. After surveying a project as an anonymous guest (who are allowed to see most of the project's files), one may ask to be granted observer status; this will as a rule be granted to anybody. Then, after delivering some contributions of sufficient quality to the project, one may obtain the status of developer. So one has to prove oneself sufficiently professional only if the status of developer is desired.

This design will do for many smaller projects, consisting of just a few modules. Each module is run with an 'owner' at the top. The design is still quite similar to the 'simple structure' from the initial, informal phase—still with a rather large amount of discretion, but the structure will be evolving for larger projects: of necessity, the design will move in the direction of curtailing discretion. An often-discussed example, at the lower end of discretion, is Linux. In that vast project with a range of modules, 'trusted lieutenants' (above the layer of maintainers of the modules) are the ones who take all proposed changes into consideration, with the final say still exercised by Linus Torvalds. This is thus a very centralized design. In order to show that *intermediate* designs exist and to highlight the choices that can be made regarding role division and decision making, two other, larger OSS projects will now be analyzed in some detail: FreeBSD and Mozilla. These two projects have made slightly different choices as far as design is concerned.

FreeBSD and Mozilla

Let us first consider FreeBSD, an operating system that has been in development by volunteers for decades now (the following account is mainly based on Jørgensen (2001), but updated for recent developments by consulting http://www.freebsd.org/doc/en_US.ISO8859-1/books/dev-model/). In developing the source code modules the project distinguishes roles referred to as 'general hats'. The following roles are in use, quite comparable to those from Tigris: contributors, committers and maintainers. After accessing and exploring files anyone may start contributing comments and/or code. By definition one then becomes a *contributor*, no formalities are involved. *Committers* are contributors who have formally obtained write access to the code tree, and actually may commit code—either of their own, or from fellow-developers (without write access). *Maintainers* (recruited from the committers) are at the top of a module and coordinate the incorporation of new code.[3]

The procedure to be followed by a committer to get his code accepted is instructive. After writing a contribution he is urged first to discuss the changes with fellow committers.

[3] Notice that for this large project a whole range of other 'hats' are in use. Some 'administrative hats' with suggestive names are Documentation Project Manager, Quality Assurance, Release Coordination, Security Officer, Standards, and Bugmeister.

Preferably, code has to be reviewed several times. As the next step, he has to test the proposed changes by integrating them in the module involved and trying to compile the module as a whole (the build should not 'break'). This is a pre-test in his own downloaded copy of the current development version. If the test has succeeded, it is up to the committer to actually integrate the changes into the official code tree. From then on, the changes are open to debugging and commenting by anyone. So in effect, committers decide (semi-) autonomously about the incorporation of pieces of code. The fact that the results of one's work immediately become visible and available to all is reported to be a gratifying experience.

Not just anybody may rise up through this hierarchy. True, anyone may look around and/or contribute, but in order to become a committer, one has to qualify (http://www.freebsd.org/doc/en_US.ISO8859-1/books/dev-model/model-processes.html). After having made a number of high-quality contributions, a developer may request committer status. It is upon recommendation by a committer that the so-called Core Team (above the layer of maintainers, consisting of nine members) will vote about granting that status.

The other OSS project analyzed here is Mozilla, a well-known web browser, mail client (and more). In order to manage the development of the roughly 80 modules it employs basically the same division of roles as Tigris and FreeBSD (Holck and Jørgensen 2005; information updated by consulting http://www.mozilla.org/about/roles.html). Anyone is invited to contribute to the project (whether code, debugging, translation, add-ons, interface design, or documentation). After formal application a contributor may become a *committer* (acquiring write access to the tree). Modules as a whole are maintained by *module owners,* who appoint assistants to help them if needed (called *peers*). Their main task is to review all code changes to their module.[4]

The procedure for contributing code is identical to FreeBSD: discuss informally, test in your own working copy of the tree, incorporate in the main tree, and let others debug. There is one significant difference, though. The *decision* about incorporating code changes in the official tree is no longer within the committers' authority. For several years now they have needed to ask for permission in two steps (http://www-archive.mozilla.org/hacking/code-review-faq.html). First a contributor requests approval from his module owner. Thereafter a so-called 'super-review' is called for: someone, preferably with expertise in another domain, has to judge whether the patch fits into the broader Mozilla code base and give his approval. Several 'super-reviewers' have officially been appointed for the purpose.

In accordance with this design, the procedure to become a committer is quite strict (http://www.mozilla.org/hacking/committer/). Basically, contributors first need to demonstrate that they know what they are doing, by having contributed some good patches. After that, they may formally ask to obtain committer status. For this, they need two people who want to vouch for their competences. These 'vouchers', usually their module's owner or a peer, are responsible for them for a period of 3 months. Moreover, one of the super-reviewers must support their nomination.

From these descriptions it transpires that both FreeBSD and Mozilla are in the middle range of discretion granted to outside contributors. Write access to the code tree is never granted immediately or indiscriminately. However, decision making in Mozilla is more centralized than in FreeBSD. Supervision has tightened. The layers of module owners and super-reviewers are the judges of code incorporation, no longer the code developers themselves as in FreeBSD.

Influential Mozilla leaders (namely B. Eich and D. Hyatt) are quite explicit about the strategy that is to be followed:

> The faux-egalitarian model of CVS access and pan-tree hacking that evolved from the earliest days of Mozilla is coming to an end. (…) [One of the key elements in the new roadmap is to] continue the move away from an ownership model involving a large cloud of hackers with unlimited CVS access, to a model, more common in the open source world, of vigorously defended modules with strong leadership and clear delegation (…). (http://www-archive.mozilla.org/roadmap/roadmap-02-Apr-2003.html)

It may be concluded that as OSS projects grow in size and complexity, the trend towards introducing structure is ineluctable. In the process, every project will have to find its own design solution—within a range of possible solutions. Whichever design is chosen, some trust in evidence before is partly substituted by structure. As rules and regulations take over, the importance of inferring that outside, anonymous contributors are trustworthy is reduced.

As a corollary of this section on OSS design it should be noted that a recent trend can be discerned in many OSS projects that outsiders are perceived as less trustworthy from the outset (cf. de Laat 2007: p. 171). Fears seem to be

[4] Just as in FreeBSD several other roles have been coined for managing the whole undertaking: the 'super-reviewer' (performing an extra check on the integration of new code; cf. below), the 'bugzilla component owner' (managing the testing and debugging of a component of a module), the 'release driver' (managing the release of new versions), and the 'benevolent dictator' (handling conflict resolution) (http://www.mozilla.org/about/roles.html).

mounting about sloppy code, or buggy code, or even Trojan horses being introduced into the main tree of a project. Moreover, concerns about intellectual property rights are mounting: whether deliberately or not, contributors could incorporate code bearing a licence that is incompatible with the existing licensing arrangements, or could import patented matter—with all the attendant dangers of subsequent patent litigation. So the hacker ethic—as described above—is considered to be eroding. In response, potential committers of code are likely to be subjected to *screening*—that is, more screening than the usual procedure for ascertaining technical skills which amounts to demonstrating some good work (cf. the procedures as described above for FreeBSD and Mozilla). In Debian, for example, contributors who want to become 'real' developers with write access to the tree not only have to demonstrate their technical capabilities, they also have to reveal and prove their real-life identity, and show their factual understanding of and ideological attachment to the cause of OSS ('Debian new maintainer process'; described on http://debian.org/devel/join/newmaint; cf. also Coleman and Hill 2005). This tightening of entry qualifications is a clear move from inferring trust in a general sense (a 'climate' of trust) to inferring trust in a specific sense. Solid proof of being a 'true' hacker is required.

Wikipedia: rules and regulations

Let us now turn to developments in Wikipedia. In this project, too, rules and regulations soon came to be applied. I contend that the same type of analysis applies as that developed above for OSS. Governance tools were introduced, such as modularization (several subprojects were introduced under the umbrella of Wikimedia, such as Wikipedia, Wiktionary and Wikibooks; within them, each entry is a module in itself) and formalization (rules for editing pages, for discussing changes, for reporting and handling vandalism, and so on). These do not merit further discussion here since they clearly seem to belong to the category of 'enabling' tools in a virtual environment. In addition, a division of roles and rules for decision making was introduced. These parameters merit closer attention, since the resulting design may vary from a high-discretion design to a low-discretion one. Which kind of design actually did crystallize within Wikipedia? I intend to show, first, that the design that was gradually introduced is characterized by a very high degree of discretion—even higher than for OSS—and accordingly by great trust shown towards outside contributors. Furthermore, it is shown that, at the time of writing, pressures are mounting to reduce this discretion and be more careful about granting trust; all this as a response to increasing vandalism and harassment.

The best way to explore the design is to start with the division of roles (those employed within the English Wikipedia in particular) (Wikipedia: UAL). Just as in the Tigris division of roles, what this is all about is obtaining permission to perform specific activities upon specific resources on the site. Users come in three varieties. The *anonymous user* (no account) may read and edit all entries. As soon as (s)he has created an account (username, password), a *user* may in addition create new pages, as well as e-mail other users who publicly mention their address. A user automatically obtains the 'autoconfirmed' status with special privileges—such as moving pages, uploading files, and editing 'semi-protected' pages (to be explained below)—as soon as (s)he has made 10 edits and has been registered for 4 days in a row.

Above these user levels other roles have been defined that mostly have to do with protecting Wikipedia against disruptive behaviour. In order to hold disruptions in check, *administrators* (also known as *sysops*) obtain the right to protect pages, delete pages and block users (see below); about 850 of them are active at the time of writing. Very trusted users may become *bureaucrats*, who are entitled to appoint users as administrators or fellow bureaucrats (about 30 active at this time). Both role occupants are not simply appointed from above; they must apply formally and be accepted by the broader community after public discussion, which is usually lengthy.[5]

The administrators, as 'police officers' (my terminology) of Wikipedia, have several measures at their disposal to deal with incidents of disruptive behaviour. They may introduce 'page protection': a page involved in a dispute can no longer be edited or moved, usually for 7or 14 days (Wikipedia: PP). 'Full protection' means that no user is admitted, while 'semi-protection' means that autoconfirmed users still are. Whichever protection applies, administrators may still touch and edit the page involved. Protection is useful as a cease-fire period, to allow contestants to resolve their conflicts. Similarly, administrators may delete pages, as a last resort, to deal with vandalism, but also in cases where reliable resources are absent, or copyrights are violated (Wikipedia: DEL).

Another measure that administrators can apply in case of serious disruptions by *particular* users is to 'block' them, that is, bar them from any further activities on the site (except for reading) (Wikipedia: BP). A block can last a day or longer, depending on the circumstances. It is

[5] Although these are the main roles, a whole array of others can be found that I shall not go into here such as: the 'rollbacker' user (who obtain the means and the authority to quickly revert revisions), the 'checkuser' (who may run a check on all IP-addresses used by accounts of users suspected of misdemeanour), and the 'oversight' user (who may permanently hide revisions of pages from all users) (Wikipedia: RIGHTS).

intended to protect the Wikipedia project, not to punish the user involved. A particular example of behaviour that is considered incontrovertible evidence of edit warring is performing three reverts of any single page within 24 hours. Anyone indulging in this may be blocked from further editing (the 'three-revert rule'; Wikipedia: 3RR). In the same vein, someone who stealthily creates an alternative account so as to push their personal point of view twice, or to stir up controversy (a 'sock puppet') also risks being blocked (Wikipedia: SOCK). The same measure may apply to a user who lets someone else register in order to push his/her *own* point of view ('meatpuppet'), or alternatively push the *opposite* point of view ('straw man sock puppet').

Note that the police officers involved are also assisted by software bots, which are entitled to revert changes that most obviously emanate from vandalism or edit warring. As a result of this policing of Wikipedia by both men and machines, such changes are reportedly corrected within a very short period of time (median correction time from 2 to 3 min; figures refer to October 2005; Viegas et al. 2007). This rapid repair mechanism is a form of what is called 'soft security'—protecting the system in unobtrusive ways, almost invisibly, and after the fact.

How may one characterize the design crystallized within the English Wikipedia? Users may edit at will, change whatever they like, even if they are unregistered and anonymous. Trust in their potential contributions is high. Meanwhile, as we have seen, limits have been set on contributing: no edit wars, no vandalism, and no harassment. Rules such as the three-revert-rule or the sockpuppet rule are bureaucratic rules that define how far one may go. Enforcement of these rules is entrusted to administrators who may apply disciplinary sanctions. However, most Wikipedians consider these rules necessary for conflict resolution. Moreover, the 'officials' involved are urged to 'exercise care' and 'behave in a respectful, civil manner' (Wikipedia: ADM), resolve conflict by 'patience and talking', and stick as long as possible to the basic premise that fellow Wikipedians are acting 'in good faith' (Wikipedia: AFNN). Taking all these elements together I would argue that the design was—and still is—a high-discretion one, with the regular user at the helm, as far as that is reasonably possible in an open environment. This design still applies, presumably, while users are considered to be the main assets on which the destiny of the encyclopaedia depends. Without their massive voluntary collaboration, Wikipedia would have suffered the same fate as Nupedia. Peer production of (encyclopaedic) knowledge is taken to its logical limit: 'democratic' production by all.

Remarkably, the trust granted is even higher than was ever the case in OSS. For software, contributors who want to acquire write access to the code tree have always been held to some proof of their coding capabilities. Becoming a committer was always a privilege. Moreover, maintainers often do own modules in the sense of actively overseeing code patches from committers. In Wikipedia, by contrast, users have always had immediate write access to articles and other pages, no capability test or review of submitted changes is required. The motto is: let everyone be a Wikipedia committer.

Flagged revisions scheme

However, there are increasing signs that the era of unquestioning trust in strangers is coming to an end. Discretion is circumscribed in tiny steps. A first, hardly noticeable step had to do with an incident concerning the journalist John Seigenthaler: an anonymous user created a biographical entry about him containing false content (May 2005; Wikipedia: Seigenthaler_hoax). It went undetected for several months. In response, Wales barred unregistered users from creating new pages. From December 2005 onwards, an anonymous visitor may no longer create a new page but only read and edit existing entries.

A more serious encroachment on full discretion for all is the call for review: all changes should be checked for vandalism before incorporating them in the 'stable' version of a page. While the system is still under discussion for the English Wikipedia, it has already been unrolled gradually in the German, Russian, Hungarian, Polish and Arabic versions (since 2008). The software involved can create many varieties of reviewing systems. As regards the kind of entries to be reviewed, some argue for reviewing only the most sensitive ones (like biographies of living persons), others for reviewing all of them. Furthermore, who is to be censured? Only anonymous users, or contributors at large? As for the reviewers themselves, should they be a select group of trusted users, or all (registered) users? And finally there is the question of what a visitor actually gets to see on the screen: the 'stable' version, or the 'experimental' version containing one or more as yet unreviewed edits?

The approach chosen by the German Wikipedia is the following (German Wikipedia: Gesichtete_Versionen; Wikipedia: Flagged revisions/Sighted versions; http://de.labs.wikimedia.org/wiki/Hauptseite). All entries fall under the system, and edits from all contributors are to be reviewed (but see exception below). Reviews are carried out by *Sichter* (literally: sifters). Registered users automatically obtain such rights ('aktive Sichterrechte') after performing at least 300 edits and having been active for at least 60 days. With less experience—at least 150 edits and 30 days of activity—users become exempted from the review process themselves: their edits or articles do *not* need to be reviewed and automatically turn up in the

'stable' version ('autoreview rights'; 'passive Sichterrechte') (German Wikipedia: Gesichtete_Versionen). On the screen, unregistered users get to see the so-called sighted (flagged) version (as default)—although they can click on the newest, unsighted version if they wish. For registered users the most recent version is the default. The trick of this default setting is that anonymous vandalism no longer gains immediate gratification: their changes do not show up in the official version.

While Wikipedians in the German tongue (and some other languages, mainly Eastern European) are mostly satisfied with the flagged revisions scheme, it is hard to swallow for those using the English tongue. Heated and protracted discussion is still raging on talk pages. Various alternatives are being proposed, such as 'delayed revisions'—delaying edits made by users below the autoconfirmed status: they turn up in the official version after 2 hours (Wikipedia: Delayed revisions); or 'deferred revisions'—only suspect edits, identified by an abuse filter, come under the flagged revisions scheme (Wikipedia: Deferred revisions). There is also vehement defence of the most recent version being shown to *all* users.

An opponent of the flagged revisions scheme observes: "The idea that we trust some users more than others on content is terrible" (17 January 2009; English Wikipedia, on the talk page about flagged revisions). Indeed, such trust as is granted is effectively differentiated by the scheme. Lines of division are drawn: between those who may do the reviewing and those who may not (*Sichter* vs. other users); between those who may edit without review and those who may not (users with autoreview rights vs. users without them); and between those who get to see the most recent version immediately, and those who are referred to the (possibly less recent) 'stable' version (registered users vs. anonymous users). The autoreview rights in particular are interpreted as creating a new elite by themselves. Trustworthiness regarding contributions is no longer assumed from the outset; it has to be demonstrated by one's editing track record within Wikipedia. The mechanism of inferring trust from past performance is therefore instituted in connection with these new roles. Remarkably, in the initial German proposal the threshold for obtaining auto-review rights was considerably *higher* than for obtaining 'aktive Sichterrechte'; fierce discussion obviously lowered the threshold.

Effectively, Wikipedia's design is moving towards a lower level of discretion; a move that is laudable to most German Wikipedians, but detestable to most English ones. This difference in appreciation can be linked to a distinction made earlier: between enabling and constraining rules. Obviously, the English feel offended by the curtailing of their discretion and the attendant differentiation of privileges. The scheme is interpreted as outright constraint—to them, bureaucracy (in the pejorative sense of the term) is setting in. On the other hand, the German interpretation of the flagged revisions scheme is that these bureaucratic rules enable the proper working of the encyclopaedia while curbing vandalism. Discretion is gladly sacrificed in order to gain in efficiency. Why, in the end, English speaking and German speaking Wikipedians differ so much in their diagnosis is an intriguing question—one that is ripe for investigation.

In the near future, Wikipedia—more surely the German than the English language version—also intends to introduce a more strict kind of review, one that checks the quality of articles. This will be done by *Prüfer*, also referred to as über-reviewers or surveyors. A useful comparison with Citizendium can be made here. That open source general encyclopaedia of more recent origin (2007) explicitly honours expertise, and distinguishes the roles of 'authors', 'editors' and 'constables' (http://en.citizendium.org/). Their 'authors' are comparable to registered users in Wikipedia, their 'constables' are the equivalent of administrators in Wikipedia. And the Citizendium 'editors'? These are acknowledged experts who guide the crafting of articles and approve the various versions; they come close to the proposed *Prüfer*. So in conception and design Wikipedia can be seen to be moving closer to Citizendium. Finally, note that Wikipedian *Prüfer* and Citizendium 'editors' are functionally similar to module owners in the average OSS project. As regards design, open-source encyclopaedias and OSS are arguably converging. One difference remains, though: code repositories are *owned* by the leader(s) of the project, while encyclopaedic entries are owned by nobody.

Conclusions

Open-source communities rely squarely on the contributions of mostly anonymous strangers in cyberspace, so a central concern, whenever these focus on ever-evolving content, such as software modules or encyclopaedic entries, is the problem of whether and to what extent such volunteers can be trusted to contribute in good faith and in a competent fashion. It has been argued that such communities do indeed have a whole array of mechanisms at their disposal to handle this matter of trust: Table 1 lists the mechanisms involved.

When rules and regulations are still few and far between, the full weight falls on the processes of inferring or assuming the presence of trust. Some assurances may be generated by an underlying shared culture. When hackers started to use the Internet to develop OSS, they had been cooperating with each other for decades in 'real life' and developing a shared 'hacker ethic'. On the Internet they

Table 1 Open-source communities and mechanisms used for handling the problem of trusting contributors

	Open-source software	Encyclopaedias
Assumption of trust (substantial trust)		Trust in 'encyclopaedic' capabilities
Inference of trust (weak form)	Hacker ethic	Wikiquette
Inference of trust (strong form)	Past performance	Past performance
	Entry examinations	
Substitution of trust (from a small to a large extent)	Design (from high-discretion to low-discretion)	Design (from high-discretion to low-discretion)
Examples discussed in more detail	*FreeBSD*	*Wikipedia*
	Mozilla	*Citizendium*

now bumped into a younger generation of new hackers. If Mizrach (1997) is right, their 'new hacker ethic' shared a considerable overlap with the old one. As a result the potential open source audience as a whole was tied to some kind of hacker ethic and their trustworthiness would appear to be guaranteed. The Wikipedia experiment was not so fortunate: no 'encyclopaedic ethic' existed at the outset, so when vandalism and harassment started to emerge—revealing that not all anonymous contributors could merely be assumed to be trustworthy—the community hurriedly embarked on a campaign to educate actual and potential Wikipedians. In so far as contributors come to feel bound by this emerging 'Wikiquette', they can surely be trusted to further the encyclopaedic cause.

In general, inference of trustworthiness preferably relies on solid signs from the particular trustee involved. Open-source communities, however, depend on anonymous strangers who usually do not reliably signal anything and are merely represented by the IP addresses assigned to them. Such communities can therefore only try to gauge the existence of a climate of trust among their potential contributors in general and infer in a *weak* sense that they probably can be trusted. Due to the veil cast by the Internet, open-source communities are condemned to forming probability estimates of trustworthiness in place of certainties. Note, though, that another, more robust option for inferring trust is becoming available. Lately, within OSS circles the hacker ethic is seen to be eroding. In response some projects subject role occupants to stricter screening, especially when granting commit privileges is at stake. Solid proof of one's allegiance to the hacker ethic is required. 'Strong' inference of trust, therefore, may replace 'weak' inference of trust.

As long as an appropriate kind of ethic seems to be lacking within the community, assuming trust is the prominent mechanism, by default. Wikipedia has been shown to be a case in point. Posting encyclopaedic entries on the Internet as a wiki is an appeal to fellow Wikipedians to show their editing capabilities. It is a sign of 'substantial' trust (in the sense of McGeer 2008). This theorizing about the 'normative pressure' emanating from opening up content would seem to confirm the conjecture that, especially in cyberspace, assuming trust is an important mechanism for creating trust in the first place (cf. de Laat 2005). In this vein both Wikipedia and diaristic blogs revealing personal intimacies (as analyzed in de Laat 2008) seem to rely boldly on the mechanism that trust can be produced ex post.

Sooner or later open-source communities start to introduce rules and regulations to manage the complexities involved. The governance tools distinguished above are modularization, formalization, division of roles and decision making. While the first two parameters would seem to meet with universal approval among participants, the last two need to be introduced with care. An important choice has to do with the amount of discretion granted to collaborators. In a high-discretion design, role division is minimal and decision making decentralized; as a result, the trust granted remains high. A low-discretion design, with elaborate division of roles and centralized decision making, leaves little discretion to collaborators; as a result, trust is to a large extent substituted.

This third mechanism of substitution of trust was first explored with reference to OSS projects. A typical division of roles consists of contributors, committers and maintainers (or module owners). I have shown how FreeBSD tends towards a higher discretion design, while Mozilla tends towards a design of lower discretion. Similarly, the encyclopaedic Wikipedia project has become the subject of design. The fact is that their contributors retain a broad measure of discretion: anonymous users may still edit entries of their choice (and may create new entries after registration). Rising vandalism has not been combated by reducing the discretion of ordinary users, but by introducing 'administrators' who are granted powers to block particular users and freeze articles involved in edit wars. As a result, users stay at the helm in Wikipedia.

Nevertheless, rampant vandalism has led to mounting pressures within Wikipedia for another measure: review of all changes (edits) with regard to vandalism. This system is undoubtedly a step that will reduce the discretion of ordinary users. It leads to the introduction of levels of

trustworthiness, measured by one's past performance within Wikipedia. Interestingly, this is not uniformly appreciated across language versions. It has already been introduced in the German Wikipedia (and some others), where the majority applauds the rules as a contribution to the fight against vandalism. Their English-speaking fellow Wikipedians, however, vehemently resist its introduction as an encroachment on their editing rights. To them, this is the onset of bureaucratic control. This finding suggests that intercultural perceptions of open-source design may differ considerably. This would seem to be an interesting research field, ripe for exploration: how essentially the same virtual organization is perceived, evaluated and designed across different cultural domains.

As a logical extension, plans are afoot within Wikipedia to appoint 'super-reviewers' who are to check articles for their quality. If and when such a system of review plus super-review may become standard practice, the user will no longer reign supreme in crafting Wikipedian entries. Such a development would imply two remarkable processes of 'structural convergence': on the one hand, Wales' encyclopaedia would then be governed in a similar fashion to comparable—but smaller—virtual encyclopaedic undertakings like Citizendium, h2g2 and Knol. These projects also apply review procedures to guarantee quality. On the other hand, generally speaking, roles within online open encyclopaedic projects would then resemble those that are usually discerned in OSS. The open-source communities for producing encyclopaedias on the one hand and software on the other would be managed in similar ways. The communities involved would seem to be concurring on the verdict that open-content production cannot do without a process of moderation. For all of them, unquestioning trust in users has proved to be an unworkable assumption.

References

Websites mentioned in text and/or references were last visited on 17 May 2010.

Adler, P. S., & Borys, B. (1996). Two types of bureaucracy: Enabling and coercive. *Administrative Science Quarterly, 41*(1), 61–89.

Andrews, C. K., Lair, C. D., & Landry, B. (2005). The labor process in software startups: Production on a virtual assembly line? In R. Barrett (Ed.), *Management, labour process and software development* (pp. 45–76). London/New York: Routledge.

Baier, A. (1986). Trust and antitrust. *Ethics, 96*(2), 231–260.

Baier, A. C. (1992). Trusting people. *Philosophical Perspectives, 6*, 137–153.

Benkler, Y. (2006). *The wealth of networks: How social production transforms markets and freedom*. New Haven/London: Yale University Press.

Coleman, E. G., & Hill, B. (2005). The social production of ethics in Debian & free software communities: Anthropological lessons for vocational ethics. In S. Koch (Ed.), *Free/open source software development* (pp. 273–295). Hershey: Idea Group.

De Laat, P. B. (2005). Trusting virtual trust. *Ethics and Information Technology, 7*, 167–180.

De Laat, P. B. (2007). Governance of open source software: State of the art. *Journal of Management and Governance, 11*(2), 165–177.

De Laat, P. B. (2008). Online diaries: Reflections on trust, privacy, and exhibitionism. *Ethics and Information Technology, 10*, 57–69.

Fox, A. (1974). *Beyond contract: Work, power and trust relations*. London: Faber and Faber.

Gambetta, D. (1988). Can we trust trust? In D. Gambetta (Ed.), *Trust: Making and breaking cooperative relations* (pp. 213–237). Oxford: Blackwell.

Granovetter, M. (1985). Economic action and social structure: The problem of embeddedness. *American Journal of Sociology, 91*(3), 481–510.

Himanen, P. (2001). *The hacker ethic and the spirit of the information age*. London: Vintage.

Holck, J., & Jørgensen, N. (2005). Do not check in on red: Control meets anarchy in two open source projects. In S. Koch (Ed.), *Free/open source software development* (pp. 1–26). Hershey: Idea Group.

Jørgensen, N. (2001). Putting it all in the trunk: Incremental software development in the FreeBSD open source project. *Information Systems Journal, 11*, 321–336.

Lane, M. S., Van der Vyver, G. L., Basnet, P. & Howard, S. (2004). Interpretative insights into interpersonal trust and effectiveness of virtual communities of open source software (OSS) developers. *15th Annual Australasian Conference on Information Systems (ACIS)*, Hobart, Tasmania. Retrieved from http://aisel.aisnet.org/acis2004/93.

Levy, S. (1984). *Hackers: Heroes of the computer revolution*. London/New York: Penguin.

Luhmann, N. (1968). *Vertrauen: Ein Mechanismus der Reduktion sozialer Komplexität* (4th ed.). Stuttgart: Lucius & Lucius, 2000 (N. Luhmann, Trans., 1979. Trust and power. Chichester: John Wiley).

McGeer, V. (2008). Trust, hope and empowerment. *Australasian Journal of Philosophy, 86*(2), 237–254.

Mizrach, S. (1997). Is there a hacker ethic for 90s hackers? Retrieved from http://www.fiu.edu/~mizrachs/hackethic.html.

Pettit, P. (1995). The cunning of trust. *Philosophy & Public Affairs, 24*(3), 202–225.

Pettit, P. (2004). Trust, reliance and the Internet. *Analyse und Kritik, 26*, 108–121.

Robbins, J. (2005). Adopting open source software engineering (OSSE) practices by adopting OSSE Tools. In J. Feller, B. Fitzgerald, S. A. Hissam, & K. R. Lakhani (Eds.), *Perspectives on free and open source software* (pp. 245–264). Cambridge, MA: The MIT Press.

Sitkin, S. B., & Roth, N. L. (1993). Explaining the limited effectiveness of legalistic "remedies" for trust/distrust. *Organization Science, 4*, 367–392.

Viegas, F. B., Wattenberg, M., Kriss, J., & van Ham, F. (2007). Talk before you type: Coordination in Wikipedia. *Proceedings of the 40th Annual Hawaii International Conference on System Sciences*.

Zucker, L. G. (1986). Production of trust: Institutional sources of economic structure, 1840–1920. *Research in Organizational Behaviour, 8*, 53–111.