Open Source Software—A Tellurian Perspective

Tellurian Pty Ltd

February 18, 2004

Abstract

Tellurian is a software development house which specialises in contract software development and in support of custom software in the medium to large enterprise. Open source software provides IT consultants and software engineers with a broad base of reusable components, trial systems and complete solutions. By taking advantage of these tools, Australian IT vendors can compete very effectively with overseas software houses for the delivery of software solutions. In this paper, we provide a brief overview of open source software as it relates to our business and the services we provide and outline some of our observations about open source.

This paper was written for presentation at a meeting of the Open Source Special Interest Group of the Open Interchange Consortium. The paper is copyright © 2004 Tellurian Pty Ltd., All rights reserved.

1 Introduction

1.1 Tellurian background

Tellurian was formed in 1995 to provide a vehicle for custome software development services and give sustainable support to a major client. In addition, Tellurian provides contracted support of open source software to corporate users, enabling them to benefit from a vast array of software with a safety net of having contracted technical expertise available when it is needed.

1.2 A definition of Open Source Software

For information about how open source software process works, see the references at the end of this paper, especially [Raym 00]. For our purposes, we will use the definition of open source, from [OSI web]. Open source is:

> A method and philosophy for software licensing and distribution designed to encourage use and improvement of software written by volunteers by ensuring that anyone can copy the source code and modify it freely.

The open source development community are highly committed to open source projects and invest a considerable amount of emotional energy in developing and improving their work. This devotion to a project, along with the fact there are often many users and developers, working on many platforms, leads to highly portable and quite robust code.

2 Open source software in Tellurian projects

During the past eight years, Tellurian has worked on many projects which have relied directly on open source, both in the development phase and for deployment:

- machine control of wool sorting machine (gcc, Linux, X-Windows, PostgreSQL),
- nursecall scheduling and paging system (gcc, Linux, PostgreSQL) + non-open source (Java, client code),
- general purpose, management reporting system (perl, Apache) + proprietary (Oracle, Tru64 Unix),
- Windows network services (ISC DHCPd, tftp, packet driver),
- embedded, systems such as a firewall and web proxy (Linux, various tools),
- user-level bandwidth management for an ISP (FreeBSD and ipfw).

Use of open source software can be at a technical or non-technical level. At a non-technical level, we obtain an application (eg OpenOffice.org) and use it without making any attempt to understand or tweak it beyond what is available through the application menus. At a technical level, we obtain an open source software base (eg FreeBSD and its firewall software) and deploy it in our own way, utilising our understanding of the software to do so. If we detect problems with open source software, Tellurian contributes changes to the authors for inclusion in the next release.

We are heavily dependent on software to support our business operations and within any project we find ourselves using open source for:

software development whether it is under Linux, Windows, Mac OS and other Unix operating systems. We use open source design and code maintenance tools, compilers, interpreters and debuggers.

documentation using systems such as TeX and OpenOffice.org as well as a small number of diagramming tools,

administration support such as document tracking, time sheets, project management, bug tracking and work request tracking,

communication including email, internet access and file sharing,

intranet and web service delivery, especially for the administrative tools and for internal project information and procedures,

custom hardware systems using a collection of Open source components and our own build and deployment tools to create diskless thin clients and other devices.

Not all the software tools used within Tellurian are open source, for example, we use proprietary software when:

- the client mandates its use (either for software development or documentation),
- we can't find equivalent open source software, for example, we use PC-Lint as one code checking tool,
- old habits take precedence (for example, where we happen to like a particular proprietary product),
- if the solution is not critical or is only for a quick fix to an immediate issue,

- a commercial solution provides better performance which we need.

Overall, we have found the benefits of open source software to be:

- we save direct software license costs (direct costs and TCO[1]),
- we can plan long term because we do not have to worry about whether a particular vendor will go out of business or cease support for a product (longevity and equity),
- we develop skills in the integration, deployment and maintenance of a wide variety of software,
- we can actually provide meaningful support to our clients because we can diagnose and address problems using the source code.

3 Observations

The following observations have been made by direct experience and by review of news and industry information over the last ten years.

3.1 Misconceptions

We have observed many discussions about open source software where people seem to base their argument on a number of misconceptions. Sometimes it is difficult for IT management to keep up with changes in the industry, however it is important that decisions with the potential to affect the future of the organisation are informed correctly. Some of the misconceptions we have seen are:

Open Source is a fad No. Actually very 'old' in concept and in practice. There is a paper [**?**]about the design of the multics operating system (circa 1965) where the authors identify the importance of publishing the system to enable public review and more informed use of the system.

There is no support Not so. Very often the open source product developer has made a significant emotional investment and is keen to help when bugs are found. Mailing lists and the wider community provides support and advice. This information is often archived and available (see google). Various IT vendors (eg., Tellurian Pty Ltd) also provide support contracts for open source software.

Open source is not a viable business model Not true. Many businesses make money by the use of open source. Publication of company source may force a shift toward service, rather than product fees but may also reduce corporate risk and increase the life of the software. The open source approach favours a consultant/community cooperation model, rather than a corporate controlling model. In [Raym 00], Raymond explains the benefits of treating your users as co-developers.

Open source is not credible, it's just a bunch of hackers Not so. Open Source is supported by very big names in IT (HP, IBM, Sun, Oracle, SAP) For example see [CMSEI web] .

It is worth noting, however, that open source software is not a panacea and users should try to use the most appropriate tool for the job. There is a very wide range of open source software and also a very broad range of software quality. Before committing to the use of an open source solution, an

[1]Total cost of ownership

organisation would be well advised to follow a careful acquisition process, as for any other software product.

If, during an evaluation process, there turns out to be two equivalent solutions, one open source and one proprietary, the following considerations may be relevant:

- The open source solution will probably outlast the proprietary solution.
- The quality of the open source solution can be reviewed more easily (quality of proprietary solutions can be hard to measure).
- The open source solution can be tuned to corporate requirements and can form part of the corporate IP.
- The open source solution cannot be taken away by forced upgrade or supplier collapse.
- The proprietary solution may be easier to install and may be easier to operate without employee training.
- The proprietary solution may perform better.

3.2 Advantages

Cost According to TheOpenEnterprise.com (Tom Smith, 10 Nov 2002), a survey of 260 companies indicated a belief that Open Source Software has at least 25% lower total cost of ownership than proprietary software (http://www.techweb.com/wire/story/TOE20021008S0001). Note that it is extremely difficult to compare costs for proprietary and open source software since the TCO depends on many factors, other than initial purchase price. This is a topic for ongoing debate, however it is unlikely that the TCO for open source software would be higher than proprietary software, especially when the end-of-life issues occur for proprietary software.

Equity The question of investment into open source software versus proprietary software can be compared to the question of owning versus renting a home. When you own your home you have long term equity. You can modify the house to your needs and it is worth your while to do so. The same is true of open source software. You can modify the software to suit your needs and you can keep the software.

When you are renting a house you may not adapt the property and you have little control of when it will be taken from you. Similarly, when using proprietary software, you cannot customise the software and you can't control when the version of the software you are using will be no longer supported by the vendor. At least when renting a house, there is a a guaranteed lease period, not so with software. Software updates be required at any time and these updates sometimes come with a whole new license model and cost structure.

Longevity Much of the oldest software still in use today is open source (eg TEX/LATEX). The open source model has shown itself to be an excellent approach to provide long term maintenance and excellent portability for software. Since the open source software is generally fairly portable, it is also easier to update it if necessary, when the underlying operating systems is upgraded.

Viruses No software system is immune from viruses. Open source software had tended to be very robust so far. There is a perceived risk that it is simple to insert bogus code into open source software and there have been attempts to insert backdoors into the Linux kernel which have been detected and stopped. Backdoors have also been inserted into commercial code but these are much harder to detect and remove. Without the source code it is impossible for the end user to do so.

From a biological perspective, diversity provides excellent insurance against problems which occur in mono-cultures, for example against viruses designed to attack specific host systems. It is likely that diversity in software is good protection for software infrastructure in the same way.

Bug fix turnaround With open source, is often possible to communicate with the individual developers of open source software whereas this can be difficult with proprietary code. We have found this to be a rewarding experience, since by reporting bugs helps we can help improve the quality of the open source code.

The core team for each open source product may change over time but when this occurs, there is often a handover process giving some continuity. In the proprietary arena, software is not always supported once its short shelf life has expired or when it becomes unmaintainable because a key person has left the organisation.

Hardware costs Open source operating systems have remarkable portability. Linux, for example runs on everything from PDAs to IBM mainframes. Low cost hardware can be used to provide core IT services. Redundancy can be achieved by simply purchasing spare equipment, matching the production hardware, this can be installed and configured without the need for extra software licences.

Choice Open source software presents a very large, diverse body of solutions which work on many platforms. Users are able to make informed decisions regarding the solution which best fits their individual needs.

3.3 Intellectual property

The use or otherwise of open source software is closely related to issues of intellectual property. Software developers do not always like to ‘give away’ their intellectual efforts and open source software is sometimes seen as a way to ensure that the need to pay patent royalties does not limit the continuing innovation of software in commercial and non-commercial arenas.

According to Dravis in [Drav 03], the US patent office issues in excess of 20,000 software patents each year, more than 1,500 of them for Microsoft. This represents an amazing rate of development of intellectual property and it is simply impossible for developers to know whether their work infringes any of these patents. Also, there is a lot of debate about whether these patents always make sense in that there may be many cases where a patent is granted for a technique which is actually prior art.

From the Tellurian perspective, when developing software for a client, we generally work from the ground up or from the legacy software belonging to the client. This process usually creates new intellectual property (IP) for our clients and sometimes develops IP for Tellurian. The client is generally keen to use this method because they own the results of the investment.

We have found that commercial agreements are careful to define intellectual property[2] and to define ownership of any developed IP. However, the organisation, having collected potential IP, rarely develops a patent application for it and often fails to draw maximum benefit from it by reusing it within other areas of the business.

[2]Usually a blanket statement about intellectual property being anything built ‘during the course of the project’

There may be a commercial risk here, since IP which is not properly registered at the time it is developed may be invented elsewhere and registered by a third party. This may restrict commercial operation of the first organisation, especially if this IP turns out to be of particular commercial value or a critical component of a commercially valuable product. It may be that by publishing the code, the organisation must face early competition from others employing the published code at the same time, the organisation can more easily demonstrate its technology as prior art if patent disputes arise in the future.

3.4 Risk

Our larger clients seem to have a strong resistance to the use of Open Source Software, possibly because the need to manage risk and the perception that this is best done with proprietary software. Other clients embrace open source technology (Linux and associated tools, in particular) and believe the savings in software and maintenance costs, combined with its other benefits provide a significant competitive edge for the business.

Tellurian attempts to help minimise perceived open source software risk by providing support for open source software under contract. With this service, Tellurian is offering to identify and resolve the issue in the case of open source software failure.

Proprietary tools have a future risk: the risk that the vendor will be not be around in the future and or that it will lose interest in supporting the software. This is a risk for the organisation and for the staff who have invested their time developing expertise in the software.

3.5 Longevity

Even more important, is that open source software is maintained over a much longer time period than is common for proprietary software. This may be because the criteria for maintaining open source software is only that there is a (small) number of technically competent people who are interested in the software and that these people are able to communicate easily and inexpensively (i.e., over the internet). In contrast, the criteria for maintaining commercial software is that it is profitable to do so (either directly or indirectly).

3.6 Equity and Investment

The long life of open source software is especially important, regardless of other considerations. If the software has a long life and your organisation can customise it (internally or outsourced), it is worth investing in this software as corporate equity.

Organisations invest in software, by licencing, development and training. Individuals, especially IT staff, also invest heavily in understanding their software environment from a technical and emotional perspective. The fact that people make these investments must be considered when planning the medium to long term future of an organisation. Proprietary software packages (including proprietary operating systems), will inevitably fade over time as the vendor's commercial incentive to maintain the software disappears. At the same time corporate and personal investment in these packages will lose value and staff may lose motivation.

If an organisation has encouraged staff to develop expertise in proprietary tools, it is not a good idea for the organisation to try to immediately change direction to pursue the long term advantages of open source without helping staff redirect their personal investment in a similar direction. Any organisation attempting to do so is likely to meet significant resistance to the change. We therefore recommend that IT managers be sensitive to these issues and consider the long term value of open source skills when advising his/her staff about their professional development plans.

4 Conclusion

In this short paper, we have described a variety of ways in which our company benefits from the use of open source. Having seen many proprietary solutions vanish over short time periods, at Tellurian, we tend to see proprietary tools as limited quick-fixes, not long term solutions. At Tellurian, we continue to use proprietary tools if we are emotionally attached to a proprietary tool or if we need a quick fix (games fit this category well) or if a proprietary tool is the best tool available and we can bear the future risk.

When we need a long term solution, when we want to invest effort in mastering a tool, when we need a secure, flexible or modern solution, we use open source. We almost always use open source.

5 Resources

The following references provided some background for this paper and may provide interesting further reading. The internet and open source software are tightly coupled and I don't think either could exist without the other. There are many ways to find help and google (http://www.google.com) is an excellent starting point. There are also many ways to find open source and sourceforge (http://www.sourceforge.net) is excellent.

References

[Broo 75] Frederick P. Brooks, Jr., *The Mythical Man Month*, Addison Wesley, 1975.

[CMSEI web] Carnegie Mellon Software Engineering Institute web site. http://www.sei.cmu.edu/opensystems/

[Drav 03] Paul Dravis *Open Source Software — Perspectives for Development*, Nov 2003. http://www.infodev.org/symp2003/publications/OpenSourceSoftware.pdf

[Multics web] The Multics overview web document http://www.multicians.org/fjcc1.html and for more information about Multics: http://www.multicians.org/

[OSI web] Open Source Initiative web site. http://www.opensource.org/

[OSSI web] Open Source Software Institute web site. http://Open source-institute.org/

[Raym 00] Eric S. Raymond, *The Cathedral and the Bazaar* http://www.catb.org/∼esr/