Karl-Dieter Crisman

Open Source Software and Christian Thought

Karl-Dieter Crisman

This article introduces the distinction between proprietary and open source software, and discusses connections with Christian thought. It argues that this distinction is important for Christians to understand, and covers four main resonance points between Christian thought and open source software: stewardship, building community, helping the underprivileged, and promoting creativity. It also addresses how the approaches differ.

With ubiquitous computing power, the practice of science has changed dramatically—as has the range of ethical issues Christians must respond to concerning privacy, computing access, and whether to enhance human abilities.[1] This article surveys, and suggests, responses of the Christian faith tradition, in particular, to the idea of *open source software* (OSS). Open source refers to the way software is developed. It not only has implications for how we do science, but also has deep resonance with a number of core values in Christianity.

Especially with respect to creativity and freedom, the position taken here is that OSS better expresses a Christian approach. However, it is also not a panacea or a unique way to "write or use software Christianly," and such a survey cannot possibly be comprehensive. The intention here is to open the door to further discussion of the issues involved. As Redeemer University computer scientist Derek Schuurman says in his recent book, *Shaping a Digital World,* technology really is not value-neutral;[2] tools and methodologies Christians use in science have potential to reflect our beliefs as much as the questions we choose to address.

Open Source Software

The *source code* of a program is the original instructions to the computer, written by programmers. Usually this code is in a higher-level human-readable language such as C++, Java, and Python. By contrast, the software itself is usually a binary file—one only the computer can really read and interpret. This sets up a fundamental distinction:

> *Open source* software is software whose source code may be modified and redistributed. The source code of *proprietary* software may typically not be modified or redistributed without express consent.

We will delve later into further important distinctions, but the right to redistribute the source code is the most fundamental operating difference. One reason this is confusing is that, although nearly all OSS has no immediate acquisition cost, it is *not* the same as no-cost "freeware."

Consider the Firefox web browser or LibreOffice office suite;[3] these are OSS, but there are no-cost or loss-leader equivalents which are not. On the other end, many of us (whether we know it or not) use the Linux operating system kernel in embedded devices, in Android phones, or in company/university backends—in the latter case, with expensive service

Karl-Dieter Crisman *is an Associate Professor of Mathematics at Gordon College and a Faculty Fellow of the Center for Faith and Inquiry there. His research is in the mathematics of voting and choice, and he has taught students and faculty how to use and develop for Sage mathematics software (http://www.sagemath.org) on three continents.*

contracts. Similarly, the OSS Apache and nginx web servers, invisible to the end-user, dominate that market.[4] One may not even be aware that one's software is open source.

In science, the ideas of open access (for example, *PLOS*), open wikis, and open standards are more familiar, and we will spend some time on current technical science-related arguments regarding OSS shortly. But open source is only related to these, not identical to them, and to truly understand this (as well as the theology), a small amount of the history of OSS is necessary.

A Brief History

Many programmers of open source consider it (accurately) to be a movement, or even a philosophy. This view stems from a change in the role of programmers over the decades as software, not hardware, became the more marketable product. Political scientist Steven Weber characterizes this transition in his *Success of Open Source,* "The narrative of the programmer is … of the craftsperson from whom control and autonomy were taken away."[5]

Steven Levy's book *Hackers* tells this story (which is almost a mythology by now) in far more detail.[6] To oversimplify, it says that programmers from the 1950s to the 1970s, whether working in garages or on huge IBM mainframes, could be artisans who shared ideas and code, while today they are fungible resources. Whether this narrative is always true is less important than that it can feel true—that one may wish to see innovations built upon, not endlessly reinvented.

The mechanism asserting this control over a program is the *copyright license*. Standard intellectual property protocol for software grants the user a limited license to use copyrighted material. A free version may be restricted to nonprofit activities; some licenses restrict use outside the United States or prohibit use for certain activities, for example, creating weapons, and most explicitly require release from liability. Since the source code typically can be used to recreate the program, it is usually not included, and attempts at reverse-engineering are also usually prohibited by the license.

The key innovation by early open source developers—most famously in the case of Richard Stallman in the early 1980s—takes the idea of such licenses and inverts it, with a radical rethinking of intellectual property. *Open source licenses explicitly allow redistribution of source code.* There is no space to go into the full history of the various open source licenses, including the role that AT&T's phone monopoly played in it, but to really understand the motivations behind open source, any of the many books on this in the endnotes are well worth the read. Weber summarizes it well: "Property in open source is configured fundamentally around the right to distribute, not the right to exclude."[7]

It would take another article to examine properly why people write huge amounts of software under licenses that make it nearly impossible to charge for the software itself—not to mention how it often happens in a highly decentralized way, with corporate support coming after, not before, success. Some explanation will come up when we discuss community and creativity below, but for now we will follow open source guru Eric Raymond and others in stating that work on OSS, whether starting a project or working on a bug, comes from "scratching a developer's personal itch."[8]

One of the most pervasive licenses, originating with Stallman, does not even allow modifications to be distributed without allowing subsequent modification and distribution ("copyleft").[9] This means that not only can one modify the code (open source), but that it also can *never be directly used in a proprietary product.* Advocates of this stance often reject the term "open source," which dates to early 1998,[10] and use Stallman's original "free software,"[11] because copyleft licenses are to protect the freedom of the developer and all future users to do whatever they want with the software.

This is a controversial distinction. On the one hand, one might ask whether this sort of restrictive freedom is indeed free. On the other hand, one might ask whether allowing code that was initially open to become closed is in the spirit of freedom. There is a very healthy ongoing debate on this issue (including among the referees of this article!), and we have no intention of resolving it, on a Christian basis or otherwise. What is crucial for our purposes is that partisans of both perspectives are using the language of morality and freedom (in addition to other, more technical rationales) for at least some of their motivation for using such licenses; we will return to this several times.

Science and Open Source

Before considering theology and OSS, it is worth noting that there is a healthy discussion regarding OSS in scientific contexts, and it is not of purely academic interest. This will also help flesh out what OSS is.

First, OSS is everywhere in science. Well-used tools such as Biopython and Bioperl are sponsored by a foundation for open bioinformatics, while the R data analysis project is ubiquitous enough in biostatistics, psychology, and even finance that there are companies whose business model is entirely based on support or extensions for R.[12] The industry standard for technical document preparation in mathematics and physics continues to be programs based on LaTeX. In geography, the GRASS GIS has been under continuous development for decades, as has Octave in numerical calculation. In addition, there are hundreds of toolkits and files that researchers make freely available which run routines and scripts based upon standard proprietary tools such as MATLAB; such code is often OSS.

This illustrates a point commentators have made numerous times. In Weber's words, software is an "*antirival*" good.[13] Not only does the value of software not diminish if more people use it, including freeloaders (as opposed to the "tragedy of the commons"), but its value may also increase with additional users—for example, when they contribute bug reports or other suggestions. In domains with some programming expertise, OSS leverages this further with user-contributed fixes.

As science becomes more and more reliant upon computation (and scientists more knowledgeable in it), the value that most scientists receive from software is not the resale value of any software they use or develop, but the freedom to solve their own problems, whether in proteomics or optimization. Significant customizability and the ability to quickly fix problems in a program may even be a prime motivation behind starting a new project.

At the same time, because of the focus on use and not necessarily on development, the sciences will also ask hard questions about any new software. Questions of support arise—whether for support staff or for third-party/hardware support. Particularly in pedagogical contexts, the nature of the learning curve for the user interface and "sunk" investments in a program bear consideration.

We find that such questions accentuate the complexity of the issue; they do not provide a clear technical preference of one form. Some companies provide excellent support with their proprietary licenses and frequent updates; others may be sold to another firm which lets a product languish, or just go out of business with the customer at a loss. Similarly, some open source programs have robust ecosystems of online support and an easy transition from other programs, while others are known for challenging, developer-centered interfaces or lack of access to hardware.

However, scientific considerations, such as transparency and reproducibility of research, have gained in importance recently. Several editorials in scientific journals over the past five years, along with the explicit policies of some journals, have urged this. In particular, data and analytic code should be made available and usable for peer-reviewed work, while the analysis itself should be reproducible by "executing the code on the data provided and produc[ing] results matching those that the authors claim."[14]

Though this is possible using code built upon programs like MATLAB (as *PLOS*'s statement allows[15]), it does beg the question of whether to trust the results of programs whose source one cannot see, especially if the algorithm *is* the research, for instance, a new statistical test. On a more practical level, it is an incentive to release one's own code as OSS to enable further advances in science.

Those close to, or in, mathematics or computer science have additional issues to consider. For pedagogical reasons, students should often be able to verify and understand algorithms, and it is incredibly instructive to do so in the software they are actually using. Similarly, one might also want to know whether a correct algorithm has been correctly implemented.[16] Again, this is only possible with some way of viewing source, though not necessarily with rights to modify and redistribute.

Finally, considering how research has expanded beyond the domain of large universities in the West raises the question of who can afford to use certain types of software. Someone finishing doctoral work may move to an under-resourced college or return to a university in the developing world, where they may not be able to afford a license to use code developed for their dissertation. Differential pricing schemes may be part of the solution; open source has this potential as well.

Christian Thought and Open Source

In all three of these latter examples (transparency, pedagogy, and access), one senses the ethical imperative coming through. Open source users and creators have a history of explicit moral value judgments in their self-perception and motivation. So it should be no surprise that there are explicit connections to Christian thought.

There are four major areas of resonance. It is worth noting that they are quite ecumenical and form natural connections to many secular value systems as well. We will return to freedom after considering each of the following in turn:

- Stewardship
- Building community
- Helping the underprivileged
- Promoting creativity

The claim of this article is that, although none of these are guaranteed simply by using OSS (and many proprietary programs have potential in each of these areas), open source has more potential to live up to these virtues, and hence, in general, gives more opportunity to connect with these Christian ideas.

Stewardship

The most obvious attribute of OSS is its low acquisition cost, and for many Christian commentators this question is best subsumed into that of direct stewardship of monetary resources for a given organization. We will leave aside for the moment whether this is all that stewardship is, and ask whether using OSS is like the servant who doubled the ten talents.

As a representative of many who say yes, consider erstwhile Charlotte, North Carolina, house church leader Donald Parris's detailed documents on switching church functions such as accounting and scheduling to computers using Linux. He asks,

> Cost is not the only factor in your [software] decision, and not even necessarily the most important. Yet, if you could reasonably manage your church for less money than you currently do, wouldn't you want to explore that option further? You might be able to … boost a missionary's efforts …[17]

The vision is compelling.

With *any* software, however, support and training costs, hosting, and other aspects of "total cost of ownership" (TCO) must be considered. In a recent microeconomic analysis, the software acquisition cost portion of (self-reported) TCO by firms in countries of all development levels was shown to be only one of the determining factors when considering whether to use OSS.[18] (This study would have been more valuable with far more specifics about the products acquired.)

As an example more familiar to many readers, although OSS learning management systems such as Moodle have partners offering paid support,[19] the lack of a sole institutional support point (as with rival Blackboard), can be a key deciding factor when the first priority is 100% uptime. Further, with extremely cheap internet access and free cloud-based solutions such as Google Docs, the dynamic has changed even further.

On the other hand, one blog post by a church solutions firm goes so far as to suggest that only churches with a lot of tech-savvy members need use OSS[20]—this is simply false.[21] There are OSS solutions needing no special training, and proprietary programs needing a great deal of it; this is not a specifically OSS versus proprietary argument. And certainly for many small organizations, acquisition cost *is* the largest part of TCO.

This brief stewardship discussion should make it clear that, although one may make a good argument for a specific open source product on price alone, this is only a part of a Christian reflection on OSS.

Community

It may be a surprise that for many open source users, fostering community plays a central role. One extremely popular content management system advertises itself with the tagline, "Come for the software, stay for the community."[22] Stallman, of "free software" fame, does not really disagree with the characterization that he "built up an entire political movement to address [the] issue of … crushing loneliness."[23]

Users and developers of OSS share and grow ideas intentionally together, including in person, and often speak of this aspect. One referee spoke quite eloquently of a "real Acts 2 vibe" and companionship. This resonates with Christian themes: Israel is to be a microcosm of true community; Jesus calls the disciples his true family; and the epistles are full of descriptions of how to live in community.

However, open source community is not identical to Christian community. Some OSS communities can be exclusive or even hostile to those new to it, to members with a different licensing philosophy, or to those who transgress unwritten norms.[24] Some projects can at least be perceived as uniting against the common foe of a particular computer company. Research does not suggest that this is a primary motive, but it is *a* motive.[25]

Finally, as a worldview (not necessarily in personal interaction), OSS communities value people for contributions. Extensive research bears out that in a "typical" project, reputation gained by meaningful contributions means something.[26]

In order to analyze this, we must compare it to an alternative model. Companies such as Google and Apple also welcome limited-user community, for example, in the form of help forums, but without the cultivation of "every user as a potential volunteer"[27] and the prototype of servant leadership. Training sessions in open source communities nearly always are about building one another up,[28] not about selling the features in the latest upgrade.

Larry Wall, an evangelical Christian, is the founder of a popular open source programming language, Perl.[29] He has often spoken explicitly about this connection: "[I modeled] the Perl movement on another movement … the founder [of which said], 'He who wishes to be greatest among you must become the servant of all.'"[30] Indeed, because developing OSS is typically voluntary, there is a real subversion of traditional hierarchies. Project founders cannot be autocrats, but they must convince developers of the technical superiority of their ideas. Wall says,

> I began by talking about the virtues of a programmer: laziness, impatience, and hubris. These are virtues of passion. They are also virtues of an individual. They are not, however, virtues of community. The virtues of community sound like their opposites: diligence, patience, and humility.[31]

Freed from making a sale, the open source model can relate to these spiritual gifts. Asking someone to use OSS means trying to win a contributor, not trying to acquire a customer; Schuurman mentions "the notion of common grace" to explain this.[32]

Raymond explains why this is true for developers: "authority follow[s] responsibility" for any piece of code.[33] However, any *user* has this potential as well, for example, by contributing to a help forum, suggesting better translations, or trying new versions. Proprietary systems usually pick exclusive groups with nondisclosure agreements; in the open source world, such groups must be as open as possible for success.

Philosopher Pekka Himanen compares social bonds in open source to a Plato-like academy of "companions in learning," rather than to an authoritarian monastery.[34] Jesuit spokesman and theologian Antonio Spadaro rightly calls this particular analogy *"molto riduttivo"*;[35] still, it seems very similar to Jesus's teaching his disciples to see him not only as master, but also as a friend who has given them power of their own.

There is a further reason for Christians to be involved with open source communities. The primary means of communication is electronic—one of the most powerful, fast, and nuance-free forms there is. Hence, there is rampant potential for misinterpretation and rash words. Becoming *charitable* contributors to such communities is one of the most powerful places Christians can be witnesses in the digital age; because of the distributed nature of OSS development, they can rise to positions of real influence and respect, and cause others to do likewise.

Helping Others

One of the most compelling moral arguments made in favor of OSS is that it allows sharing of one's resources with others. This is a prominent theme of Glyn Moody's *Rebel Code*, a major history of the GNU/Linux operating system.[36] Parris and Stallman are both extremely vocal on this. For Parris, it is partly because software piracy and "convenient" copying is so prevalent inside the church. Perhaps ironically, Stallman, a staunch atheist, asks Christians to call proprietary software demonic, since, in his view, a putative Satan would like nothing better than to hook people on things that are fun "on condition that they refuse to share it with anyone else."[37]

(Scriptures such as Luke 6:34–35 probably more than superficially support this view.[38])

In general, there is debate over whether new technology is good because it promotes efficiency, or bad because it creates dependence. Jacques Ellul's views are well known: "Everything in the technological world is a means and only a means, while the ends have practically disappeared."[39] Nonetheless, within the open source community (both from Christian and secular perspectives), there is a consensus that working on OSS can indeed help others, particularly the underprivileged.

Marco Fioretti, the founder of an Italian group of Catholics supporting OSS,[40] quotes liberally from papal encyclicals and scripture about repairing divides between rich and poor nations, and Baylor professor Alan Jacobs refers to the open source community as the "cyber-Amish."[41] In a personal communication, the founder of one open source program says,

> I think Sage is a form of contribution in a social sense. Open source is the way some nerdy types can best voluntarily contribute to society … I think the really good people that are a pleasure to work with see the social contribution as something they greatly value.

Himanen cites the same idea, "a desire to rid the network society of the survival mentality that so easily results from its logic."[42] Wall concurs, "People really do help people for the sake of helping people."[43] This motivates those of no faith as well as those of faith.

This argument also holds water empirically. To be concrete, OSS properly organized—which is not a given—has a number of important advantages in granting opportunity toward those not in the digital elite. We give some examples from a development point of view, though other ways of helping others can be analyzed similarly.

From the pure acquisition cost perspective for its graduates, the African Institute of Mathematical Sciences, for example, has a blanket policy to use OSS in all its dealings.[44] The original vision of One Laptop Per Child was to use Linux for this same reason. Bandwidth is also a cost issue in many circumstances.[45]

Similarly, especially in the developing world, low-end hardware or old operating systems is a real issue.[46] Typically, it is not worth the effort for a normal software developer to continue providing versions of their software for such situations. With open source, the potential exists to keep things operating far longer than typically viable, although it is not a guarantee that it will happen. As one example, MATLAB is currently dropping support for Windows XP, which still has significant market share worldwide, while GNU Octave maintains it.[47]

Two closely related points should be considered: learning from OSS, and localization. A number of studies (including some cited above) show that, outside a context where English is an expected knowledge base, using or writing software without the language or visual cues appropriate to a culture is unlikely; however, those with the language/cultural skills have many opportunities. Because the source is available, one can learn programming paradigms (as already happens with web pages because of Tim Berners-Lee's insistence that one must be able to "view the source"); for the same reason, localization is truly possible even for smaller firms in the OSS context. A good example is an Italian thesaurus contributed to OpenOffice by a school in Bologna.[48] But localization is not a guarantee, and requires infrastructure, as in the ANLoc African Locales Initiative.[49]

On the academic side, economists have done analyses of accelerating knowledge transfer to the developing world using OSS[50] and have published case studies of whether and how it is used.[51] Josh Lerner and Mark Schankerman's analysis examines current practice and draws mixed conclusions, but reinforces the idea that, even with large amounts of "comingling" of code, both small firms and large firms benefit from using OSS in varied cultural contexts.[52] Weber makes a lengthy related argument with respect to exclusion from the digital economy and dependency theory, though his argument also connects to freedom.[53]

It may thus be a little surprising that, despite many OSS Christian software projects, such as for church databases, there are few examples of institutional religious activity in the open source world.[54] Columnist Bruce Byfield suggests several interesting ideas on why this might be, in a 2006 article in Linux .com.[55] Catholic commentators such as Fioretti have tried to interest their coreligionists in this, but it was only in 2011 that Spadaro, in his flagship Jesuit journal *La Civiltà Cattolica*, wrote a longer article about the "hacker ethic."[56] Because it bears upon the idea of helping others by giving, it is worth quoting one of his main observations about the "gifts" involved in open source activity: *"Non spinge a dare e ricevere, ma a prendere e lasciare che gli altri prendano,"* loosely translated, OSS is a "gift" that "does not incite to give or to receive, but to take and leave so that others may take."[57] This is not quite the same as the biblical idea of giving of oneself.

Spadaro makes compelling observations regarding the difference between the gift of God in salvation and the gift of contributions to software. This is important because it highlights the difficulty that nondevelopers might have to connect it to Christian practice. Namely, the personal and self-giving nature of such a relationship must slowly be acquired, and it is incumbent upon the user to activate it by becoming involved in a community. Why this might be truly important—and worth doing—brings our final connection to theology.

Creativity

One of the highest essences of God is that he is "Creator of Heaven and Earth." God is *creative*, and even early in the biblical creation account he asks Adam to be creative, naming the beasts. Without the creative Word, "nothing was made that has been made"; this lies deep within the tradition of *imago dei*.[58]

Allowing full reign for creativity is deeply connected to open source development—all commentators agree—and is central to the "hacker ethic" Himanen expounds. Interestingly, Himanen places this creativity in contrast with Christian ideas about the purpose of life in ways that appear not fully thought through;[59] Spadaro's essay rightly ends with an extended paean to enabling of the creative element, connecting it to "the dynamic element in the Church … belonging to the Holy Spirit."[60] Samir Chopra and Scott Dexter's (partisan, but nonetheless scholarly) work on OSS and philosophy, *Decoding Liberation,* has an entire chapter on aesthetics and free software, with many pages devoted to creativity and collaboration.[61]

This connection may be surprising to those unfamiliar with coding, but it is vital. Computer and mathematics guru Donald Knuth says, "I think people who write programs do have at least a glimmer of extra insight into the nature of God."[62] Another prominent Christian in the field (both are Turing Award winners), Frederick Brooks, claims early in his best-known work, "I think this delight must be an image of God's delight in making things."[63]

Spadaro quotes Levy quoting Christian programmer Tom Pittman about this as well,[64] but a quote that they do not use is even more insightful: "… perhaps I had learned something about God. In this I have a definite advantage over the painter and the composer: I can create something that will interact with me, as man interacts with God."[65]

Larry Wall states explicitly that he believes that promoting healthy creativity is tantamount to helping achieve God's potential for humans: "In my little way, I'm sneakily helping people understand a bit more about the sort of people God likes."[66] Elsewhere he asserts that God's creative control is subtle, so ours should be as well.[67] Schuurman spends several pages on the cultural mandate and the uncovering possibilities inherent in creation as part of computer science.[68]

The broader point implicit in all of these assertions is that *anyone* can engage in this creativity.[69] OSS, in this view, gives anyone the potential to be creative without having to start from scratch; rather, he or she can modify or build on existing code. Even a small documentation fix, or the act of reliably reproducing the conditions under which a bug does and does not occur, is seen to be part of the creative scientific process. Software professional Ken Shafer places such maintenance firmly in the locus of biblical stewardship of God-given gifts.[70] By way of analogy, one might think of glass artist Dale Chihuly, who has been physically unable to work on his oeuvre for decades, yet shepherds a whole pool of workers who bring ideas to fruition; each is participating in the creative impulse, though under the hierarchy of the master.

Calvin College education professor Ron Sjoerdsma provides the biblical narrative of Bezalel and Oholiab (Exodus 35) as a paradigm.[71] These artisans were not just gifted to craft the beautiful ornaments for the tabernacle of the Lord, but they were inspired to teach others as well. This analogy of the need for skilled workers in technology to be mentors at the same time, to teach others to contribute in a community effort, is evident.

We suggest that with this point of view, OSS becomes a good place to start empowering others to use their God-given creative gifts to benefit the whole community and to redeem our digital world, with even tiny contributions being worthwhile.

Freedom

And ye shall know the truth, and the truth shall make you free (John 8:32, KJV).

Although freedom arose early in our discussion, we now raise it in a Christian context. Before discussing what Christians can affirm in OSS discussions of freedom, it is important to discuss what they cannot.

Many OSS enthusiasts explicitly connect the success of open source to so-called libertarian ideals—that is, to various economic and social freedoms, free from oversight or compulsion, whether from government, business, or religion. The idea is that code freedom works and is only the first step to wider freedoms—some even using language labeled by other observers as having "Messianic overtones." Raymond is probably the most vocal exponent of this, and he skewered Spadaro's essay because of the incompatibility of such a viewpoint with hierarchical ideas in Catholic thought (which indeed is a substantial part of the essay).[72] Weber is far more cautious but also outlines a set of conditions under which distributed production and this kind of freedom might be plausibly invoked in other disciplines or areas of life.[73]

Attractive as this may be to some Christians, this kind of freedom is not a Christian freedom. Christ's "yoke is easy," but there is still a yoke, and Paul still calls himself a servant or slave of Christ.[74] Freedom is not the same thing as total liberty: some changes in computer "code" will break the software. This is not a straw man; even the most passionate advocates for such liberty, such as Raymond, agree in their writing that there are limits. Chopra and Dexter spend considerable time interacting with moral philosophers from Mill to Popper and beyond to justify the copyleft licenses in terms of rejecting the fullest liberty.[75] One point raised is the danger that complete liberty perpetuates preexisting inequities in social structures—turn this on its head, and connecting open source to "libertarianism" seems far less germane.

To think about freedom more productively, consider Orwellian ideas, predicated on the very real fact that others using technology can control us or see our thoughts and actions. The recent furor over the scope of the National Security Agency's data collection and security bugs in both open source and proprietary software are in front of our consciousness. The same is true about companies, especially as they move their software—and hence our creative work—to the nebulous "cloud."

In our view, the ideas of reproducible research can help here. More important than the openness of the software is openness of the process and the data. In a word, *control of information is the key*. This kind of freedom is consonant with Christian thought; we do not want to be dependent on any human, but rather on Christ alone, to have freedom to worship, to spread the Gospel, and so forth.

Open source better enables this sort of freedom via *open standards,* a form of *commoditization*.[76] In a major anthology of open source musings from the mid-2000s,[77] article after article discussed how items from car tires to cement to toilet paper rolls are not locked in with proprietary formats,[78] but are standardized to enable more innovation building on those technologies. In this view, standardized protocols and file formats are necessary to enable flourishing in the digital realm. Even Lerner and Schankerman's advocacy "of neutrality between OSS and proprietary … [is] underpinned by a very strong commitment to … standards whose specifications can be used and implemented by all who desire to do so."[79] Work computations, home budgets, and homework should not be dependent on a proprietary format.

Whole countries and constituencies have adopted (or have threatened to adopt) requirements that documents be prepared in global standards for this very reason.[80] The web, as we know it, would not exist without the standards of HTML and CSS. Even now, archivists are finding that depending on any one company's existence for keeping file formats available risks losing part of our heritage.

Open standards allow for anyone to access their data with any appropriate tool, regardless of location or economic circumstance, as do tools available to all. Weber asserts that "certain bodies of code are essential … in the same way that pens and paper were."[81] Chopra and Dexter devote much space to human-technology interfaces,[82] and to whether we will have control of the code in use in such interfaces; perhaps grandiosely, they view open source as allowing Ellul's prisoners of technology to unlock their own prison.

OSS cannot completely solve the issue of locking patrons into an ecosystem. But open source enables community-driven standards to be implemented and improved upon. This is a sense of freedom Christians can stand behind. It is not just freedom to distribute; it is freedom from potential cyber-bondage. This may sound dramatic, but we are still in the infancy of data defining us, and we little know what directions the future may take.

Epilogue

Many thanks are due. The Sage math software community introduced me to OSS, and Gordon College has been a hospitable place to try new things with it, including examining connections to faith. A Center for Faith and Inquiry fellowship enabled me to present these thoughts in workshops at the conference of the Association of Christians in the Mathematical Sciences and at the youth ministry conference of Open Boston. Finally, the two referees made many helpful comments from two very different viewpoints on the issue.

Our digital future is unclear; facial recognition technology and "big data" mining would have seemed completely futuristic just a decade or two ago, and likely will soon seem as commonplace as email. OSS is a big part of that future, and this is surely not the last Christian word on it.

This article restricts itself to connections between open source and Christian thought—not necessarily suggesting implications for theology, though an ill-defined "open source theology" has already appeared (type this phrase into your browser and see what you get). Likewise, the "software as a service" model represented most prominently by social networks and online applications "in the cloud" needs discussion from a Christian standpoint, as does the question of how open source is paid for.

What is already certain is that many Christians practicing in this sphere see a deep synergy with our faith. Open source is a fascinating and subversive paradigm; hopefully this article will help to spark the conversation at a higher and more visible level than before. ♦

Notes

[1] As an example, see controversy over Google glasses or Ray Kurzweil's embrace of "the Singularity" in Kurzweil, *The Singularity Is Near: When Humans Transcend Biology* (New York: Viking, 2005).

[2] Derek Schuurman, *Shaping a Digital World: Faith, Culture, and Computer Technology* (Downers Grove, IL: InterVarsity, 2013). His earlier article on the same topic, "Forming a Christian View of Computer Technology," *Journal of the Association of Christians in the Mathematical Sciences* (December 2007), is available freely at http://www.acmsonline.org/journal/2007/schuurman.pdf.

[3] It is hard to escape the irony of the fact that *PSCF* required use of a particular proprietary program for this article on open source software.

[4] One of the most respected surveys is Netcraft's, e.g., http://news.netcraft.com/archives/2014/08/27/august-2014-web-server-survey.html.

[5] Steven Weber, *The Success of Open Source* (Cambridge, MA: Harvard University Press, 2004), 25.

[6] Steven Levy, *Hackers: Heroes of the Computer Revolution,* 25th anniversary ed. (Sebastopol, CA: O'Reilly Media, 2010).

[7] Weber, *The Success of Open Source,* 1.

[8] Eric S. Raymond, *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*, rev. ed. (Sebastopol, CA: O'Reilly Media, 2001), 19–64.

[9] The GNU General Public License, available at http://www.gnu.org/copyleft/gpl.html.

[10] And the decision by Netscape to license its browser under an open source license. See, e.g., Raymond, "Revenge of the Hackers" in *The Cathedral and the Bazaar*, 167–92.

[11] This inevitably requires the clarification that this means "free as in freedom, not free as in free beer." Sometimes the adjective "Libre" is used to maintain this distinction, and the acronym FLOSS is especially popular in Europe.

[12] Such as http://www.revolutionanalytics.com/ or http://www.mango-solutions.com.

[13] Weber, *The Success of Open Source,* 154.

[14] A recommendation from Roger Peng, "Reproducible Research and *Biostatistics*," *Biostatistics* 10, no. 3, (2009): 405–8. Similar editorials in *Science* and *British Medical Journal* have appeared in 2011 and 2012. Also related are David Joyner and William Stein, "Opinion: Open Source Mathematical Software," *Notices of the American Mathematical Society* 54, no. 10 (November 2007): 1279; and Darrel C. Ince, Leslie Hatton, and John Graham-Cumming, "The Case for Open Computer Programs," *Nature* 482, no. 7386 (2012): 485–88.

[15] "PLOS Editorial and Publishing Policies," section 7, http://www.plosone.org/static/policies#sharing.

[16] I am indebted to my colleague Russ Bjork for this point.

[17] Donald C. Parris, *Penguin in the Pew: Bringing GNU/Linux into the Church 2.0* (Raleigh, NC: Lulu Press, 2007), 43.

[18] Josh Lerner and Mark Schankerman, "The Demand Side: Assessing Trade-offs and Making Choices," in *The Comingled Code: Open Source and Economic Development* (Cambridge, MA: MIT Press, 2010), 103–56.

[19] See http://moodle.com/partners/.

[20] *Elexio Blog*, "Is Open Source Church Software a Good Cost-Saving Option?," blog entry by Ken Stewart, June 18, 2013, http://blog.elexio.com/blog/bid/305336/is-open-source-church-software-a-good-cost-saving.

[21] In personal communication, a colleague confirms that the Mormon (LDS) faith provides computers with OpenOffice

for standard congregational use. See also *Asay: Offline and Off-topic,* "It's Official: Open Source Is Righteous," blog entry by Matt Asay, December 7, 2005, http://asay.blogspot.com/2005/12/its-official-open-source-is-righteous.html.

[22] *Drupal – Open Source CMS*, https://drupal.org/.

[23] Sam Williams, *Free as in Freedom: Richard Stallman's Crusade for Free Software* (Sebastopol, CA: O'Reilly & Associates, 2002), 198.

[24] A very famous example of this in the Linux OS kernel is described in detail in Glyn Moody, "Low-Down in the Valley," in *Rebel Code: Inside Linux and the Open Source Revolution* (Cambridge, MA: Perseus, 2001), 162–81.

[25] Tux, the penguin mascot of Linux, can often be seen stomping on various Microsoft products and in posters on office doors of computer experts.

[26] In various venues, such as Andrea Bonaccorsi and Cristina Rossi Lamastra, "Comparing Motivations of Individual Programmers and Firms to Take Part in the Open Source Movement: From Community to Business," *Knowledge, Technology, and Policy* 18, no. 4 (2006): 40–64; and Sandeep Krishnamurthy, "On the Intrinsic and Extrinsic Motivation of Free/Libre/Open Source (FLOSS) Developers," *Knowledge, Technology, and Policy* 18, no. 4 (2006): 17–39. One good general source is Joseph Feller, Brian Fitzgerald, Scott A. Hissam, and Karim R. Lakhani, eds., *Perspectives on Free and Open Source Software* (Cambridge, MA: MIT Press, 2005).

[27] Karl Fogel, *Producing Open Source Software: How to Run a Successful Free Software Project* (Sebastopol, CA: O'Reilly, 2005), 132–33.

[28] This was a very common comment made in training sessions that I ran for the program I work with.

[29] Originally named for the "pearl of great price."

[30] Larry Wall, "2nd State of the Onion," talk given at the Perl Conference 2.0, August 1998, http://www.wall.org/~larry/onion/onion.html.

[31] Ibid.

[32] Schuurman, *Shaping a Digital World*, 98.

[33] Raymond, "Homesteading the Noosphere," in *The Cathedral and the Bazaar*, 19–64.

[34] Pekka Himanen, *The Hacker Ethic and the Spirit of the Information Age* (New York: Random House, 2001).

[35] Antonio Spadaro, S.J., "Etica hacker e visione Cristiana," *La Civiltà Cattolica* (March 2011).

[36] Moody, *Rebel Code*.

[37] Marco Fioretti, "Christian Endorsement of Free Software Increases," *website of linux.com*, April 16, 2006, http://archive09.linux.com/articles/53644.

[38] Thanks to Craig Story for this reference, one of many relevant ones.

[39] Jacques Ellul, *Perspectives on Our Age: Jacques Ellul Speaks on His Life and Work*, ed. Willem H. Vanderburg (Berkeley, CA: Publishers Group West, 1981), 50.

[40] Elèutheros, http://www.eleutheros.org/en/.

[41] Alan Jacobs, "Life Among the Cyber-Amish," *Books and Culture* (July/August 2002).

[42] Himanen, *The Hacker Ethic and the Spirit of the Information Age*, 141.

[43] Wall, "2nd State of the Onion."

[44] See http://www.aims.ac.za/en/apply/structured-masters-course-proposals.

[45] As of 2013, various sources report millions of American households still using dial-up services.

[46] A very interesting example of this is Caroline Hardin, "FOSS in School Communities: An Experience Report from Peace Corps Volunteers in Ghana," in *Free and Open Source Software and Technology for Sustainable Development,* ed. Sulayman K. Sowe, Govindan Parayil, and Atsushi Sunami (Tokyo: United Nations University Press, 2012), 286–312.

[47] See http://www.mathworks.com/support/sysreq/roadmap.html; they are dropping support for Windows XP and Mac OS X Lion, which is a full decade younger than XP, at the same time.

[48] Yu-Wei Lin and Enrico Zini, "Free/Libre Open Source Software Implementation in Schools: Evidence from the Field and Implications for the Future," *Computers and Education* 50, no. 3 (April 2008): 1092–102.

[49] Martin Benjamin, "Language Data as a Foundation for Developing Countries: The ANLoc 100 African Locales Initiative," in *Free and Open Source Software and Technology for Sustainable Development,* ed. Sowe, Parayil, and Sunami, 164–82.

[50] Jamil Alkhatib, Mohab Anis, and Hamid Noori, "Open Source: The Next Big Thing in Technology Transfer to Developing Nations," *International Association for Management of Technology Proceedings*, 2008.

[51] Such as Imed Hammouda, "The Open Source Ecosystem in Tunisia: An Empirical Study," in *Free and Open Source Software and Technology for Sustainable Development,* ed. Sowe, Parayil, and Sunami, 185–211.

[52] Lerner and Schankerman, *The Comingled Code,* 61–102.

[53] Weber, *The Success of Open Source,* 248–54.

[54] SIL (the parent of Wycliffe Bible Translators) enabled development of a universal typesetting program, and several sources report some activity in the LDS.

[55] Bruce Byfield, "Why FOSS Isn't on Activist Agendas," *website of linux.com*, December 13, 2006, http://archive09.linux.com/feature/58726.

[56] Spadaro, "Etica hacker e vision Cristiana."

[57] I am grateful to my colleague Damon DiMauro for assistance with an idiomatic translation.

[58] See Schuurman, *Shaping a Digital World,* 36–38.

[59] Himanen, *The Hacker Ethic and the Spirit of the Information Age*, 13–19.

[60] Spadaro, "Etica hacker e vision Cristiana."

[61] Samir Chopra and Scott D. Dexter, *Decoding Liberation: The Promise of Free and Open Source Software* (New York: Routledge, 2008).

[62] Donald E. Knuth, *Things a Computer Scientist Rarely Talks About* (Stanford, CA: Center for the Study of Language and Information, 2001).

[63] Frederick P. Brooks Jr., *The Mythical Man-Month: Essays on Software Engineering* (Boston: Addison-Wesley, 1995), 7. Interestingly, although Brooks's writing (technical and otherwise) is infused with scriptural references and his faith, his comments on open source and Eric S. Raymond's views in "What Are Better Design Process Models?," in *The Design of Design: Essays from a Computer Scientist* (Boston: Addison-Wesley, 2010), 51–60, do not seem to make any such connections.

[64] Spadaro, "Etica hacker e vision Cristiana."

[65] In his online essay, Tom Pittman, "Deus ex Machina, or the True Computerist," http://www.ittybittycomputers.com/Essays/DeusExMac.htm.

[66] "Larry Wall on Perl, Religion, and …," on the website *Slashdot,* September 6, 2002, http://interviews.slashdot.org/article.pl?sid=02/09/06/1343222.

[67] Wall, "2nd State of the Onion."

[68] Schuurman, *Shaping a Digital World,* 32–36.

[69] In one British study, teachers at schools who had opted to simply use open source experienced significant gains in self-perceived comfort with technology, compared to colleagues using nearly identical proprietary software. See "Open Source Software in Schools," technical report, British Educational Communications and Technology Agency, 2005.

[70] Ken Shafer, "Software Development and Maintenance as Stewardship," *Dynamic Link: Christian Perspectives on Software Development*, Issue 1 (2007–8): 6–7.

[71] Ron Sjoerdsma, "Teachers, New Technology—the Bezalel Principle," *Christian Educator's Journal* 38, no. 1 (October 1998): 4–5.

[72] Eric Raymond, "Imprimatur me!" *Armed and Dangerous* (blog), http://esr.ibiblio.org/?p=3094. Apparently Raymond replied before fully reading the original article, based on the very long set of comments on his post about it; the last comment has Raymond halfway through, and no indication of any further public dialogue between them.

[73] Weber, *The Success of Open Source,* 264–72.

[74] Matthew 11:30; Romans 1:1.

[75] Chopra and Dexter, *Decoding Liberation*.

[76] For a good historiographic perspective on this topic, including an interesting evaluation of modern open standards as a "consensus of critiques," see (esp. the conclusion of) Andrew L. Russell, *Open Standards and the Digital Age: History, Ideology, and Networks* (New York: Cambridge University Press, 2014).

[77] Chris Di Bona, Mark Stone, and Danese Cooper, eds., *Open Sources 2.0: The Continuing Evolution* (Sebastopol, CA: O'Reilly, 2005). See also the original influential Chris Di Bona, Sam Ockman, and Mark Stone, eds., *Open Sources: Voices from the Open Source Revolution* (Sebastopol, CA: O'Reilly Media, 1999).

[78] There certainly have been such incompatible standards in the past, and not just Beta versus VHS. The plethora of weights and measures in medieval Europe is an oft quoted example of how lack of standardization can impede innovation.

[79] Lerner and Schankerman, "The Demand Side: Assessing Trade-offs and Making Choices," in *The Comingled Code,* 210.

[80] As only a representative sample, see "Sharing or Collaborating with Government Documents: Proposal," January 28, 2014, http://standards.data.gov.uk/proposal/sharing-collaborating-government-documents; Martin LaMonica, "Massachusetts to Adopt 'Open' Desktop," *CNet News,* September 1, 2005, http://news.cnet.com/Massachusetts-to-adopt-open-desktop/2100-1012_3-5845451.html; Michael Reed, "Munich Linux Migration Project LiMux Reports Success," *Linux Journal* (January 2, 2012), http://www.linuxjournal.com/content/limux-munich-linux-migration-project-reports-success; Anita Chan, "Coding Free Software, Coding Free States: Free Software Legislation and the Politics of Code in Peru," *Anthropological Quarterly* 77, no. 3 (Summer 2004). A comprehensive list from 2010 is James Andrew Lewis, "Government Open Source Policies," Center for Strategic and International Studies, http://csis.org/publication/government-open-source-policies.

[81] Weber, *The Success of Open Source,* 246.

[82] "Cyborgism," but referring to everything from glasses to prosthetics and beyond.