Open Source Software

A Factfile provided by the Institution of Engineering and Technology
About This Factfile

The Institution of Engineering and Technology acts as a voice for the engineering and technology professions by providing independent, reliable and factual information to the public and policy makers. This Factfile aims to provide an accessible guide to current technologies and scientific facts of interest to the public.

For more Factfiles and Position Statements on engineering and technology topics please visit http://www.theiet.org/factfiles.

The Institution of Engineering and Technology

The Institution of Engineering and Technology (IET) is a global organisation, with over 150,000 members representing a vast range of engineering and technology fields. Our primary aims are to provide a global knowledge network promoting the exchange of ideas and enhance the positive role of science, engineering and technology between business, academia, governments and professional bodies; and to address challenges that face society in the future.

As engineering and technology become increasingly interdisciplinary, global and inclusive, the Institution of Engineering and Technology reflects that progression and welcomes involvement from, and communication between, all sectors of science, engineering and technology.

The Institution of Engineering and Technology is a not for profit organisation, registered as a charity in the UK.

For more information please visit http://www.thei.org

© The Institution of Engineering and Technology 2014

The Institution of Engineering and Technology is registered as a Charity in England & Wales (no 211014) and Scotland (no SC038698).

Enquiries

policy@theiet.org
Introduction

The shift to Open Source Software is one of the most important current trends in technology, yet it is surprisingly little discussed compared with other developments - including those that, ironically, have only been made possible by the adoption of open systems.

Open Source can offer huge benefits, enabling faster innovation and reduced total cost of ownership. Whilst transitioning from closed to open systems is no trivial task, unless this step is taken, businesses risk being left behind as their competitors capitalise on the new possibilities this offers.

A brief chronology of open systems

Open systems in computing and informatics refers to a class of systems which are built using Open Source Software (OSS) standards and that offer a good level of portability and independence from the hardware platforms on which they operate. They usually include a right to edit and redistribute and have particular characteristics including:

- **no intentional secrets**: the standard must not withhold any detail necessary for interoperable implementation.
- **availability**: the standard must be freely and publicly available under royalty-free terms at reasonable and non-discriminatory cost.
- **patents**: all patents essential to implementation of the standard must be:
  - licensed under royalty-free terms for unrestricted use, or
  - covered by a promise of non-assertion when practiced by Open Source Software.

There are three key types of licence under which OSS may be released:

- the GNU® General Public License (GPL) requires that altered or extra code added to GPL software be also licensed under the GPL. This ensures the propagation of OSS but can cause licensing conflicts if GPL and proprietary software are combined.
- the Berkeley Software Distribution (BSD) licence gives anyone the freedom to release updates or modifications of the software under any licence they wish.
- the Lesser GPL (LGPL) is a compromise between the restrictive GPL and the permissive BSD. Altered LGPL software must continue under LGPL, but extra code can be added under almost any licence the author wishes.

The concept of open systems dates back to the late 1960s and early 1970s, as the first steps were taken to link heterogeneous computer systems together across communications networks. Networking developments took place principally in two communities, the defence/academic world, leading to the development of ARPANET and ultimately to today's internet, based on the TCP/IP® protocols, and by the public telecommunications operators using a different protocol stack (the “X” series) to achieve similar aims. The reality of multiple competing solutions led to the creation of a single open systems interconnection (OSI) reference model.

Once open solutions were available for networking, attention turned to the attached computer systems. Could the software, both operating system and applications, be made more independent of the particular hardware platform or vendor used? The most widely recognised precursor of today's open operating systems came again from the telecommunications world. AT&T had first developed “Unix®” in 1969 at its Bell Laboratories subsidiary. In 1973 Unix was very largely re-written in the high level programming language “C” making it much easier to transfer between multiple hardware platforms. In its later incarnations in the 1980s, Unix embodied standardised programming interfaces and management of attached devices.

Unix was the inspiration for the development of Linux® the basis for the vast community of open software and applications in use today. The first Linux Kernel® was released by Linus Torvalds in 1991. Some 20 years of development and operating experience, sourced from around the globe, have now gone into the Linux based open operating systems.

**From closed to open systems: a spectrum**

It would be wrong to regard the question of whether an operating system is open or closed as simply a binary choice. The reality is that most practical implementations lie somewhere on the spectrum between these two extremes. Many current systems draw on the Unix/Linux legacy whilst arguably now being closed. Examples include the Apple operating systems OSX and iOS, Oracle Solaris and Blackberry BBX. Others build in their own proprietary additions for example IBM’s AIX and HP’s HPUX. Still others remain very largely open such as Google's Android.

Even where a core Linux system is used, there will often be a need to purchase maintenance and support services. The major vendors, such as Red Hat and SUSE, have every incentive to build in some competitive differentiation by customising their various services and tools, particularly in the area of system management.

**Virtualisation: from open systems to cloud computing**

Virtualisation is again a development first introduced in the 1970s. A single mainframe computer could run many separate instances of the same operating system at the same time under the control of a “hypervisor”; in effect each instance of the operating system simulated a stand-alone machine. All were hosted simultaneously, but separately, on the same hardware. The use of such virtualisation techniques with Linux
style open systems has generated major benefits with single machines supporting multiple virtual servers. For applications that have relatively low load factors the advantages are substantial, lower capital costs through better utilisation, plus ease of management and provisioning.

The final step to create cloud computing was the development of the technique known as “orchestration”. This allows many virtual machines to be managed across a range of physical hardware. As load increases for a particular application, further virtual machines can be started automatically and, if necessary further physical systems assigned to support of these in-demand virtual machines and applications.

Almost all of the successful web businesses launched in the last 10 years have drawn on this base of virtualised open systems; familiar names such as Google, Amazon, YouTube, Facebook and Twitter all operate in this way.

Advantages of Open Systems in information processing

Access to greater innovation: The combination of Open Source Software, open systems interoperability and open standards has created a self-reinforcing community of shared research and development and a pooling of creative ideas. This leads to a stream of innovative applications for test and development. Those that show promise are implemented and adopted more quickly.

Open Source is driving innovation. In a speech at the Linux leaders annual summit in June 2012, Red Hat CEO Jim Whitehurst commented that the information age is finally evolving into the information economy because of the standardisation enabled by Linux and Open Source. He said that: “...Open source has gone mainstream ... open source is the default choice of the next generation IT architecture...” and “...More innovation will happen first in open source and that’s a radical change from even five years ago...”

Open Source’s responsiveness. Alex McLachlan of Indigoblue Consulting shared in his blog that: “One of Open Source's unique selling points (if I can use that phrase) is its responsiveness. We've had a really good example of this on our website recently, which demonstrates how issues with Open Source Software can get fixed really quickly and efficiently. We use the Drupal open source website content management system and wanted to add in the Amazon module that lets you link directly to Amazon contents. We had an issue with how the links to Amazon were appearing... The guy doing the development posted the problem on the Drupal website and resolved the cause of the problem within a few days. This experience contrasts with experiences I've had with companies supporting products, where it can be difficult to get fixes made between product releases, so any significant issue can mean waiting until the next product release.”

Enhanced ability to support a plethora of systems: The world is a highly heterogeneous environment. Business organisations use a wide variety of current and legacy systems all at different stages of their lifecycle. Some are widely supported, some less so. The ability, through open systems, to draw on a very extensive range of standards and interfaces shared and developed over 20 years can be a key differentiator.

Access to skilled, motivated and innovative staff: A generation has now grown up with the internet and Open Source. They want to work with these systems and tools rather than the closed and proprietary approaches that are frequently seen as more limiting. Open systems developments (such as below) are seen as particularly desirable.:

- Drupal (an open source content management platform powering millions of websites and applications. It's built, used, and supported by an active and diverse community of people around the world);
- Hadoop (The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using a simple programming model. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of

Quicker response to changing requirements: The pace of change in many markets, especially those with a consumer focus, is now relentless. The focus on customer engagement and involvement and the influence of burgeoning social networks place further demands on immediacy of response. Access to a wide range of cost-effective development tools, ability to scale rapidly and access to global community of shared knowledge all favour the use of the open approach.
jQuery® (a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development);

Rapid exploitation of new technology developments: Under the continuing impetus of “Moore’s Law” new developments in hardware (such as ultra high resolution displays, enhanced wireless networking and network based storage) continue apace. New developments in software and systems, for example in Business Analytics (the so-called “big data”) are also a regular occurrence. It is important to have timely access to these. The Open Source community once again offers rapid access.

Drawing on a global community for knowledge tools and problem solving: Developed over the last 20 years, the Open Source community now numbers in the millions of software architects, analysts, designers and programmers. The community also extends to key universities and facilitates access to leading edge research. There are extensive facilities for collaboration...

GitHub™ is a social network for open source programmers. It is based around the “Git” the extremely fast, efficient, distributed version control system for the collaborative development of software created by Linus Torvalds. It supports code repositories, issue tracking, collaboration management, code reviews and so on. More than two million software developers are linked through GitHub.

openstack™ is a global community of technologists, developers, researchers and cloud computing experts originally set up under the aegis of the US National Aeronautics & Space Administration (NASA). It currently involves more than 3000 people from more than 180 organisations. It is based around a massively scalable “cloud” operating system.

Limiting restricted dependencies: The Open Source community avoids dependency on single sources and sharply limits the potential of pricing power. For example, in terms of support for the Linux operating system, competition is based on quality of customer service and additional management facilities rather than on access to the individual system releases. The purchaser has more freedom regarding when to upgrade to a new release...

Lower total cost of ownership: Open Source Software is generally free to obtain rather than being subject to the licence fee normally charged for access to proprietary software. However this is not the correct basis on which to compare costs. It is more appropriate to consider the total cost of ownership (TCO) over the lifetime of use of the software. The London School of Economics (LSE) published a report on TCO for the United Kingdom Cabinet Office in 2011. This report found that: “The highest score for strategic drivers was for reduced vendor lock in. A close second was value for money.” The report (based on a survey of 32 organisations) found that TCO was often lower overall with Open Source Software. However, based on a model of five life cycle phases (Search for a solution, Acquisition, Integration, Use and Retirement) structured into 14 separate areas of cost, analysis should really be carried out on a case-by-case basis. In so far as generalisation is possible, industry lore is that a TCO saving of around 20% is achievable.

Full visibility of (and confidence in) the source code: There remains no broad agreement in computer security circles as to whether open visibility of source code contributes to, or detracts from, system security. Access to the code is clearly helpful to an attacker, but in the same way extensive public review and correction of the code can minimise the avenues for attack and offer more rapid identification and blocking of
successful attacks. These issues are discussed in some depth in the 2002 paper *Security in Open versus Closed Systems - The Dance of Boltzmann, Coase and Moore* by Ross Anderson, Professor of Computer Security at Cambridge University. In so far as a consensus exists it is that in pure security terms there is little difference between open and proprietary code. A report by the Communications and Electronic Security Group (CESG) of the UK Government Communications Headquarters (GCHQ) published in December 2011 concluded that: "Open Source, as a category, is no more or less secure than closed proprietary software". Trust in "security by obscurity" though does seem misplaced in the proprietary world, as the code is not obscure to insiders. Furthermore, full visibility of code helps to ensure that no “backdoors” or surprises are hidden away…

Lessons from the transition from closed to open systems

As the path away from dependency on proprietary systems becomes evermore well worn, it is helpful to document the “lessons learnt” by those who have already made, or are deeply engaged, in that transition. There is no value in revisiting pitfalls already explored.

Maintaining a common development/support team across existing and replacement systems: the key reasons for maintaining common teams are:

- allowing the staff that have been responsible for supporting the old systems to be trained in the new programming languages, methodologies and toolsets, thus demonstrating commitment to avoiding them becoming a “stranded asset” as the old systems are progressively retired.
- maintaining the “institutional memory” of why particular design and implementation choices were made, these may still be relevant to the new developments.
- giving development teams a role in support as well, thus ensuring that they are fully aware of, and live with, the consequences of implementation defects.

Accepting the challenge of the task and being realistic about the costs, manpower and timescale required: this is a major undertaking. Often this represents a significant business change, not simply an IT change. New systems must be developed whilst maintaining high levels of service to customers through the existing systems. In the words of Henry Kissinger: “…the urgent often forces out the important”. Everyone was busy before, now there are lots of extra change related tasks as well. Emotions rise as workloads increase and people are inexorably sucked into the imperatives of day-to-day business rather than fully addressing the change.

Recognising that it is more than replicating existing functions: a careful analysis is required to:

- weed out requirements that are simply no longer relevant.  
- carry out needed rationalisation and simplification of requirements changes that have accumulated in the old system over years; and
- include new features and capabilities to support anticipated customer requirements.

Despite the best of intentions to minimise change, it may simply not be possible to freeze all new development requirements for the duration of the change programme. Some degree of revision during the development process may be inevitable.

Significant changes in operational procedures will need to be planned and tested including:

- **Ensuring data integrity during rollback/recovery:** Where elements required as part of a specific transaction are distributed across multiple databases, whether on specified servers or within a cloud, special measures are required to ensure data integrity is maintained if the transaction is aborted or some element is subject to failure. This can represent a significant challenge for example in rolling back such a system to a known prior state. The traditional database properties of “atomicity”, “consistency”, “isolation” and “durability” (ACID) all need careful attention in the context of maintaining integrity in the presence of aborted or failed transactions.

- **“Seeing the wood for the trees” in fault management:** In distributed or cloud based systems a single initial failure can lead to a rapidly branching tree of subsequent errors and alarms. Expert system techniques may be required to isolate rapidly the root cause otherwise hidden in the noise of cascading failures.

Acknowledgement


The permission of Amadeus to reference this material is gratefully acknowledged.
References

3. GNU is an open source operating system.
8. Operating system core.
10. For the full text from the Google blog, see: [http://googleblog.blogspot.co.uk/2009/12/meaning-of-open.html](http://googleblog.blogspot.co.uk/2009/12/meaning-of-open.html).
16. Moore’s Law, named after Intel co-founder Gordon E Moore, comes from a 1965 statement by Moore that the number of transistors contained in a state of the art integrated circuit would double every two years. This corresponds roughly to a doubling of processing performance every 18 months.
17. See: [https://github.com/features/community](https://github.com/features/community).