

Open Source for Open Government – A Strategic View.

Presentation to ECPRD - ICT Seminar, Den Haag, 4 September 2002.

Eur Ing Dipl Sci Andrew Hardie, FBCS, FIMIS, FIAP, MIEEE, MΣΞ
Independent Parliamentary Information Systems Consultant.
Email: ash@cellar.demon.co.uk

1. Abstract

The use of Open Source Software (OSS) in government and public administration is no longer a far fetched idea. Around the world, it is now being taken seriously both by legislators, eager to ensure transparency and best value for money in the delivery of public services, and by the large software companies who stand to be most affected by a shift from the proprietary to the open source software development model. Open Source has moved from being a hacker's utopian dream to become a serious contender in a market worth € 6.6 billion this year for European e-government alone. High stakes, indeed, but money is not the only issue for the public sector market. In the post-Enron era, corporate governance and the way companies shape and manage their markets have become high-profile political topics. Reducing the dependence of government on large companies, in a market sector where corporate ethics have been demonstrably bad, is a new strategic consideration and one which is not about just the software but also the public data created and managed by that software.

2. Background

The Feira European Council in June 2000 initiated the e-Europe 2002 Action plan. Amongst the recommendations to come out of this were:

Public administrations should:

- Develop **internet-based services** to improve access of citizens and businesses to public information and services,
- Use the Internet to **improve the transparency of the public administration** and to involve citizens and business in decision making in an interactive fashion. Public sector information resources should be made more easily available, both for citizens and for commercial use,
- Ensure that digital technologies are fully exploited within administrations, including the use of **open source software** and electronic signatures.

Much has happened since then. The Commission's Fifth Framework program had over 25 projects to develop or evaluate OSS¹. The subject of OSS in government and public administrations is now a mainstream topic, with actions and initiatives not just in Europe but around the world and a conference on the subject planned in Washington in October, "Open Source: A Case for e-Government²". The topic has now reached the stage where serious consideration of OSS in public administration needs to be made on a strategic basis, not just on crude cost comparisons. It's no longer just about money, if it ever really was. It's about proprietary lock-in, security, efficiency, effectiveness, transparency and the integrity of public information. Most of what applies to public administrations applies to parliaments but they have some special considerations of their own, especially in the areas of public record, transparency and accountability. It's time to consider Open Source Software in a new light.

3. EU Policy developments

What did the e-Europe 2002 Action Plan achieve? It claims as its achievements:

- Internet penetration in homes doubled
- Internet access prices have fallen
- Most companies and schools on the net
- World's fastest research backbone network
- E-commerce legal framework largely in place
- More government services available online
- Smartcard infrastructure is emerging

Whilst one could argue that the first two would probably have happened anyway, propelled by the dynamics of the market, the plan was effective in encouraging implementation in the other areas and was particularly valuable in raising the awareness of the link between the exploitation of ICTs and competitiveness for policy makers. It was, of course, drawn up at the height of the "dot.com" new economy boom and the world today is very different, in so many ways.

However, the e-Europe initiative is still alive and well. On January 28 2002, the European Council adopted a resolution on a common approach and specific actions in the area of network and information security, asking Member states

- "by the end of 2002 to take significant steps towards effective and interoperable security solutions based on recognized standards where possible – which could include **open source software** – in their e-government and e-procurement activities ..."

The new e-Europe Action plan "e-Europe 2005 – An Information Society for all", was initiated by the European Council meeting in Barcelona in March, calling for (emphasis added):

- "the Commission and the Member States to foster the use of **open platforms** to provide freedom of choice to citizens for access to applications and services of the Information Society, notably through digital television, 3G mobile and other platforms that technological convergence may provide in the future."

The Action Plan was adopted³ in May, with the key objective being:

- "to provide a favourable environment for private investment and for the creation of new jobs, to boost productivity, to modernise public services, and to give everyone the opportunity to participate in the global information society. eEurope 2005 therefore aims to **stimulate secure services, applications and content based on a widely available broadband infrastructure.**"

In no less than three places in the plan, Open Source Software is specifically mentioned (emphasis added):

- "Interoperability. By end 2003, the Commission will issue an agreed interoperability framework to support the delivery of pan-European e-government services to citizens and enterprises. It will address information content and recommend technical policies and specifications for joining up public administration information systems across the EU. It will be based on open standards and encourage the use of **open source software.**"

- “Community research activity on security will continue under the Sixth Framework Programme. Priorities will be: trustworthy network and information infrastructures with an emphasis on emerging technologies (e.g. broadband, wireless architectures, ambient intelligence); the identification of vulnerabilities and inter-dependencies in infrastructures. It also intends to support standardisation with a view to wider use of open standards and **open source software**.”
- The detailed analysis of good practice should result in templates or guidelines. These provide proven, well-documented approaches to tried and tested applications for *e-services*. They would be modular and customisable for any particular user and would typically consist of a methodology, an associated set of tools and **software in open-source form**. This would result in a critical assessment of success factors and lessons of failure, which may lead to transfer and dissemination of good practice across Europe, particularly in the less favoured regions.

Then, in July, the Commission IDA (Interchange of Data between Administrations) office published a study they had commissioned from Unisys entitled “POSS: Pooling Open Source Software”⁴ which recommends that software developed for and owned by public administrations should be issued under an Open Source license to promote reuse. Reusing freely the work of others is at the core of the Open Source movement, as is intense, open peer review during and after development (see Eric Raymond’s classic paper “The Cathedral and the Bazaar”⁵). Both of these are absent from the proprietary development model; reuse is prohibited and beta testing is not peer review. As Raymond describes it, the cathedral building approach is characterised by much internal debugging work by a small team and long intervals between releases; the bazaar view is release early, release often (made possible by Internet delivery, of course) and listen to the feedback to get more corrections into the program, sooner.

External scrutiny (as opposed to peer review scrutiny as part of the development process) of the software used to perform, or assist, public decision making is another issue. For example, several authors have pointed out the importance of independent scrutiny of any software used in electronic voting (both for voting and for personal identification), to discover any mistakes, accidental or not. If you independently scrutinise conventional election processes, then e-voting must be scrutinised also; the opportunities for error or fraud are too great.

By extension, a similar case can be made for the software used for public interaction with government, both for services and information provision, to ensure that privacy, data protection and equal access laws are being followed. Civil Society scrutiny is a very important aspect of democracy and transparency; Open Source software has an important role to play in helping that scrutiny as government moves more and more towards e-government. Security through obscurity doesn’t work. Microsoft source code is obscure, in the sense of hidden, but that hasn’t stopped numerous security breaches. The best security is provided by wide examination of the code (“given enough eyeballs, all bugs are shallow”, to quote Eric Raymond, paraphrasing Linus Torvalds). This must be a high priority in the area of personal identity in e-government services. (A new Open Source Digital Identity Infrastructure project, PingID⁶, hopes to create an end-to-end digital identity management platform to handle multiple online identities.) Here’s a challenge for governments to test their commitment to Open Source: make all the code for citizen interaction with e-government available for independent scrutiny of its integrity and security – what a contribution to transparency that would make!

The POSS study also recommends that also that a “clearing house” should be set up, along the lines of SourceForge⁷, to which administrations could donate the software for reuse and which could provide expertise and “help create a community of developers, users and policy makers”. Thriving communities of users and developers are what make the best Open Source programs so successful (e.g.

Apache, Perl, Zope). Suggesting the involvement of policy makers as well is a very welcome development and one that was called for in 1999 in Mitch Stoltz's NetAction paper "The case for Government Promotion of Open Source Software"⁸.

4. Developments elsewhere

Away from EU policy making, developments elsewhere have added to the momentum for public administrations to consider the use of OSS. In Germany, following an initiative earlier in the year to develop a standardised "Government Linux" distribution, the Interior Ministry has awarded IBM a large contract for public sector systems based on SuSE Linux. The Interior Minister, Otto Schilly, said "We are raising computer security by avoiding a monoculture and we are lowering dependency on a single supplier". Recent damaging virus and worm attacks, such as "I Love You" and "Nimda", exploiting weaknesses in near-ubiquitous Microsoft programs have clearly concentrated the mind. Not only are they damaging financially, but they offer opportunities for information warfare⁹ attacks by cyber terrorists. Anti-virus software only stops the virus; it doesn't fix the weakness the virus exploits. Who knows how many new virus delivery tricks are waiting to be used when the time is right?

In France, Mexico, Argentina, Brazil and Taiwan, legislators are considering if the Open Source model should be mandated for public service use. Not surprisingly, Finland, is actively investigating and considering Linux and other OSS products. Russia is reportedly examining the use of Open Source Unix-like operating systems, for reasons of "security, price and openness".

In the United States, where NASA and the US Navy are already widely using OSS, the US Army commissioned a study¹⁰ from the Mitre Corporation which concluded that:

- "... open source methods and products are well worth considering seriously in a wide range of government applications, particularly if they are applied with care and a solid understanding of the risks they entail. OSS encourages significant software development and code re-use, can provide important economic benefits, and has the potential for especially large direct and indirect cost savings for military systems that require large deployments of costly software products."

Mentioning care and risk is important: just because the source code is available doesn't mean that the software is any easier to manage and deploy or inherently lower risk but it does mean that it is easier to assess the risks than proprietary "black box" software because the source code is available.

Also, the issue of cost between Open Source and proprietary solutions is a lot more complex than the crude "Open Source is free, proprietary isn't, so it's obviously cheaper". Total cost of ownership is a much better measure of the cost. A number of recent studies, by Gartner, Forrester, etc., have shown that hardware and software together represent only about 20% of the total cost of ownership of a system, with a figure of 8% usually quoted as the typical software proportion. An 8% saving may be worth having but "free vs purchased" software is not a very strong cost argument in the face of the people costs of installation, support and maintenance.

Remember that, typically, 40% of the cost of developing a popular program is maintaining it. If you choose, for policy or operational reasons, to pay for support for the OSS you use, the cost will be similar to that of paying for support for commercial software.

One hidden, and harder to estimate, cost factor is the cost of downtime. Here, Open Source can add to the cost equation in its favour; Unix in its many forms, including Linux, is one of the most reliable operating systems ever written, widely used on mission-critical systems. The same cannot be said of Windows NT, yet it is still in widespread use, partly because of the difficulty and cost of migrating to either the more reliable Windows 2000 or XP or other non-Microsoft platforms.

In the UK, a direct result of the e-Europe 2002 Action Plan has been the publication of a policy document “Open Source Software – Use within UK government”. In a statement that, although somewhat late to recognise the existence and contribution of OSS, will warm the hearts of the Open Source movement, it declares that:

- “Open Source software is indeed the start of a fundamental change in the software infrastructure marketplace, but it is not a hype bubble that will burst and UK government must take cognizance of that”

One of the policy decisions is:

- “UK Government will explore further the possibilities of using Open Source software as the default exploitation route for Government funded R&D software.”

Not quite “Open Source everywhere” and somewhat vague, but still an important decision. However, another of the policy decisions states that

- “UK Government will seek to avoid lock-in to proprietary IT products and services.”

It’s difficult to see that as not being a reference to Microsoft and the search by governments to avoid proprietary lock-in, for reasons of security as well as cost, has set alarm bells ringing in the head offices of the big software companies. Instances of public sector customers using the threat of OSS as a way of negotiating further discounts from commercial software suppliers risk turning that alarm into anger.

In the case of Microsoft, this concern was demonstrated, rather more publicly than they might have wished, in Peru where the General Manager of the local Microsoft office, Juan Alberto González, wrote a letter¹¹ to Edgar Villanueva Nuñez, one of the two Peruvian Congressmen who had presented a parliamentary Bill¹² whose main objective was to employ exclusively free software in all the systems and computing equipment of every State agency. In this case, free software was defined as offering unrestricted use and copying of the program, unrestricted access to the source code, ability to reuse the software to develop other software and distribute freely the new software. It does not, necessarily, mean free of cost.

Latin America is, of course, very sensitive to the issue of US influence in the region (and the European political scene is not without some of its own anti-US sentiment) but it is important to not see this as being anti-American – the problem is not the US, but the anti-competitive and monopolistic behaviour of some of their global corporations. After all, the Open Source movement began in the USA, with BSD Unix, Richard Stallman and, even earlier, with the principle that software developed at public expense should be made freely available – remember Livermore Basic for the 8080 microprocessor?

Not surprisingly, Microsoft was not very keen on this Bill, and their letter set out some of their now familiar objections to the Open Source movement. More surprising was the Congressman’s reply and the wide publicity the exchange of letters received, elevating him to near-hero status in the Open Source movement. In his detailed letter¹³, he responded strongly and persuasively to all Microsoft’s

points and restated the three basic principles that inspired the Bill – principles that lift the debate above accusations and posturing and which all public administrations need to consider very carefully:

- **Free access to public information by the citizen.** To guarantee the free access of citizens to public information, it is indispensable that the encoding of data is not tied to a single provider.
- **Permanence of public data.** To guarantee the permanence of public data, it is necessary that the usability and maintenance of the software does not depend on the goodwill of the suppliers, or on the monopoly conditions imposed by them.
- **Security of the State and citizens.** To guarantee national security or the security of the State, it is indispensable to be able to rely on systems without elements which allow control from a distance or the undesired transmission of information to third parties.

Here, are the real issues for the policy-makers and the implementers of ICT systems for public administrations to consider. The third point refers to what Congressman Villanueva Nuñez refers to as “spy code” - code intended to transmit user information to the software supplier. The first two points touch on the same issue – closed, proprietary file formats holding the user’s data, formats that require the availability of the proprietary software that created the files in order to use them. Such proprietary file formats can also be regarded as a form of “viral marketing”; if someone sends you a proprietary file they created you, too, need the same proprietary program to make full use of the information.

XML is an important step in moving away from proprietary file formats but it is not a magic solution. Applying XML well is hard, because it requires you to think about what your information means and how you use and reuse it; it can’t be done like HTML or SGML were, each in a narrow context. There are also concerns that reliance on XML vocabularies can create new difficulties and protocols based on XML, such as SOAP, can just shift the proprietary activity to a new layer (as well as breaking one of the fundamental models that made the Web so successful). Such issues make formulating good policy and strategy even harder.

5. A strategic view

All the activities and events described above may help to convince you that OSS needs to be added to your “must think about that” list, as if you need another thing to worry about, but may not help you decide what strategic view to take of the subject or what you should put in the information strategy for your own organisation. What are the real strategic issues?

5.1 Software reuse

On the question of re-using software created by others, there are clear benefits but also clear obstacles and risks. Object-oriented programming may have advantages for programming teams during software development but the often-promoted benefit of creating re-usable software has failed to materialise on a large scale. Nowhere is this more so than in distributed applications. DCOM may have been used to develop a lot of programs but, as a mechanism for creating re-usable components, it has been far less successful. The Open Source alternative, CORBA, has made even less impact. Complexity of reusing the components, each with their own methods and information formats, is the main problem. Indeed, some authors are now questioning the usefulness and scalability of the whole RPC model which, in many ways, runs contrary to the Web model. In other words, it is still best suited to the closed network, proprietary client-server systems for which it was developed; despite it having been “Web-ified”, it doesn’t scale to the Internet. VPNs may give it somewhere to go, but success on the scale of the Internet is unlikely.

Yet, the best route to creating re-usable software is clear. You only have look at what kind of software is widely re-used now:

- Complete, but highly configurable, applications, e.g. Apache, Squid, Zope, MySQL.
- Complete programming languages, e.g. Perl, Python, PHP.
- Ready-to-use add-ons that leverage the use of these applications and languages, e.g. Apache or Perl modules (such as CGI.pm, Template Toolkit or XML::Parser) and CGI or PHP libraries.
- Good documentation available.

Actually, these are also the rules for creating good commercial software. So, if you want to create re-usable software, you need to create something that can be re-used easily and is sufficiently generic to be applied in applications ranging from slightly different to very different from the circumstances in which the software was developed. Public administrations are all different. Even parliaments, which appear very similar, are not all exactly the same.

Thus, a proven application which can be customised to local needs and language is clearly going to be more easily reused than an inflexible, special-purpose application or a hard to understand and apply component library.

5.2 Proprietary lock-in

On the question of proprietary lock-in and public service information, a move to open, non-proprietary information formats is clearly desirable, for many reasons, especially ease of information reuse, both inside and outside the organisation. XML is the leading contender but implementing an XML-based content strategy on an enterprise-wide basis is definitely a non-trivial task. However, it is also a great opportunity for reconsidering your information and reviewing your information flows. A new Internet Draft “Guidelines for the use of XML within IETF protocols”¹⁴ is an important new contribution to assist with the deployment of XML and, although there remains much work to do before it can move forward to a consensus document, it is already a very useful reference for anyone considering the design of an XML-based architecture.

Migrating and cross-training users away from their familiar Microsoft desktop applications are other serious issues, especially at staff grades where proficiency with particular applications is considered a job qualification and pay issue. However, there are ways to use those familiar applications with non-proprietary formats – Office XP goes a long way to making the use of XML possible, not just for import and export, but as a working format, especially in MS Word.

Once the decision to move away from the proprietary file formats has been taken, then the need for all staff to use the same proprietary application disappears. Higher grade staff can instead be offered tools that are function or event based, which better suits their role. Web-oriented front ends can create and modify information that is then processed to create an XML format that can be used with Office XP applications to create final form documents or reports.

The automatic generation and processing of “Word XHTML” documents is not only possible, but has numerous advantages which outweigh the loss of some advanced document features that arises by switching from Word format to Web format. Familiar Word “Styles” can be used to create decidedly unfamiliar XML structures; most users find adapting to structure-based XML editors very difficult conceptually – to them, structure is layout. Providing a scheme that allows users to continue to think layout, but actually generate structure, is clearly a “win-win” scenario.

This approach to document creation, because it creates documents that can be processed automatically (without the need for Microsoft Word), can also help to reduce another kind of lock-in, but one that is nothing to do with Microsoft: the tyranny of the paper-based layout. Too many documents are created only with the paper layout and appearance in mind instead of looking on the paper form as one instance of a derivative product generated from a reusable information original. Webmasters who spend their working lives trying to turn documents formatted with tabs, spaces and hard returns, from authors who refer to page numbers and insert numerous footnotes, into acceptable documents for the Web may well appreciate improvements here.

5.3 Security

Security is, of course, an issue that stretches far beyond the scope of this paper and is as much about people and their behaviour as it is about software. However, some points are hard to refute:

- Every security weakness in a software monoculture is the equivalent of a single point of failure.
- Security through obscurity demonstrably does not work for ICT systems, especially for networked systems – TCP/IP is fully public, yet most Internet attacks are on the proprietary applications, not on the open network protocols.
- Software whose source code is open to peer review and scrutiny has a far better chance of achieving lower error rates than software developed using the proprietary model.

5.4 Recommendations

In the same way that your information strategy now needs to take account of XML, it should also take into account the possibilities offered by OSS and indicate where it should and should not be applied. If you are developing special software to meet a particular need in your parliament, would you be willing to make it available for reuse? More importantly, would you be prepared to undertake or fund the additional work that would be required to make the software easily reusable (e.g. internationalisation and documentation)? Many of you may be using some OSS already; if the policy shift in favour of OSS gathers pace, all of you may have to consider this sooner than you think, and have answers ready. What OSS will you use and where and how?

6. How to choose OSS

Apart from the Linux operating system, one look at the SourceForge or FreshMeat sites will reveal the astonishing amount of free software around and the rate at which it is growing. However, much of it is destined for obscurity, being too specialised or just badly written. How, then, to choose OSS?

The principles set out above for developing OSS for reuse are, of course, also the same ones to apply when choosing software to reuse. One area where open source offerings provide something commercial software usually does not, is in applications that build on multiple existing popular applications to deliver new functionality, e.g. the content management systems AxKit and XML Comma. The number of Web applications built on top of Perl, PHP and MySQL runs into many hundreds.

Two additional factors in choosing OSS are:

- The software is widely used, indicating usefulness, quality, performance, etc.
- The existence of a thriving user community, e.g. a web site, a discussion group or “Wiki”, or a mailing list, indicating software that is under constant maintenance and development.

In terms of wide usage, clear OSS stars are Linux itself (i.e. the operating system), Perl, Apache, Squid, Cocoon, Axkit, Zope and the many modules and libraries available for them. Many of the “star” programs are supplied, almost ready-to-use, with the pre-packaged Linux distributions, such as Red Hat and SuSE. Red Hat is the market leader, but SuSE seems to have the edge in European localisation.

As for communities, librarians are a very good example of an active community and, as argued elsewhere¹⁵, have a new and special role to play in electronic information creation and management as metadata authors. There have also been some significant recent OSS library products developed, such as Koha¹⁶, a library catalogue system, and Greenstone¹⁷, a digital library creation suite developed in conjunction with UNESCO and the Human Info NGO, both developed in New Zealand, and the iVia¹⁸ Internet portal and virtual library, developed at the University of California.

Another specialised area, but one that is very relevant to parliaments, is Content Management Systems, with a many active communities of Web site managers. It is also an area where the commercial offerings tend to be very expensive but there is a wide choice of OSS offerings, such as AxKit, XML Comma, Bricolage, Midgard, Zope, PHPNuke, PostNuke, Cocoon and Slash. Given that all content management systems require extensive customisation to suit particular needs, the OSS packages can compete effectively with the commercial offerings.

OSS can provide the operating system and mature applications to meet many requirements as well as programming languages and development aids with which to create your special purpose systems. Deciding where to apply them for best effect, and in support of your information strategy, is not as easy as following the seemingly safe commercial herd decision. “Just buy Microsoft” may be the new “Just buy IBM”, i.e. no IT Director ever got fired for doing it, but there is a difference. When it was true for IBM, they were supplying “closed” mainframe systems with very limited external connectivity; they controlled just about every aspect of the system. The Internet has changed all that; connectivity is pervasive and multi-vendor. Microsoft came late to the Internet and, many argue, they still don’t do it very well, especially on the server. So, where to make best use of the strengths of OSS?

7. Where and how to apply OSS

- **Start with your Web servers.** This is the area where OSS is strongest. Linux, Apache, MySQL, Perl, etc., will meet all but the most demanding applications, both for Internet and Intranet applications, and provide a stable base platform for other applications.
- **Examine your application-specific network service platforms.** Proxies, caches, mail transporters and firewalls (“Internet Appliances”) are also strong OSS areas. Caches are not just a way of reducing traffic on your link to the Internet; they are also a cheap, yet frequently overlooked, method for leveraging the performance of your Intranet Web servers. Cache pre-loading is one of the key techniques of Compaq’s “Zero Latency Architecture”. The URI Web model, of course, is what makes this possible; you can’t preload a cache with SOAP requests.
- **Reconsider your enterprise network architecture.** Your NT/XP servers are probably being used just for enterprise login and file and print sharing. SAMBA can be used to provide similar file sharing functionality, but file sharing by drive or folder mapping is an old-fashioned technique that doesn’t scale well, becoming harder to navigate and search as it grows. Sharing user-created folder structures is a recipe for confusion because there is no consistent naming or organising scheme; instead, each user’s personal mental information model is exposed for everyone else to try to comprehend and navigate. Consider specialised, Web-based file server

resources that can provide better organisation and retrieval and reduce the dependency on proprietary file sharing methods. SAMBA also does print sharing and most network printers are also capable of being their own print servers; clients can speak to them directly. Enterprise-wide login is one of the most pervasive proprietary dependencies and one of the hardest to eliminate - OSS software has a long way to go in this area. Similar problems arise with remote configuration management.

- **Play to the strengths of the Web and its ways.** The original WWW design was incredibly simple, yet it grew to become the largest and most successful network in history. HTML was, and still is, a notoriously bad markup language. HTTP is a protocol so simple as to be almost obstructive. What made the Web a success was the generalised naming and addressing model: URIs. This simple, yet powerful, technique of named resources allowed the Web to grow at exponential speed because the resource that lies behind the URI can be anything – no pre-agreement required. An abstract model of what the Web grew to become (modelled afterwards, not before!) was defined by Roy Fielding in his Ph.D. dissertation¹⁹ and his paper based on it “Principled Design of the Modern Web Architecture”²⁰, which has become a seminal document in the debate over Web evolution. He also coined the term “REST (Representational State Transfer)” to describe his model of the Web. The REST model is what has made the Web so successful – it’s not a panacea but why take the risk of deviating from it if you don’t have to?
- **Reduce your dependence on protocols that break the Web model.** SOAP is no longer simple and breaks the REST model, because the same URI can have different operations embedded in the SOAP message. Unless there is a very good reason to use it, or other RPC protocols, avoid them. SOAP may be XML over HTTP but that doesn’t make it magic; it’s still really just DCOM for the Internet, with its own internal addressing scheme and data model. Rethinking object methods as addressable resources can be hard but can usually be achieved.
- **Reduce your dependency on proprietary content formats.** Try using the Word XHTML content format. If users keep forgetting to “Save as Web Page”, remember you can create the empty document for them via a Web server and, by using a simple form to request a new “blank” document, you can automatically fill in the document level metadata needed to identify and process that document later by using the Word “custom properties” feature. You will lose some more esoteric Word features, but the benefits can far outweigh the disadvantages.
- **Migrate your legacy document collections.** How many of you have large collections of documents in various versions of Word or, even, WordPerfect or earlier formats? You need to migrate these, preferably to a non-proprietary format, while there is still a window of opportunity to do so, and then look to OSS to maintain them thereafter. The same is true of SGML collections, because most SGML parsers are proprietary and unlikely to be maintained for much longer.

8. Conclusions

- OSS isn't new – it has been around for a long time. Unix users are no strangers to it; Unix, the Internet and OSS grew up together. Windows users, on the other hand, are used to either closed commercial software or free, but also closed (“shareware”), software; OSS is new to most of them and its development model may seem strange and unsustainable. However, history has shown otherwise. A lot of very good software has been developed this way.
- Public procurement is obliged to seek best value for money; a market without competition is very unlikely to deliver that. But, public policy shift towards OSS is less about money and more about reducing exposure to supplier lock-in and exploitation, reducing the risks arising from a closed, software monoculture and maximising value for money by reusing software where possible. The opportunities for open scrutiny of the software used to interact with citizens for e-government and e-voting is another great benefit that policy makers need to consider. Security through obscurity doesn't work.
- The large software companies may complain bitterly about any new public policy favouring OSS but they largely have themselves to blame. The marketing and accounting practices of many dot.com era companies have created a mood of deep cynicism around the world, in investors and customers; both feel exploited and angry. That cynicism and anger has now attracted the attention of politicians, who are in the mood for change. Repairing the damage is going to take a long time. Trying to crush Open Source will only make matters worse and further harden policy-makers against the software companies; remember the old political aphorism, “when in hole, stop digging”. Something else the software companies need to consider is the effect on their own staff. It's often said that Microsoft's most valuable assets walk out of the door every night. If OSS becomes the new exciting place to be, how many of those assets might choose not to walk back in?
- For you, as providers of ICT systems and services to your parliaments and, thereby, implementers of public procurement policy, you need to make a strategic assessment of where and how to best use OSS. Applying the most popular OSS, for Web and Intranet servers and “appliances”, is low risk and can yield very good results. Migrating your commercial-software dominated enterprise servers is harder and riskier but will be necessary if the policy to reduce supplier dependence is taken and is sensible anyway on the “reliability via diversity” principle. Migrating your client workstations is the hardest and riskiest of all but the difficulty and risk are reducing all the time as the OSS environments like KDE and Gnome evolve, in competition with each other.
- As for your strategies, look on OSS as an opportunity, underpinned by emerging public policy, to enhance the flexibility and reliability of your systems by diversifying and to maximise value for money by freely reusing software developed and proven by others.
- Also look on OSS as a way of being able to make a contribution back to your user community. Instead of just delivering papers to each other at conferences like this, think about one day delivering the software you have developed as well.

9. References

- ¹ Free / open source software actions in European Programmes www.cordis.lu/ist/ka-4/tesss/impl_free.htm.
- ² Open Source for E-Government Conference www.egovos.org.
- ³ eEurope 2005 Action Plan
www.europa.eu.int/information_society/eeurope/news_library/eeurope2005/index_en.htm.
- ⁴ POSS Pooling Open Source Software,
<http://europa.eu.int/ISPO/ida/jsps/index.jsp?fuseAction=showDocument&parent=highlights&documentID=550>.
- ⁵ Raymod, Eric “The Cathedral and the Bazaar”. www.tuxedo.org/~esr/writings/cathedral-bazaar/
- ⁶ PingID Digital Identity Infrastructure www.open-mag.com/features/Vol_24/pingid/pingid.htm.
- ⁷ SourceForge site: www.sourceforge.com
- ⁸ Stoltz, Mitch “The case for Government Promotion of Open Source Software; a NetAction White Paper.”
www.netaction.org/opensrc/oss-whole.html
- ⁹ InforWar / InfoSec Web site www.inforwar.com.
- ¹⁰ Mitre Corporation study of Open Source in government
http://www.mitre.org/support/papers/tech_papers_01/kenwood_software/index.shtml
- ¹¹ Letter from Juan Alberto González, General Manager, Microsoft, Peru, to Congressman Edgar Villanueva Nuñez. English translation: www.pimientolinux.com/peru2ms/alt_ms_tovillanueva.html
- ¹² Use of Free Software in Government Agencies; Bill presented to Peruvian Congress. English translation:
www.pimientolinux.com/peru2ms/free_software_bill.html
- ¹³ Reply from Edgar Villanueva Nuñez to Microsoft. English translation: www.gnu.org.pe/resmseng.html
- ¹⁴ “Guidelines for the use of XML within IETF Protocols”, Hollenbeck et al.
<http://search.ietf.org/internet-drafts/draft-hollenbeck-ietf-xml-guidelines-05.txt>
- ¹⁵ Hardie, Andrew “Librarian of the Future – Receptionist or Author?” Published in *Vesmir*, in Czech, November 2000; English version distributed at ECPRD ICT seminar, Paris, 2000.
- ¹⁶ Koha: Library Catalogue front end & OPAC, circulation tracker and acquisitions & budgeting system.
www.koha.org
- ¹⁷ Greenstone: software for building and distributing digital library collections. www.greenstone.org
- ¹⁸ iVia: Software for Internet Portal and Virtual Library Creation and Management
<http://infomine.ucr.edu/iVia/index.php>.
- ¹⁹ Fielding, Ph.D. dissertation http://www.ebuilt.com/fielding/pubs/dissertation/rest_arch_style.htm.
- ²⁰ Fielding & Naylor, “Principled Design of the modern Web Architecture”.
<http://www.cs.virginia.edu/~cs650/assignments/papers/p407-fielding.pdf>.