Benjamin Ducke

7 Free and Open Source Software in Commercial and Academic Archaeology: Sustainable Investments and Reproducible Research

7.1 Introduction

While computing and software-based research pervade every aspect of modern archaeology, the pool of software specifically created for archaeological applications remains small. This is no small paradox, given that we live in a world of digital plenty; a world where software for just about any purpose has been available gratis, in the form of “Free and Open Source Software” (F/OSS), for decades. In key areas such as GIS, database management and statistical computing, there are free alternatives that are every bit as capable as even the most expensive paid-for offerings (see the expositions of F/OSS GIS by Neteler and Mitasova 2008; Sherman 2008; Rey 2009). And if some open source software doesn’t quite do the trick, then it can easily be modified and enhanced, thanks to the full disclosure of its source code. After all, the effort required to learn a programming language and start creating customised software is minute, compared to what archaeologists commonly invest in learning spoken languages, Latin, Old Greek, or even basic statistics and calculus. In the Age of the Internet, all that it takes is to go online. The tools, the knowledge and the support are all there, freely available to everyone around the globe.

Why then, do university departments, research projects and individuals still pay the considerable license fees asked for by vendors of proprietary, closed-source software? Why do they not take that same money and invest it into the digital assets of their own discipline, into staff training, transferable programming skills and tailor-made software that’s free to share with colleagues? And why, after all that is known and published about the prevalence of errors in complex programs (e.g. Hatton 1997; Merali 2010), do they still entrust their valuable research data to software that allows no access to its source code, that cannot be peer-reviewed and thus effectively acts as a black box in data processing (Morin et al., 2012)? There are many factors that must be taken into consideration when trying to find answers to these questions. But a lack of “quality” or other presumed shortcomings of F/OSS itself can be safely excluded. Rather, what seems to be holding back wide-spread endorsement of F/OSS in archaeology is a mixture of die-hard habits, as well as a lack of awareness of the intricate links

Benjamin Ducke: Freien Universität, Berlin, Germany

© 2015 Benjamin Ducke

This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivs 3.0 License.

between economic and academic aspects of software, and between technological and social capital in the open source community.

The present chapter addresses these issues and discusses pathways to successful investments in archaeological F/OSS; although due to space restrictions, this will have to be done in a brief and somewhat superficial manner (see Fogel 2009 for an excellent, much more detailed account of how to run a successful F/OSS project). It will open with a discussion of the economic aspects of F/OSS, before it will go on to discuss social and academic aspects. This order reflects the author's personal experience, that sustainable research, software-based or otherwise, requires a sound funding model. At the same time, good academic software practice cannot be sustained without a strong focus on transparency and reproducibility of research. Given the complexity of modern software applications, the ability to track data through complex processing chains is key to successful collaborative research, as is the ability to peer-review the software (e.g. Barnes 2010; Morin et al. 2012). Therefore, this chapter will discuss in some detail how the academic quality and usefulness of data-based research can benefit from the use of F/OSS.

Verifiable information on the successes or failures of F/OSS in different fields of application remains difficult to obtain. The detailed report by Wheeler (2005) provides some numbers on the key issues of market share, reliability, performance, scalability, security, and total cost of ownership. In addition, this chapter cites a number of other sources that do exist in the form of peer-reviewed, printed publications in English, originating from diverse academic disciplines. Three real-world case studies serve to supply some empirical evidence for the validity of the statements and the usefulness of the recommendations made in this chapter. The author has been directly involved in all three projects selected as case studies. Although this adds substantial prejudice to the discussion, it is hoped that the reader will nonetheless benefit from the insights and experience available to someone who has been a long-time, active participant in several F/OSS projects. The innovation cycles of archaeological computing are naturally connected with the much bigger dynamics of the bigger IT industry and business world. Therefore, this chapter will occasionally broaden its perspective to a more global view, whenever that helps to understand the current role of F/OSS in archaeology. However, for all practical details and as far as the real-world case studies are concerned, it will focus on applications within archaeology.

Advocates of Free and Open Source Software like to emphasise that the “Free” stands for much more than gratis computer programs, namely everyone's liberty (the term “libre software” is being used with increasing frequency) to use, modify and also redistribute software. This property of F/OSS clearly sets it apart from “freeware” (which is a more limited give-away) and “shareware (which is usually an even more limited give-away that serves as an incentive to purchase the fully functional version). Neither of these terms overlaps completely with what is known as the “public domain”. Works in the public domain do not have any form of copyright protection; F/OSS, however, does, and its authors do not give up their copyright privileges by de-

fault. Instead, they distribute their software under a modified version of traditional copyright (cf. McGowan 2005), often referred to as “copyleft” (Stallman, 2002, p. 127–134) and specified in detail by a license agreement such as the popular GNU General Public License (GPL; <https://gnu.org/licenses/gpl.html>; see also Stallman 2002, p. 165–203; Fogel 2009, p. 162–175). Typical open source license terms are very liberal, but some limits remain (see Rosen 2005 for comprehensive coverage of different open source licenses). Some software, for example, may only be used freely for academic and non-commercial purposes.

In fact, this kind of dual-licensing, that attempts to generate revenue from those licensees that “can afford paying”, is not uncommon; and it shows that F/OSS can support various business models. It is therefore somewhat misleading to speak of “commercial software” as the antithesis of F/OSS. Better terms for non-F/OSS are “closed source” or “proprietary” software. Avoiding to think of F/OSS as non-commercial also helps avoiding another misconception, namely that F/OSS is primarily created within an altruistic sphere of students, hobbyists and enthusiasts. In fact, most F/OSS programmers are professionals that contribute open source code, because they get paid, or because they have other, “selfish” reasons (see Klemens 2005, p. 96; Ghosh 2005). These clarifications and distinctions are critical for understanding both the economic and academic properties of F/OSS.

7.2 Selected Aspects of F/OSS

The following discussion points will sometimes make strong presumptions. Adequate evidence for their validity will only be given later, in the context of the real-world case studies. At this point, the intention is merely to provide some background to the most important aspects stressed within this chapter. The selection of aspects exposed is very reductive and does not attempt to do justice to the full complexity of the open source phenomenon. In particular, nothing will be said about the technological merits (or shortcomings) of specific F/OSS versus specific proprietary software. This would be a futile undertaking, given the pace of software evolution, and the widely diverging user skills and expectations. More extensive coverage, also on the roots of the open source movement, is provided by the well-known works of Raymond (2001) and Stallman (2002), with the latter striking a more ideological tone. For those capable of reading German, the book by Grassmuck (2004) provides another free and detailed source of information.

7.2.1 Open Source Economics

No academic software culture can prosper in the absence of continuous investment. The mere fact that a thriving F/OSS community exists in this world proves that the

idea of free software is fully compatible with a competitive capitalist economy. The availability of free, highly effective solutions for code sharing and online team work allows global collaboration with unprecedented intensity. That a modern business model does not necessarily involve proprietary, in-house programming and selling of licenses, is indeed one of the most important lessons to be learned from successful F/OSS projects (cf. Fogel 2009, p. 75–88; Lerner and Tirole 2004; von Hippel 2005, p. 265–346). Presumably, the underlying causes for the continuing lack of tailor-made, archaeological software solutions are to be found in a combination of the discipline's little economic significance, short-lived and project-based funding cycles, and a habitual acceptance of the “pay-per-license” model, rather than a lack of interest among its scholars and practitioners. F/OSS, on the other hand, lends itself to demand-driven and pooled funding models that offer clear and real advantages in an environment where individual decision makers have little financial resources at their disposal.

After all, the costs for a fully equipped workstation with licenses for CAD, GIS, DTP, and perhaps some software for statistical analysis, can easily approach the price of a new luxury car. Although this may not be an issue for members of the academic community, who are often fenced against such exorbitant cost by “campus license” agreements (for which the general tax payer more often than not picks up the bill instead), it is a critical problem for the much larger group of archaeological practitioners “out there”, who work for public services or for commercial companies, or are self-employed. It comes as no big surprise then, that many software users are attracted to open source solutions first and foremost because they view them as a way to save money. However, this strong “zero cost” attractor can lead to a certain conundrum in the context of a long-term economic strategy. Firstly, while it may be true that many open source alternatives to proprietary software exist, not all of them can be considered direct, “drop-in” replacements. Complex software such as GIS requires extensive user training and often employs undisclosed data formats or patented technology that cannot be used by open source competitors. This can mean that users will not be able to migrate away from proprietary software without a very costly overhead for data conversion; an effect that is known as “vendor lock-in”, and that will play a prominent role in the case studies to be discussed. Secondly, the development of F/OSS requires resources, just like any other form of software development. For larger projects, the resource requirements can be immense. As an example, the open source GIS gvSIG CE (to be discussed in detail later) consists of roughly 1.5 million lines of program source code. This represents programming work worth many millions of Euros. Clearly, endeavours of this magnitude require continuous investment (be it in the form of money or dedicated work time), carried by a broad base of users and supporters.

Sooner or later, any ambitious F/OSS project must therefore look for funding to sustain itself. Save for the discussion of selected aspects exposed via the case studies, this chapter will not get into the details of different F/OSS-based revenue models, as this would require a separate treatise (such as Krishnamurthy 2005). Generally speaking, the fact that common sources of income, mostly license fees, are not available to

non-proprietary software projects means that traditional business models also do not apply. This can be a challenge, since it requires F/OSS advocates to craft tailor-made revenue strategies and argue for their feasibility when looking for financial support from traditional-minded investors and public agencies. On the other hand, this need for creativity has opened up a plethora of new opportunities. At a time, when traditional software vendors are struggling to convince their clients of the “benefits” of ever more restrictive, top-down “cloud” and subscription-based models, F/OSS offers attractive alternatives, based on bottom-up, community-driven software development, that promise drastically increased flexibility and return on investment.

Besides the technological output in the form of software, the greatest return on investment in F/OSS comes in the form of social capital. Direct involvement in the decision making process, design and programming of complex software allows for transfer and sharing of skills and knowledge in a way that surpasses what can be gained from buying and using proprietary, closed-source software. As will be discussed later, this is of critical importance when it comes to the use of software in academic and research environments. At this point, the important thing to note is that there is an intimate link between the social and technological health of any F/OSS project: Due to the fact that social capital plays such a central role, a thriving and content user community is of utmost importance. More often than not, open source contributors are faced with the challenge of not just contributing money or program code, but also making sure that their contributions fit into the social fabric of the project.

7.2.2 Social Dynamics of F/OSS

An open source project is, by its very nature, averse to clearly marked ownership or predefined and rigid hierarchies. This makes the social dynamics, especially those of larger projects, all the more important (Ghosh, 2005; Lakhani and Wolf, 2005). Human emotions are constantly at work behind the scenes of F/OSS development, and negative ones are vented, often immediately and unfiltered, through public communication channels, such as mailing lists. Although this specific form of openness does have the advantage of preventing conflicts from simmering, it also stresses the need for good communication skills (Fogel, 2009, p. 98–117). In the absence of professional mediators, a small misunderstanding that may be quickly forgotten in “offline” communication, may stay on record and cause friction for a very long time on the Internet.

Bad social climate will result in the loss of precious social capital. And thus an open source project that does not achieve a feeling of equality and belonging among the members of its community, is destined to fail in the long run, no matter how generous the funding or how highly developed the contributors’ technological skills may be. On the other hand, a project that can build and sustain a loyal community will be able to weather even the toughest of economic times. As an example, the open source GRASS GIS (<http://grass.osgeo.org>; Neteler and Mitsova 2008) has been in the public

domain for three decades. This is a tremendous achievement that is on par with the longest running proprietary enterprises and far exceeds the meagre average of three to five years for a funded research project. It is all the more impressive, given that the project has had to find new backers and investors multiple times throughout its long history.

To achieve such longevity, an open source project must assign the highest priority to accessibility, transparency and inclusiveness in its decision making processes. Intriguingly, these same attributes are also highly desirable in academic research. They are the reason why, as will be elaborated later, F/OSS is so conducive to good scientific practice. What all this actually means for managing an F/OSS project, is something that needs to be learned by doing. Much of today's web technology is geared towards flat-hierarchy interaction, and any aspiring open source project should make ample use of collaborative tools, such as mailing lists, online forums and wikis.

One important thing that needs to be kept in mind, however, is that a web presence, as immaterial as it may be, still amounts to a form of "territorial claim" in social terms. In order to attract external collaborators, such a claim should therefore be made on behalf of the entire community, not a single project partner. If, for example, the "University of A" initiates an F/OSS project hosted at "<http://university-of-a/our-great-project>", then potential collaborators from the "University of B" will instinctively be much less inclined to join up, no matter how benevolent the initial idea might have been. This should serve as an example, learned through painful experience, of just how fleeting social capital in a highly interactive environment such as the Internet can be.

As a consequence, it is generally easier and more effective to join an existing project and share its resources, than to start from scratch. At the time of writing this, the North American service provider Sourceforge.net alone was hosting more than 200,000 active open source projects (<http://sourceforge.net>). There is a good chance that one of them will at least provide part of whatever software solution an archaeologist might be looking for. Attaching oneself to a larger, long-lived project does not only allow immediate access to greater technological and social capital. It will also ensure that investments will remain effective for much longer than the lifetime of the average academic funding cycle or budget plan.

7.3 F/OSS in Research

Archaeological research is increasingly based on computational methods and software. In fact, it would be difficult to name any aspect of the discipline that remains completely devoid of computer technology today. These developments have profound economical and epistemological implications that must be critically reviewed. This is particularly true for financial barriers to reproducible research and data processing opacity, both of which are unavoidable effects of the use of proprietary software.

Therefore, whereas in a commercial context open source may be one suitable option among several, in an academic context it is really without alternative (see also Ince et al. 2012). Proprietary software acts like a “black box” in research (Morin et al., 2012, p. 159) and thus stands at odds with good scientific practice. Firstly, it prevents researchers and students from ever fully understanding every detail of the data processing. Secondly, it transgresses against what is arguably the most fundamental requirement of science, that of reproducibility (see below). Given the same data and methods, any number of independent researchers should be able to recreate each other’s studies, verify or falsify them and improve upon them (see also Chambers 2008, p. 6–7). This is especially critical in archaeology, a discipline that has no established concept of proof, by which to assess the validity of specific methods, hypotheses or conclusions.

F/OSS presents itself as an obvious alternative; one that is better aligned with both the economic constraints of project-based research and the demands of good scientific practice. So far, however, little independent research has been published that speaks clearly about these aspects and their relevance to archaeological research. There is some urgency in rectifying this situation. As long as the rapid evolution of software is not being accompanied by increased scrutiny regarding the transparency of computational methods and data processing, the opacity of software-based research will increase and become ever more problematic (see Dücke 2012 for a current, albeit partial view on this problem).

7.3.1 Publish (Your Source Code) or Perish!

“Academic computer science has an odd relationship with software: Publishing papers about software is considered a distinctly stronger contribution than publishing the software. The historical reasons for this paradox no longer apply, but their legacy remains.” (Hafer and Kirkpatrick, 2009, p. 126)

Those archaeologists with an interest in the theory and application of computational methods will have been confronted with the statement that computers are “nothing more than tools”. Tools, of course, are designed to fit some relatively primitive purpose, and are thus simply not important enough to bother with such mundane concepts as computational transparency. This opinion, still commonly held in academic archaeology, is a crass underestimation of the role that mathematics and computing play in modern research. After all, computers are the technological manifestations of mathematical reasoning, an advanced aspect of human intellect, and computing is applied mathematics. Thus, the connection between software and the mathematical methods used to model, understand and ultimately solve real-world research problems is very explicit. Computer programs, that frequently run into the millions of lines of program code, are perhaps already humanity’s most comprehensive, certainly its fastest growing repository of formalised knowledge.

An illustrative example from archaeology is the publication on Bronze Age trade networks by Knappett et al. (2008). The part of this study that has been published in paper form amounts to a mere introduction to the research theme. The associated Internet resources (<http://theory.ic.ac.uk/time/networks/arch>) are somewhat more informative. But for those interested in the details of the mathematical models and formal reasoning behind the study, the source code of the software is the only place that can provide detailed insights. As another example, numerical simulation using agent-based modelling (ABM) has made a considerable impact in archaeology in recent years (e.g. Kandler et al. 2012). ABM is representative of a modern computing approach that is entirely unsuitable for traditional publication. There is simply no way in which printed static screen images or textual descriptions could adequately convey the impressions gained by interactively modifying and observing a dynamic ABM simulation. And it is not possible to understand what an ABM is doing, unless there is full disclosure of the program code behind the simulation. It is therefore no coincidence that proprietary solutions are all but absent from ABM research (<http://www.openabm.org>; see also Janssen et al. 2008). For similar reasons, the F/OSS project R (<http://r-project.org>) has grown into an extensive repository of applied statistical research, marginalising the scientific role of proprietary offerings in areas such as spatial analysis, that are of central importance to archaeological research (Bivand et al. 2013; Chambers 2008).

As software-based research becomes ever more pertinent in archaeology, the effectiveness of traditional forms of publication must be doubted wherever mathematical models and computing are concerned. In this respect, the prevailing *modus operandi* (not just in archaeology) leaves much to be desired. Indeed, withholding the source code from the academic community amounts to withholding the very means which enable others to peer review and learn from publicly funded research (Morin et al., 2012):

“Despite increasing reliance on computing in every domain of scientific endeavor, the computer source code critical to understanding and evaluating computer programs is commonly withheld, effectively rendering these programs ‘black boxes’ in the research work flow.”

However, the scope of this “black-box problem” is not limited to the aspect of withheld knowledge. The use of proprietary software in academic research, even for the most routine tasks, is an inherently flawed approach to scientific practice and scrutiny. This will become more obvious when thinking about the critical concept of reproducibility of software-based research.

7.3.2 Reproducible Research

“[...] the results of scientific calculations involving significant amounts of software should be treated with the same measure of disbelief as an unconfirmed physical experiment.” (Hatton, 1997)

Modern archaeological research employs complex software, most notably GIS, that provides hundreds of data processing functions. Bearing in mind the real limits of software testing and quality assurance, it cannot be assumed that each one of them is free of errors and will always produce the expected results. In addition, even the most basic numerical methods (algorithms) exist in a variety of implementations; be it because closed source code forces programmers to re-invent the same algorithm multiple times, or because perceived shortcomings call for modified and improved versions. As a consequence, not even the most basic operations, such as a simple line-of-sight analysis in GIS, can always be accurately reproduced across different software platforms (Ducke, 2012, fig. 1).

However, if it is impossible to read the actual program code that works on the data, it is also impossible to account for unexpected results. In addition, as everybody knows all too well from personal experience, any given software has a significant number of errors (cf. Merali 2010), and any problems encountered in the use of software may be due to one of these “bugs” as much as to flawed input data or other factors. Therefore, full publication of the source code has been a demand in science for some time (e.g. Buckheit and Donoho 1995; Donoho et al. 2009; Barnes 2010).

There are few means to assess the quality of archaeological research objectively. Reproducibility is one obvious criterion, as it seems indisputable that only reproducible results can be built on and verified or (more importantly) falsified by independent peers. It seems therefore imperative to uphold the ideal of fully reproducible research and to not sacrifice it needlessly, certainly not for the sake of convenience or for the mere habit of using proprietary software. After all, even if closed source software could square its own circle and function in a flawless and fully transparent manner, its cost would impose another, equally significant, limitation on the reproducibility of research.

7.3.3 Data-Centric Research

Nothing has been said so far about the relations between F/OSS and the second main resource of computing, the data. For obvious reasons, denying others access to research data ultimately leads to the same problems regarding transparency and reproducibility, as denying others access to documentation or program source code. This has been recognised in archaeology, as the Journal of Open Archaeology Data (<http://openarchaeologydata.metajnl.com/>) and related endeavours demonstrate (see also Kansa 2012). Although this chapter cannot discuss the interrelated, and without

a doubt important, aspects of “open data” and “open access”, it should be noted that F/OSS provides the best technological solutions for making data open and accessible, particularly in the long term, to the general public or at least the academic world. The fact that the Internet basically runs on open source software speaks volumes in this respect.

Good research requires discipline, self-regulation, and constant questioning of one’s chosen methods and preconceptions. In the context of software-based research, one of the most dangerous traps into which one might fall is that of “application-centric” thinking. In this all-too-seductive mode of thinking, the researcher allows the available capabilities of specific, reassuringly familiar, software to determine the analytical methods. The best safeguard against letting the software impose limits on the potential outcome of the research, is to acquire a habit of “data-centric” thinking instead.

As the name suggests, a data-centric approach firmly places the data into the centre of the research workflow. This approach recognises that the data represents the most valuable and irreplaceable investment, and that the choice of application software is a concern of secondary importance. After all, given the diversity of today’s software offerings, there is always an alternative. Modern data management technology supports this perspective by allowing users to create shared data infrastructures, accessible through standardised interfaces and protocols. End-user software can then be attached to a central data repository, allowing access to the data on different levels, via multiple user interfaces. With regard to GIS and spatial data infrastructures, a robust technical criterion for the inclusion or exclusion of software is how well it supports the standard protocols and data formats specified by the independent Open Geospatial Consortium (<http://www.opengeospatial.org/>).

Besides facilitating the ideas of data sharing and transparent processing, open and standards-based data infrastructures also alleviate the risk of data getting locked into undisclosed proprietary formats and thereby enable long-term data storage, archiving and accessibility. From an educational and academic point of view, the data-centric approach is attractive because it favours broader, transferable skills over narrower, application-specific skills. From an economical point of view, it opens up a broader range of investment options and more finely grained control over software spending. It is generally more cost-efficient to modify or even create software that integrates well into an existing, open infrastructure, than to license all the components required for a complete proprietary infrastructure. This is certainly the case for spatial data infrastructures and GIS, for which complete open source solutions exist (cf. Sherman 2008).

7.4 Case Studies

Having discussed some economic, social and academic aspects in a rather abstract manner, it is now time to let the proverbial rubber hit the road and take a look at the potentials and challenges of F/OSS in the real world. The following three case studies provide inside looks at attempts to include F/OSS as a central component in the development and usage of software to support archaeological applications and workflows. All of them demonstrate an involvement in the theory and practice of open source that goes far beyond the simple “gratis software” approach.

The discussion of each case study includes some background information (most importantly answering the question why F/OSS was chosen to play a central role), a description of the main challenges and the means employed to address them, and a short analysis of why critical goals were achieved or not achieved. Note that these short exposes cannot provide complete accounts (in fact, all of the cases discussed here are ongoing projects), but they may still provide valuable lessons that reinforce the advice given so far and help the reader devise good F/OSS strategies.

7.4.1 Oxford Archaeology Digital: F/OSS Migration in the Workplace

Oxford Archaeology (OA) is one of the world’s largest providers of archaeological services (<http://thehumanjourney.net>). Commercial archaeological practices such as OA have become indispensable pillars of heritage management in the UK, providing employment for thousands of archaeologists across the country and evolving in the contact zone of archaeology, landscape conservation and the construction industry. This environment places the highest demands on accountability, cost-efficiency and flexibility. It was therefore no small decision for OA to initiate a migration away from proprietary solutions and towards F/OSS. After all, having invested into proprietary databases, CAD and GIS for decades, the threat of encountering costly vendor lock-ins was omnipresent.

One important driver behind the decision to switch to F/OSS was the unpredictability of licensing costs in the long term. Although not all proprietary licenses have an official time limit after which they must be renewed, many do have a factual one. Since software vendors frequently change (“update”) the file formats used by their software, and since these formats are generally undisclosed, users who do not renew licenses frequently will soon be unable to exchange data with their clients and contractors. However, the vendor is free to modify the terms of the license agreement with each new deal, exposing the licensee to the risk of rising prices or other changes for the worse, each time through the cycle. When the latest such issue hit OA in the form of significantly increased licensing cost for their proprietary desktop GIS, a decision was made to prioritize finding an F/OSS replacement for the proprietary GIS licenses.

Another, not less important, driver, was the realisation that the potential for software-based innovation in commercial archaeology and the establishment of new digital revenue models could only be unlocked on the basis of F/OSS. Given the economic climate of commercial archaeology, attempting to earn money through customised software development, consulting, and paid-for technical support and training was simply not viable, as long as it involved prohibitively high licensing cost on the side of either the service provider or the client. The result of this insight was the founding of Oxford Archaeology Digital (OAD), a division within OA with the objective to create and promote open source software for archaeology and to find new F/OSS-related business opportunities (<http://oadigital.net>). More than five years on, OAD still stands out as one of the most concerted efforts to realise these aims. Among its greatest successes is the release of gvSIG OA Digital Edition, the first F/OSS GIS to provide a complete drop-in replacement for proprietary desktop GIS. The involvement of OAD in the development of the open source desktop GIS gvSIG is a particularly instructive effort that will be elaborated on in the case study on gvSIG CE.

In addition to tangible output in the form of free software, OAD also contributed research in the field of digital archaeological site documentation. It was among the first to systematically assess and publish the feasibility of using open source software to generate highly detailed, three-dimensional models from overlapping digital images (Ducke et al., 2011) and the use of ultra-portable communication devices (“smart-phones”) to replace paper forms in the field. Not all of these efforts have led to fruition, but that was not to be expected. Even Silicon Valley produces more short-lived, technological failures than long-lived successes.

Indeed, in the context of this chapter, the more pressing question is whether OA's other aim, the migration of its in-house IT to F/OSS, was a success. After all, a full migration of such a large and complex operation to open source software could serve as a benchmark case for the successful switch of a critical production environment. Unfortunately, the answer in this case cannot be clearly positive; but the lessons learned by OA are still valuable.

First of all, it quickly emerged that social inertia was one of the greatest obstacles (see also Stallman 2002, p. 245–246). Human beings can be surprisingly attached to technology, to a point where they establish an emotional connection to an inanimate piece of hardware or software. Such attachment is not entirely irrational. After all, learning to effectively use complex technology requires a huge personal investment. Not only would that investment be partly lost after the switch to another technology, but more importantly, an individual's competitive “edge”, marked by e.g. “mastery” of a certain piece of software, would also become blunted. Such cases of interest of conflict call for clear company policies, proactive communication, intense staff training and the central deployment of new technologies, all of which consume considerable resources.

Unfortunately, the drain on OA's resources was such that the F/OSS transition could only be a partial success. The organically grown, bottom-up structure of OA

proved to be a hindrance for establishing a new central IT infrastructure based on open standards and technology. The case-by-case deployment of F/OSS solutions, although successful in the initial, testing stage, proved problematic in the context of larger operations. The technological blame for this lies squarely with the proprietary file formats and interfaces. As long as it remains legal for software vendors to deny their customers the details of their data formats, there will always be a risk that the cost for switching to an alternative solution will be overwhelming. Even if an in-house F/OSS transition can be completed, there remains the issue of clients handing in or expecting delivery of proprietary data formats to fit into their own workflows. Therefore, a complete phasing-out of proprietary software seems almost impossible to achieve for an operation such as OA's within the current market conditions. The next case study will tell the much-related story of a large public institution's F/OSS migration. But this time, the resources available are several magnitudes greater.

7.4.2 gvSIG and gvSIG CE: The Role of Social Capital in F/OSS

Without a doubt, GIS is one of today's most important software platforms. Of critical importance not only to science and research, but also to businesses, consumers, and public agencies, accessible (in terms of cost and ease-of-use) GIS is a corner stone of modern information technology. Until recently, however, a "drop-in" F/OSS replacement for proprietary desktop GIS, i.e. a system that would not require its users to completely rethink their approach to GIS, that would allow them to continue working without having to convert their data to another format first, and that would cover the entire workflow, from data editing and processing to map publication, was simply not available. The fact that this situation has changed dramatically, and that archaeologists, among others, no longer need to pay for expensive proprietary GIS, is in no small part thanks to gvSIG.

The history of gvSIG (Generalitat Valencia Sistema de Información Geográfica) goes back to 2003, in which year the Spanish software house Iver was awarded a contract by the Regional Ministry of Infrastructure and Transport of Valencia (CIT) to develop a new, open source GIS. The aim of the development, endowed with generous funding, was clearly defined: to replace proprietary solutions for spatial database access, CAD and GIS with one integrated software, functional and stable enough to be used in cadastral works, spatial planning and the management of public infrastructure. However, despite these promising initial conditions, gvSIG never managed to match its main F/OSS competitor, Quantum GIS, in popularity and has instead remained largely confined to a smaller, Spanish-speaking community. Today, in times of austerity, the CIT's ambitious project has lost much of its initial momentum and development activity has slowed significantly (see Boga et al. 2011). The case of this software, therefore, provides an illustrative example of the consequences that can arise

from relying on internal strength alone, and from failing to capture external social capital.

To understand what happened, one must look back at 2009. That year saw the release of gvSIG 1.9. Initially conceived as the last incarnation of the 1.n code base, which was to be succeeded by a completely reworked version 2.0, gvSIG 1.9 was in fact the first F/OSS desktop GIS solution that could be considered a fully functional replacement for established proprietary GIS. Unfortunately, however, the quality of the software did not meet the general public's expectations. It soon became obvious that error testing had been conducted entirely within the narrowly defined workflows at the CIT, and that, when faced with different types of data and use cases, obvious malfunctions ensued. Other problem areas were the incomplete English translation of the user interface, which sometimes left users without Spanish reading skills clueless about the content of on-screen messages, and a prevalence of rough edges in the graphical user interface that hampered productive workflows.

None of these problems would have been fatal within a regular open source project. In such a case, internal developers and external contributors would file error reports and add corrections to the code, until the software would become usable again and eventually another, better release could be made. In the case of gvSIG, however, it quickly became obvious that these mechanisms could not take hold. Despite being publicised as an open source project by CIT and its contractors, gvSIG appeared far from open as regards its organisation and management. When the project surfaced on the Internet, it materialised as an opaque entity, with a strict hierarchy and an internal decision making process that favoured controlled communication and blocked outside influence on important technical decisions. This was accompanied by rigorous routines and rules for code contribution that were a far cry from the low-key, fast turn-around practice, called "agile development", so popular with open source programmers (<http://www.agilealliance.org>; see also Robbins 2005).

To make matters worse, after the release of gvSIG 1.9, the project's technical steering committee decided to focus all future development efforts on the release of version 2.0, which was to be written from scratch and had no set release date. Reverting to an older version of gvSIG was also not a feasible option for most users, as the next-older version, gvSIG 1.1.2, was far inferior in terms of its functionality and lacked many capabilities required of a professional GIS solution. Thus, users were trapped between an outdated version, an error-prone version and one that existed largely as a prospect.

In hindsight, what happened next was to be expected. Oxford Archaeology (see preceding case study), an archaeological service provider that had invested into gvSIG and depended on it as the core element of its F/OSS GIS migration, used its own resources to improve gvSIG 1.9 to the point where it could be used for productive work. The result, named gvSIG OADE 2010 remains one of the best, most comprehensive options for free desktop GIS available today. Its success eventually moved the CIT's project team to follow up on the 1.9 release with further improved versions, in time leading up to the current 1.12 release.

While OA's actions certainly solved a technological problem, they also proved to be the final nail in the coffin for friendly relations with the "official" gvSIG project. Realising that it would not be able to maintain a project the size of gvSIG OADE 2010 on its own, OA started to look for collaborators. It found them in a number of users and developers that had been equally estranged by the CIT project team's practice, and a new project, called gvSIG Community Edition (CE) was started. In technical terms, the CE project is a "fork": Since its inception, two different development teams have been contributing code to the two different versions of the software. It is the natural fate of forks to drift apart. Currently, code written for both versions is still largely interchangeable, but a point of no return will eventually be reached.

As opposed to the OADE version, which was largely welcomed by gvSIG users, the CE fork caused more controversy, not only between the different camps, but also within them. However, looking back at the original reasons that led to the fork, it becomes clear that there was no better option. Since direct collaboration had become impossible, the only other choices for the "breakaways" would have been to turn towards a competing project, such as Quantum GIS, to start a new project from scratch, or to go back to proprietary solutions. Thus, as opposed to an often voiced opinion, forks are not the worst way to resolve such conflicts, but rather a common occurrence that in many situations constitutes the least harmful path, as they preserve at least some potential for collaboration. In addition, the investments of those partnering in the CE fork have been successfully preserved.

7.4.3 Survey Tools: F/OSS for Field Archaeology

The case studies discussed so far have provided evidence that financial and social resources must both be sufficient and managed with equal diligence if an F/OSS project is to be sustained. This, final case study is an example of a project that intends to put such insights into practice. Because Survey Tools (<http://www.survey-tools.org>), a project dedicated to creating light-weight F/OSS tools to be used in field documentation and surveying, is still in its initial phase, it is too early for a verdict on whether its approach to F/OSS in archaeology will ultimately be sustainable. However, its technological focus should be of greatest interest to the archaeological reader.

The primary motivation for Survey Tools lies in gaining flexibility and technological independence. Faced with an initial situation similar to that of Oxford Archaeology and the CIT, the State Heritage Management (SHM) of the German state of Baden-Württemberg needed to find a way of mitigating its dependence on costly specialist software for field documentation. This led to an internal review of actual user needs. Under the direction of the SHM's Digital Archaeology unit, current field workflows were analysed and individual F/OSS solutions were considered. A central element in the SHM's strategy was the transition of topographic survey activities from proprietary

CAD to F/OSS GIS (more specifically, gvSIG CE). To make this possible, a new software had to be devised to act as the link between the surveying hardware and the GIS.

The result was the development of survey2gis, a flexible and user-friendly open source tool, capable of processing raw survey records from devices such as total stations and GPS and converting them into topologically cleaned GIS datasets. After a prolonged phase of testing and refinement, the software was made available to the public as the first component of the Survey Tools. After the completion of the initial funding phase, the project must now look for sustainable funding outside of the SHM. This is done through a collaborative platform on the Internet, paid-for support and subscription models, actively advertised on specialist meetings and conventions.

One of the most intriguing aspect of the Survey Tools project is its ability to show how F/OSS can unlock innovation potential. Prior to the inception of survey2gis, the SHM's field workflows had oriented themselves along the lines defined by user interfaces and functionalities of proprietary software, such as CAD. With the freedom to create new, customised software, however, also came the freedom to inspect and modify existing workflows in order to make them more efficient. As a result, survey2gis is highly customisable and includes a number of features designed to boost productivity in the field. This is a significant type of return-on-investment that is often overlooked when comparing the license fee savings against the cost of open source software development and staff training.

7.5 Conclusions

This chapter was not written as a condemnation of either proprietary software or traditional business models. F/OSS and proprietary software coexist and will continue to do so for the foreseeable future, as the diversity of user demands and expectations calls for an equal diversity of approaches. From a financial point of view, there are certainly scenarios in which proprietary offerings are worth their money, provided that they can solve a clearly specified problem or make a specific workflow more cost-efficient. If such an off-the-shelf product suits the user's needs, then it may be the most readily available solution. And as long as software is really just used a tool, withheld source code might not be an issue. However, excessive or fluctuating license fees, the risk of vendor lock-in, a lack of shared investment options, and not least the serious limitations of closed-source programs in research and education all speak in favour of considering alternatives.

The fact that intra-disciplinary software development in archaeology remains confined to sporadic investment and small-scale developments, suggests that alternatives are indeed required. At present, however, opportunities for long-term funding of archaeological F/OSS remain rare and mostly restricted to the commercial sphere. In this respect, the role of universities and the academic funding system need to be reviewed. For an outsider, it can be hard to understand why public research money

is more often spent in a way that benefits software corporations, then in a way that benefits open research and the general public. It should also be noted that technological developments in the commercial and academic spheres are ultimately linked. Companies attempting to break free of vendor lock-ins are not the least struggling, because university departments keep teaching their students application-centric thinking instead of transferable skills. The fact that such curricular alignment with the proprietary software industry is often done in the name of “the job market”, must seem ironic to employers like Oxford Archaeology, and to all those interested in investing into archaeological F/OSS.

Strictly speaking, as far as education and research are concerned, there seems no justifiable role for closed source software, except to serve a tool-like purpose for the most menial and routine tasks. This might sound harsh, but the fundamental ideals of good scientific practice, in particular that of reproducible research, are simply incompatible with trade secrets and the many barriers that proprietary software imposes on the free flow of information (Stallman, 2002, p. 57–58). Software-based research can be expensive, but there is no reason why it should be prohibitively expensive to reproduce such research. Licensing costs have become considerable obstacles for public institutions and small research projects. At the same time, the side-effects of proprietary software business, in the form of excessive “intellectual property” enforcement and software patents, all of which go far beyond the original, fair-use intent of copyright law, are threatening free science (see Klemens 2005 for a detailed account; also Stallman 2002, p. 89–92 & 105–134).

Indeed, the problems of closed source software become strikingly obvious in the context of its academic use. In the scientific domain, peer review of software should be as mandatory as that of text, and source code should be considered a part of the academic output and published accordingly. It would be curious indeed for an academic discipline to encourage peer review for philosophical treatises, where it matters least, but not for scientific software, where it matters most! The “missing functionality” argument against the exclusive use of F/OSS, at least, is no longer valid (provided that it ever was). On the contrary, projects such as GRASS GIS and the R language for statistical computing are immense repositories of scientific methods.

The limits of the usefulness of software are ultimately set by the paradigm under which it operates. In archaeology, software has traditionally been viewed as a tool that just serves a well-defined purpose, but this view is too narrow. Complex programs represent the result of countless hours spent on brainstorming sessions, elaborate project designs, fundamental and applied research, creativity and problem-solving skills. Software must therefore be published in its source code form, so that it can undergo the collaborative cycle of peer review, exchange and refinement that is commonly called “research”.

Finally, it should be noted that technology, like everything human-made, is a social phenomenon. F/OSS tends to bring this fact into the foreground. Getting involved in an open source project exposes all collaborators to social dynamics that must be

managed well, if an open source investment is to bear fruits. While this can be challenging at times, building a loyal open source community will result in insightful, careful and sustainable development, and in the growth of technological infrastructures that are open, diversified and innovative.

Bibliography

- Barnes, N. (2010), 'Publish your computer code: it is good enough', *Nature* **467**(7317), 753–753.
- Bivand, R. S., Pebesma, E. and Gómez-Rubio, V. (2013), 'Applied spatial data analysis with r'.
- Boga, A., Puga, F., Eiries, A. and Varela García, F. (2011), 'Analysis on free software communities (I): a quantitative study on GRASS, gvSIG and QGIS'.
URL: <http://nosolosoftware.com/analysis-on-free-software-communities-i>
- Buckheit, J. B. and Donoho, D. L. (1995), *Wavelab and reproducible research*, Springer.
- Chambers, J. (2008), *Software for data analysis: programming with R*, Springer.
- Donoho, D. L., Maleki, A., Rahman, I. U., Shahram, M. and Stodden, V. (2009), 'Reproducible research in computational harmonic analysis', *Computing in Science & Engineering* **11**(1), 8–18.
- Ducke, B. (2012), 'Natives of a connected world: free and open source software in archaeology', *World Archaeology* **44**(4), 571–579.
- Ducke, B., Score, D. and Reeves, J. (2011), 'Multiview 3d reconstruction of the archaeological site at weymouth from image series', *Computers & Graphics* **35**(2), 375–382.
- Fogel, K. (2009), *Producing open source software: How to run a successful free software project*.
URL: <http://producingoss.com>
- Ghosh, R. A. (2005), 'Understanding free software developers: Findings from the FLOSS study', *Perspectives on free and open source software* pp. 23–46.
- Grassmuck, V. (2004), 'Freie software. zwischen privat-und gemeineigentum. 2., korr. aufl., red'.
- Hafer, L. and Kirkpatrick, A. E. (2009), 'Assessing open source software as a scholarly contribution', *Communications of the ACM* **52**(12), 126–129.
- Hatton, L. (1997), 'The t experiments: errors in scientific software', *Computing in Science and Engineering* **4**(2), 27–38.
- Ince, D. C., Hatton, L. and Graham-Cumming, J. (2012), 'The case for open computer programs', *Nature* **482**(7386), 485–488.
- Janssen, M. A., Alessa, L. N., Barton, M., Bergin, S. and Lee, A. (2008), 'Towards a community framework for agent-based modelling', *Journal of Artificial Societies and Social Simulation* **11**(2), 6.
- Kandler, A., Perreault, C. and Steele, J. (2012), 'Cultural evolution in spatially structured populations: A review of alternative modeling frameworks', *Advances in Complex Systems* **15**(2).
- Kansa, E. (2012), 'Openness and archaeology's information ecosystem', *World Archaeology* **44**(4), 498–520.
- Klemens, B. (2005), *Math you can't use: Patents, copyright, and software*, Brookings Institution Press.
- Knappett, C., Evans, T. and Rivers, R. (2008), 'Modelling maritime interaction in the Aegean Bronze Age', *Antiquity* **82**(318), 1009–1024.
- Krishnamurthy, S. (2005), *An analysis of open source business models*, MIT Press, pp. 279–296.
- Lakhani, K. R. and Wolf, R. G. (2005), *Why hackers do what they do: Understanding motivation and effort in free/open source software projects*, MIT Press, pp. 3–22.
- Lerner, J. and Tirole, J. (2004), 'Economic perspectives on open source', *Advances in the Study of Entrepreneurship, Innovation & Economic Growth* **15**, 33–69.
- McGowan, D. (2005), *Legal Aspects of Free and Open Source Software*, MIT Press, pp. 361–391.

- Merali, Z. (2010), 'Computational science: Error, why scientific programming does not compute', *Nature* **467**(7317), 775–777.
- Morin, A., Urban, J., Adams, P., Foster, I., Sali, A., Baker, D. and Sliz, P. (2012), 'Shining light into black boxes', *Science* **336**(6078), 159–160.
- Neteler, M. and Mitasova, H. (2008), *Open source GIS: a GRASS GIS approach*, Vol. 2, Springer.
- Raymond, E. S. (2001), *The Cathedral & the Bazaar: Musings on linux and open source by an accidental revolutionary*, " O'Reilly Media, Inc."
- Rey, S. J. (2009), 'Show me the code: spatial analysis and open source', *Journal of Geographical Systems* **11**(2), 191–207.
- Robbins, J. (2005), 'Adopting open source software engineering (OSSE) practices by adopting OSSE tools', *Perspectives on free and open source software* pp. 245–264.
- Rosen, L. (2005), *Open source licensing*, Prentice Hall.
- Sherman, G. (2008), *Desktop GIS: Mapping the Planet with Open Source Tools*, Pragmatic Bookshelf.
- Stallman, R. (2002), 'Free software, free society: Selected essays of Richard M. Stallman'.
- von Hippel, E. (2005), *Open source software projects as user innovation networks*, Cambridge, Massachusetts: The MIT Press, pp. 267–278.
- Wheeler, D. A. (2005), 'Why open source software/free software (OSS/FS, FLOSS, or FOSS)? look at the numbers!'