CHAPTER 2

Free Software as a Democratic Principle

Nic Suzor, Brian Fitzgerald, & Mark Perry

A. INTRODUCTION

Software forms an important part of the interface between citizens and their government. An increasing amount of government functions are being performed, controlled, or delivered electronically.[1] This software, like all language, is never value-neutral, but must, to some extent, reflect the values of the coder and proprietor.[2] The move that many governments are making towards e-governance,[3] and the increasing reliance that is being placed upon software in government, necessitates a rethinking of the relationships of power and control that are embodied in software.

It is important for software that directly influences administrative, legislative, and judicial decisions to be examinable by the citizenry, just as the decisions themselves are open and transparent. Without this safeguard, there can be no guarantee that the decisions made by government are legitimate. If the software that is relied upon to provide either the inputs or the

1 For example, see US, General Accounting Office, *Critical Infrastructure Protection: Challenges in Securing Control Systems* (GAO-04-140T) (2003), online: www.gao.gov/new.items/d04140t.pdf.

2 Brian Fitzgerald, "Software as Discourse: The Power of Intellectual Property in Digital Architecture" (2000) 18 Cardozo Arts & Ent. L.J. 337.

3 For a discussion of e-governance, see: Thomas Riley, "E-Government vs. E-Governance: Examining the Differences in a Changing Public Sector Climate" Commonwealth Centre for E-Governance (2003), online: www.rileyis.com/publications/research_papers/tracking03/IntlTrackRptMay03n04.pdf.

evaluation in a decision-making process cannot be trusted, there can be no trust in either the process or the final decision.[4]

A responsible government must be accountable for the decisions it makes, for and on the behalf of its people. Transparency is one of the most important factors of a representative democracy.[5] The entire system rests upon the proposition that if the government is not acting in a way that is supported by the majority of the people, then the people have the ability and the obligation to force a change in the government. If the people do not have the power to understand what the government is doing, and the basis upon which it is making decisions, there can be no question of an informed democratic process.

When the processes of a government are embodied in software, it is critical that the inner workings of that software be available for examination, discussion, and critique. Without access to the human, readable source code, it is practically impossible to be sure exactly how a piece of software functions, to assure oneself that it is not biased against or for any particular individuals or sectors of society, or to ensure that there are no critical bugs or vulnerabilities that may enable others to exploit the software for their own personal benefit. The investigation of these issues in code is no less important than the investigation of bias and corruption in government offices by traditional reporters.

This chapter will argue that use of free software in governments is an ideal way to foster trust and informed discussion in a democratic society.[6] Free software provides its users with the ability to examine and understand the source code of key software applications. The use of free software would increase the transparency of government decisions, and thus also the degree to which citizens can trust those decisions, or work for change in the decision-making process.

---

4 Rana Tassabehji & Tony Elliman, "Generating Citizen Trust in E-Government Using a Trust Verification Agent: A Research Note" (Paper presented to the European and Mediterranean Conference on Information Systems, Costa Blanca, Alicante, Spain, 6–7 July 2006), online: www.iseing.org/emcis/EMCIS2006/Proceedings/Contributions/EGISE/eGISE4.pdf.

5 See Onuora Awunor, "Transparency in Government" (29 July 2007), online: http://onuoraawunor.blogspot.com/2007/07/transparency-in-government.html.

6 For a look at how free software expresses democratic ideals of citizenship engaged with the affirmation of information of individual rights and freedom in North America and Brazil, see Alexandra Pinheiro & Henrique Cukierman, "Free Software: Some Brazilian Translations," online: www.csi.ensmp.fr/WebCSI/4S/download_paper/download_paper.php?paper=pinheiro_cukierman.pdf.

It is important to note that free software is not a strict requirement for transparency. While all free software is, by definition, transparent to the user, source code can also be made available for examination of non-free software.[7] There are, however, significant benefits to the use of free software in governments beyond transparency. The four freedoms encompassed in the term "free software" include: [8]

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

While access to the source code of software is strictly necessary for transparency in governments, the other freedoms guaranteed by free software are also beneficial in the public sector. The use of free software allows for citizens to become involved in their governance by participating in the software-development process.[9] Free software also ensures that the benefits of publicly funded development and acquisition are returned to the public. In this chapter we will examine each of these propositions, and argue that there is a compelling case for the use of free software in governments.

B. "JUSTICE MUST BE SEEN TO BE DONE"

One of the most important aspects of democracy is that government must be accountable to its citizens. Decisions made by the executive, judicial, and legislative arms of government must, as far as possible, be transparent. Without transparency in the decision-making process, there can be no trust that a government is acting on behalf of its people. Accountability in government is a continual prerequisite for a functional democracy. The processes through

---

7 See, for example, Microsoft Shared Source, online: www.microsoft.com/resources/sharedsource/default.mspx.

8 Free Software Foundation, "The Free Software Definition," online: www.gnu.org/philosophy/free-sw.html.

9 For discussion of the political rights and participation of the citizenry in the information society, see: Hans Klein, "The Right to Political Participation and the Information Society" (Paper presented to the Global Democracy Conference, Montreal, 29 May–1 June 2005), online: www.ip3.gatech.edu/research/Right_to_Political_Participation.pdf.

which decisions are made in government must be both accountable and transparent, and the workings of critical software systems employed by the government to aid in that decision-making process are no exception.

1) Electronic Voting Machines

One of the most important instances of accountability in a democratic society is that of the voting process. Citizens need to be able to trust that the voting process is not either (a) being deliberately manipulated, or (b) significantly flawed and susceptible to error. Without trust in the process through which our representatives are elected, there must be significant doubts as to the legitimacy of a democratic government.

Many governments are moving towards using electronic voting machines (EVMs) to assist in the voting process. While there are certainly concerns about electronic voting as a whole,[10] there are significant advantages that make it attractive to governments. Electronic voting makes accurate vote counting nearly instantaneous, eliminating many hours of tedious, error-prone work.[11] Computer assisted interfaces can be easily adapted to assist voters who have difficulty in reading and marking a paper ballot — the information can be presented and recorded in a number of ways to account for the individual needs of the voter.[12]

If electronic voting systems are going to be used, they must be able to be trusted. A key component of this trust is the availability of the source code that controls electronic voting software.[13]

Without being able to study and understand the source code of EVM software, citizens must rely on the assurances of the vendor or government

---

10 Security problems abound. See Dave Jefferson *et al.*, "A Security Analysis of the Secure Electronic Registration and Voting Experiment (SERVE)" (20 January 2004), online: http://servesecurityreport.org/. There are also significant issues with implementation, such as in the 13th District November 2006 elections where there were 18,000 "no vote for candidate" votes on ES&S machines. See Kim Zetter, "Did Florida Foul Another Ballot?" (17 November 2006), online: www.wired.com/politics/security/news/2006/11/72130?currentPage=all.

11 For a look at some of the issues surrounding electronic voting, see: Institute of Governmental Studies, University of California, "Electronic Voting: Overview and Issues" (November 2005), online: http://igs.berkeley.edu/library/htElectronicVoting2004.html.

12 For example, the Victoria Electoral Commission recently introduced electronic voting for people with vision impairment, online: www.vec.vic.gov.au/electronicvoting.html.

13 See, for example, Jason Kitcat, "Why Electronic Voting Software Should Be Free Software" (1 March 2001), online: www.inmyarea.org/h/n/WRITING/evoting/ALL/53/.

that there are no bugs (or "features") in the software that may affect the result of a vote.[14] In a representative democracy, however, these assurances are simply not good enough. The extent to which an accident or a malicious actor can influence an electronic vote far exceeds that of a paper-based ballot. Vote tampering in a paper system would involve substituting or removing a very large amount of ballots. In an electronic system, on the other hand, a few lines of code could substantially, and nearly imperceptibly, alter the results of a whole election.[15]

If electronic voting is to be used and trusted, it is clear that the source code must be able to be scrutinized.[16] The guarantees of a private company providing the electronic equipment are not sufficient in a field as critical to the practice of democracy as voting.

While it is obvious that not every voter will have the skills or time required to evaluate the trustworthiness of any given EVM software, public accessibility increases the probability that malicious or insecure code will be identified. This review function could conceivably be performed by a core group of independent auditors, but without public review there are no checks placed upon the evaluation of those auditors. The availability of the code for public review, even if the code is not actually reviewed by a large portion of the public, engenders trust in the review process.[17]

---

14 For an overview of the electronic voting process, see: Tadayoshi Kohno *et al.*, *Analysis of an Electronic Voting System: Proceedings of the IEEE Symposium on Security and Privacy, Berkeley, 2004* (Los Alamitos, CA: IEEE Computer Society Press, 2004).

15 See, for example, a report that ES&S machines are vulnerable to numerous exploits using a magnet and a PDA with an infrared port, potentially allowing an attacker to "exercise complete control over the results reported by the entire county election system" with no more access to the system than that typically provided to voters. See *EVEREST: Evaluation and Validation of Election-Related Equipment, Standards and Testing, Final Report* (December 2007), online: www.sos.state.oh.us/sos/upload/everest/00-SecretarysEVERESTExecutiveReport.pdf at 53, quoted in Kim Zetter, "Report: Magnet and PDA Sufficient to Change Votes on Voting Machine" (17 December 2007), online: http://blog.wired.com/27bstroke6/2007/12/report-magnet-a.html.

16 Additionally, in a discussion about the role of source code disclosure and transparency of voting systems, Joseph Lorenzo Hall concludes that disclosure of full system source code will promote technical improvements in voting systems: Joseph Lorenzo Hall, *Tranparency and Access to Source Code in Electronic Voting* (2006), online: http://josephhall.org/papers/jhall_evt06.pdf.

17 For a more in-depth discussion of the dangers of closed-source EVM systems, see Mark Perry & Brian Fitzgerald, "FLOSS as Democratic Principle: Free Software as

There are a number of reasons for which manufacturers of EVM systems are reluctant to make their code available to the public. The first, termed "security through obscurity," rests upon the assumption that it is undesirable to provide would-be attackers with a blueprint of the software, which could be used to make exploitation easier. Unfortunately, relying on the secrecy of the code is a very brittle security mechanism. Once an exploit has been found by a malicious attacker, it remains open until an attack is detected, analyzed, and the breach repaired. The better approach is to design a system that is secure even if everything about the system is public knowledge. This is known as Kerckhoffs' principle, named after Dr. Auguste Kerckhoffs, a nineteenth-century cryptographer. Kerckhoffs' principle was extended by Eric Raymond, in relation to open source software, who claims that "[a]ny security software design that doesn't assume the enemy possesses the source code is already untrustworthy."[18]

The argument follows that open source software can be more secure because it allows weaknesses to be found through widespread continuous testing, and does not rely on the secrecy of potential vulnerabilities.[19] On this basis, Bruce Schneier argues that in the long term, "public scrutiny is the only reliable way to improve security."[20]

Another argument against public disclosure rests upon the commercial exploitation of electronic voting software. Providers of EVMs argue that the code which runs their systems provides their competitive advantage over other providers. These providers claim protection of their code not only as copyright works, but also as trade secrets, and are reluctant to allow any public access that could lessen their competitive advantage.[21] This argument simply cannot be accepted. If it is true that public access to the

---

Democratic Principle" (2006) 2(3) The International Journal of Technology, Knowledge & Society 155, online: http://eprints.qut.edu.au/archive/00004425.

18 Eric S. Raymond, "If Cisco Ignored Kerckhoffs's Law, Users Will Pay the Price" (17 May 2004), online: http://lwn.net/Articles/85958.

19 See Bill Caelli, "Security with Free and Open Source Software" in Brian Fitzgerald & Graham Bassett, eds., *Legal Issues Relating to Free and Open Source Software* (Brisbane: Queensland University of Technology, 2003). See, further, Peter Swire, "A Theory of Disclosure for Security and Competitive Reasons: Open Source, Proprietary Software, and Government Systems" (2006) 42 Hous. L. Rev. 1333.

20 Bruce Schneier, "Internet Shield: Secrecy and Security" *San Francisco Chronicle* (2 March 2003), online: www.schneier.com/essay-033.html.

21 See, generally, Andrew Massey, "'But We Have To Protect Our Source!': How Electronic Voting Companies' Proprietary Code Ruins Elections" (2004) 27 Hastings Comm.& Ent. L.J. 233.

source code is a requirement for public trust in the voting system, then the interests of a private provider cannot be put above the needs of the public. If it is not possible for a private developer to create an EVM solution that adheres to the standard of trust needed for democratic elections, then the development must be carried out by the government. The integrity of voting infrastructure is not something that can be compromised for the benefit of private actors.

One other criticism of publicly disclosed source code rests in the proposition that there is no assurance that the code examined is the code that is actually running on the EVM systems on the date of the election. Examination in this case may not provide any greater level of security. While this criticism has some merit, it goes more deeply to a criticism of EVM itself, not of disclosure. There is much more to software security than the integrity of the source code. If the machines in question cannot be reasonably secured from interference, then they should not be used. If no electronic machine can provide a satisfactory level of security, then electronic voting should not be seen as a satisfactory replacement for paper-based ballots. If, on the other hand, electronic voting machines are used, then the code that is assumed to be running must be available for scrutiny.

The argument in this section does not rely on making a recommendation for whether or not electronic voting systems should be deployed. Put simply, if a government does decide to implement EVM, among the large number of other security concerns that must be considered, the source code of the machines must be available for public scrutiny. The use of free software is one, but not the only, way in which the source code may be made available.[22]

2) E-governance, Online Delivery, and Electronic Control

The basis for preferring publicly accessible source code is not limited to electronic voting. As more core government services move towards online delivery, and more of citizen interaction with government becomes mediated through software, there exists a greater need for that software to be accountable to the citizenry.

There is a very real possibility that the software that controls access to government services may contain errors or omissions that make it more diffi-

---

22 Open source solutions are available for development; see, for example, Halina Kaminski & Mark Perry, "Verifiable Electronic Voting System: An Open Source Solution," online: www.csd.uwo.ca/~markp/htmls/vev.pdf.

cult for individuals, or groups of individuals, to access those services. Whatever the software, be it related to tax collection, welfare, medical funds, or education, individuals can benefit from the ability of any (sufficiently trained and resourced) member of the public to examine the logic behind the software, and to expose and report on any hidden biases or flaws.

As more of the core functions of government are being fulfilled by the citizenry using electronic interfaces, with a gradual removal of human bureaucratic intervention, greater reliance is being placed on the efficacy and correctness of the electronic systems. Recovery from software errors can be a long and tedious process, and there is always a temptation to assume that the software is correct, and that the citizen-user must therefore be incorrect. It can be very difficult, especially when dealing with such important matters as taxation, health, welfare, and education, for a citizen to have these errors corrected before serious hardship is caused.

The ramifications of such errors are decidedly non-trivial. Faulty logic in a welfare system may result in a family missing vital income for a number of weeks; an incorrect calculation of tax rules may result in undetected increases in taxation liability. Faulty input validation or broken logic rules may prevent families from accessing healthcare benefits or rebates. The public availability of the source code for these systems is vital to offer some protection from hidden bias, bugs, and resulting inequity.

The use of free software in the public sector is also important to ensure that the public retains control of public resources and information. By guaranteeing the freedoms to use, modify, adapt, and share software, governments are able to avoid vendor lock-in and third-party control of important public information. For example, the use of free software guarantees the ability to switch back-end data storage formats without changing user interfaces, and without losing existing data. Without the ability to port user interface and data standards, governments and citizens face very high switching costs, both financially, and as a function of education and retraining.

3) The Public Should Be Involved in Their Own Governance

Most of the discussion so far has centred on the availability of source code for public infrastructure software. Free software, however, can provide more benefits to the public than just accountability. Using free software has the opportunity to allow individuals to participate in the meaning-making process, as well as working to help improve software to better suit individual needs.

One of the features of market-based software development is that only profitable features are implemented, which leaves many applications without features that would be very useful to a minority of users, but for which demand cannot be adequately manifested.[23] These types of features can extend from multilingual language support, to alternative input and output modes for people with disabilities, or to support for niche market or non-profit applications. Through the use of free software, individuals or groups desiring such features can "scratch an itch," and add code (or employ someone to add code) for which they can see a benefit. This iterative production lowers the barrier to entry to production of niche features, and hence increases the utility of software to minority groups.

Even where demand for new features can be manifested, it is often not a simple process for a vendor to provide customized additions to existing software. After the initial roll-out of a system, a government may be tied to the original vendor for upgrades, and the process of negotiating and implementing new features can be extremely difficult. The inability to make changes independently means that feature requests must be made in accordance with extremely complicated contractual relationships. As the needs of government and citizens change and evolve over time, being tied to the original vendor, either contractually or through copyright, may become a very onerous limitation. The use of free software eliminates any strict tie to a particular vendor, and greatly reduces the barriers for the implementation of new features.[24]

To the extent that a functioning democracy needs to keep its software up to date with the ever-changing needs of its citizens, free software can be immensely beneficial in reducing the role of the vendor as a bar to the updating process. Free software has a direct advantage in that free software vendors may of course still provide support and additional development, but in those cases where vendors are not willing or able to do so, there are much smaller barriers to obtaining the same services from other vendors.[25] Free

---

23 See Jay P. Kesan & Rajiv C. Shah, "Deconstructing Code" (2004) 6 Yale Journal of Law & Technology 277 at 378–82.

24 For other benefits of free software, see Jason Kitcat, "Why Electronic Voting Software Should be Free Software" (9 July 2001), online: www.kuro5hin.org/?op=displaystory;sid=2001/7/9/81531/16879.

25 See Brian Fitzgerald & Nicolas Suzor, "Legal Issues for the Use of Free and Open Source Software in Government" (2005) 29(2) Melbourne U.L. Rev. 412.

software also gives citizens an opportunity to participate in the development of the software that controls their interaction with their government.[26]

4) Returns of Public Investment Should Benefit the Public

When a government develops or pays for the development of software, there are three broad options for the future of that software. It could remain locked up within the government, and never be released to the public. It could be commodified and commercialized and sold as a closed product in order to recoup the costs of development and perhaps become profitable. Alternatively, the government could release the software as free software, in order to return the benefits of the publicly funded development to the community.[27]

The advantages of this last option are numerous. The benefit to the public may be, in many cases, much greater when the availability of high-quality software commons is increased, rather than through the sale of a limited number of licences. Where exactly this line will be drawn will of course depend on the software, the market for the software, and its utility in potential derivative applications if it were made freely available. In assessing the balance, however, governments should be very clear that the benefits of a software commons increase exponentially with the amount of software that is made available, as the software can be used in any number of unexpected ways in combination with other available software.

There is a very significant advantage in allowing the private sector to repurpose high-quality government software for its own use. In many cases, this potential downstream increase in innovation will outweigh the immediate benefit of a limited number of software licences. The release of government-produced software as free software means that investment in software infrastructure has not only immediate payoffs for the purpose for which it was designed, but also long term benefits to sustained innovation

---

26 Eben Moglen notes that the choices we make in structuring copyright licences are of fundamental importance in structuring the community which develops around use of that material: see, for example, Eben Moglen, transcript of lecture given at the 3rd International GPLv3 Conference, Barcelona, 22 June 2006, online: www.fsfeurope.org/projects/gplv3/barcelona-moglen-transcript.en.html.

27 It is important to note that releasing government-developed software as free software does not remove the ability to commercialize the software, for example, by providing custom support or integration services, and realizing other benefits such as improved code, nor, indeed, from securing intellectual property rights.

in the private sector, where the code may be reused for seemingly unrelated purposes.

C. CONCLUSION: A CASE FOR FREE SOFTWARE IN GOVERNMENT

In a democratic society, there is a requirement that government resources are used in the interests and for the benefit of the public. As governments move towards adopting electronic approaches to automate vital functions and interface with their citizens, the use of free software ought to be carefully considered. The benefits of using free software include:

- increased accountability, in that citizens are able to examine the basis on which decisions are made;
- necessarily open standards, which extend data longevity and access far beyond that of any given technology or format;
- lower barriers to customization of software for minority groups and niche markets;
- the value of investment in software development is returned to the public for future reuse—allowing public, private, and non-profit organizations access to high quality software from which they can innovate and generate further value;
- less waste involved in the continual redeveloping of standard architectures; and
- citizens are given the opportunity to participate in the meaning-making process by directly soliciting inputs to the code that increasingly governs their lives.

The use of FLOSS in governments is growing, particularly in Europe. To date the adoption has primarily been for back-end operating systems (e.g., Apache and Linux) and common desktop applications (OpenOffice). There are policy pressures in the EU for adoption of FLOSS that range from community directives to municipal mandates.[28] Unfortunately, such moves have yet to take root in North America or Australia. We believe, however,

---

28 There is good data on EU government adoption, see Free/Libre/Open Source Software: Policy Support (FLOSSPOLS), online: http://flosspols.org and UNU-MERIT, *Study on the Economic Impact of Open Source Software on Innovation and the Competitiveness of the Information and Communication Technologies (ICT) Sector in the EU*, Final Report (20 November 2006), online: ec.europa.eu/enterprise/ict/policy/doc/2006-11-20-flossimpact.pdf [UNU-MERIT].

that there exists a strong argument that democratic governments ought to prefer free software to closed source software, in the best interests of their citizenry. It is critical, in our minds, that publicly funded software development, like publicly funded research, be conducted as much as possible in an open manner, with the results being shared with the community and made available for future innovation. In the future, we hope to see governments acting in accordance with this democratic principle by requiring vendors of software they acquire to release the developed code under free software licenses.

In each case where governments are procuring software development, the cost-benefit analysis should include a detailed examination of the lower development costs of free software and the public benefit of releasing the software, weighed against any increased costs caused by loss of the developer's ability to extract monopoly rents on closed-source applications.[29] To date, this analysis has largely ignored the public benefit arguments, and governments have focused on the fear expressed by private software development companies. We are hopeful that with recognition of the public benefits attainable through the use of free software, governments will be able to make more informed software procurement choices that will provide not only greater returns on investment, but also greater trust and participation in the democratic process.

---

29 See, for example, UNU-MERIT, *ibid.* at 11, which concluded that "FLOSS potentially saves industry over 36% in software R&D investment that can result in increased profits or be more usefully spent in further innovation."