

An Investigation into the Open Source Potential of the Knowledge House Information System¹

Richard Mulberry, Paul Cranner, Kevin Ginty

Centre for Internet Technologies, University of Sunderland, Sunderland, UK

cc5rmu@student.sunderland.ac.uk, paul.cranner@sunderland.ac.uk,
kevin.ginty@sunderland.ac.uk

Abstract

Knowledge House (KH) is a collaborative service which helps companies gain access to the world class skills, expertise and specialist resources available within the five universities in North East England. To support the work undertaken across the network a bespoke web-based collaborative client/project management system is used – the Knowledge House Information System (KHIS).

The aim of this paper is twofold. Firstly to investigate current affairs for both open source (OS) software and open standards. To gain a practical understanding, the paper initially deduces what we mean by these terms. To best achieve a flavour of OS today a range of case studies were selected from literature that highlighted the major talking points. The role of open standards is then defined detailing the impact they have in the industry and their relationship to OS compared. The authors' view of the future for both OS and open standards is then stated. The paper then introduces KHIS and its underlying technology. The second aim of the paper is to use the findings presented to determine whether a vanilla version of the system is an appropriate candidate for release as OS software.

¹ Paper submitted as part of the Knowledge House Trial Project. The project was funded by JISC under the 'Facilitating Collaboration' stream of the BCE programme as part of the 'Trialling Collaborative Online Tools for BCE' project.

1. Introduction

Knowledge House (KH) is a collaborative service which helps companies gain access to the world class skills, expertise and specialist resources available within the five universities in North East England – Durham, Newcastle, Northumbria, Sunderland and Teesside. KH operates via a hub and spoke model with a small central headquarters and staff distributed at all five universities. To support the work undertaken across the network a bespoke, web-based collaborative client/project management system is used – the Knowledge House Information System (KHIS).

As part of an on-going JISC project funded under the ‘Facilitating Collaboration’ stream of the BCE programme as part of the ‘Trialling Collaborative Online Tools for BCE’ project Stewart.A (2010), KH has undertaken to explore the potential for releasing an Open Source (OS) vanilla version of KHIS.

The aim of this paper is twofold.

Firstly to investigate the impact of open standards and OS software on both private and public organisations. We begin by defining what we mean by both of these terms, attempting to set the scene and see if any relationships are evident. After gaining a firm understanding of these terms we will look into evaluating the strengths and weaknesses of OS software using real world case studies showcasing criteria such as costs, maintenance, support, training and security. We then analyse current trends and contemplate what the future holds for both OS and open standards.

Secondly we introduce the KHIS platform and consider the criteria presented in order to determine whether KHIS is a suitable candidate for release as OS software.

2. What is Open Source Software and Open Standards?

2.1 What do we mean by Open Source?

Asiri.S (2003) states it is very difficult to define the notion of OS as the term’s meaning differs from practitioner to practitioner. On explanation Asiri states three endeavour examples about OS which are: non profit/freedom of ownership, support for proprietary software, and the free software movement. Including ‘free software’ as part of the definition for OS is one example of a plethora that blurs the two. Both OS and free software share common characteristics grouped under Free/Open Source Software (FOSS), creating this blur encouraging them to become interchangeable when in fact they are distinguishable.

Free software encompasses the ideology of being “free” to manipulate anyway you see fit, with the focus having as little legal restriction as possible. The GNU identified by Waring.T & Maddocks.P (2005) state it should be thought as free speech to all (I can manipulate what I want) rather than a free beer (I can get this for free) for all, which is the common misconception.

The most definitive definition for OS software is thought to come from the Open Source Initiative (OSI) which provides a criterion of what OS software must fulfil (see **Appendix A**). Scacchi.W (2007) focuses on licensing (highlighted under criteria 8 & 9 of the definition) as the main distinctions between both. This definition is supported by OSS Watch which recognises OSI as the final authority in order to avoid confusion and debate, Wilson.J (2010). It is stated that free software licences are nearly always with the GNU General Public Licence (GPL), whilst OS can choose licences and

copyrights that are required to comply with the Open Source Definition and end users. It however does not give those who had involvement in its redistribution any private intellectual property rights Osterloh.M & Sandra Rota (2007). The GPL, if utilised by OS software, influences the model by defining the notion of copyleft as a means to guarantee free distribution Cerri.D & Fuggetta.A (2007). Therefore, it is not a concrete requirement of OS to be in economic terms 'free'. Essentially the two promote the same and have commonality that software can be shared; however they conflict by the angle from which they approach. The GNU highlight that Free Software is a social movement towards software "freedom", whilst OS is a methodology to progressively produce the best possible software by opening the source code to its community. Therefore it can be said that OS and free software agreed ideology is the distribution of open code to be the most powerful force to create the best possible software. The difference is how much freedom is allowed; the OS believe needing to restrict the level of openness of the software development model i.e. licensing may change the model completely, so long as it stays within the boundaries of the Open Source Definition.

2.2 Open Source Today

To emphasise the importance of OS in the current climate we must look at examples of its implementation. We only have to log onto OS community sites dedicated to OS projects such as SourceForge with their figures to realise that it is commanding a huge amount of interest. The website 'Inforworld' identifies the top 10 OS software initiatives ever, with many of the well known examples such as Linux Kernel, MySql and Apache Server making appearances (full list shown in **Appendix B**). There are many familiar absentees, most noticeably Mozilla, along with Oracle's recent purchase SUN's

OpenOffice (who on the 28th October 2009 reached their 100,000,000 download). The figures quoted and the bundle of examples suggests that the OS was prior to the economical crisis. Morelli.R & de Lanerolle T (2009) make reference to the current student age group who have grown up with this OS movement. "In general, they are deeply immersed in the open and sharing culture that the Web has become, without necessarily seeing the connections to the free software movement that, in many ways, initiated that culture." At the current time of writing, websites such as Facebook and Wikipedia are two prime examples of this sharing culture. This fast generation of up to the second pool of data has made accessibility even greater. This sharing culture however comes with its limitations, with Facebook causing riot in the private sector for distracting employees away from their work. Furthermore in the public sector Wikipedia is seen in the academic circles as an unreliable source of information that students should avoid. Whilst these are two of the more famous examples, many similar and parody sites have put questions on the reliability of information on the internet.

2.3 Open Standards

Amongst the literature when investigating Open Standards, authors such as Kesan.J & Shah.R (2007) and Tiemann.M (2006) make reference to the difficulty of a concrete definition. In accordance with Cerri.D & Fuggetta.A (2007) the main focus of open standards is to encourage interoperability as if adhering to the same standards means that all systems work with that particular element the same. This reduces the ability for any software user to be "locked in" to a particular piece of software which is also the focus of open/free software. Having the focus is one thing but to distinguish what an open standard is and how we can compare requires a definition.

Ceri.D & Fuggetta.A go on to highlight that the key issue to the confusion is not what the term ‘standard’ is – as this is widely used in many disciplines without much manipulation – it is the level of ‘openness’ that constitute an open standard. Whilst authors such as Kesan.J & Shah.R (2007) have identified de-facto and de jure standards Ceri & Fuggetta have managed to break these down a stage further. Excluding proprietary standards the four scales of openness are: Disclosed (owned by company but made available), Concerted (Consultation process but decision of participants made by company), “Open” Concerted (open participation by all), “Open” de jure (standards managed by official international and national standardisation). The most popular definition of Open Standards comes from Bruce Perens who distinguishes that open standards are not just a specification; it is the principles behind the standard and the practices offered that support the standard. For the six principles and practices identified by Perens.R (see **Appendix C**). This was later adapted by Krechmer.K (2006) to include others with Tiemann.M (2006) also including a four level distinction of what constitutes and OS. Indeed they are scattered around the literature but there is no defining line.

Kesan.J & Shan.R (2008) open standards making its bow in the U.S state of Massachusetts, “First, the standard is publicly available at a minimal cost. Second, no entity controls the standard, or the standard is licensed on “reasonable and non-discriminatory terms.” Third, the development process for creating the standard involves public participation.” Both of these definitions confirm that one of the principles of open standards is that there is little to no ownership and there is a great deal of public involvement, which makes it differ considerably from the private de-facto and de jure standards but

the parallels start to emerge between OS and open standards.

2.4 What is the Relationship?

Having defined both open standards and OS software we can see some quite evident overlaps. We have only to look at their name to see that they both precede with ‘open’ which means that the commonality shared between the two should give us an indication. What both open standards and software share is that they are firstly freely available to the general public and that there is little to no cost involved. Secondly the public have the opportunity to have a direct involvement or influence on the outcome, these elements in essence are what constitutes the term ‘open’. The debate applicable to both as we have seen is the levels of openness.

Dalziel.J (2003) makes a comparison between the two upon development stating that it appears more natural to develop the OS product first and then make it standard compliant. The two are considered similar but it is wrong to suggest that open standards are criteria to be used by OS software or that it must be compliant. Upon Sun developing Java with an OS licence it declared it would not become an open standard as this only applied to the implementation, with the standard not being revealed Ceri.D & Fuggetta.A (2007). As we have seen also because open standards are available to all it does mean that proprietary software vendors can implement open standards.

Therefore OS software does not have to utilize open standards but it is of great benefit if they are able to do so as OS software can miss interoperability completely. Open standards encourage this competition as it is likely to bring about the best possible system, which is the objective of OS in the first place. Therefore whilst neither has any legal requirement to each other the benefit from

the relationship allows both to achieve their primary goals.

3. Open Source in the Public Sector

3.1 Introduction

With the financial climate as it is today, we would expect OS software to play a prominent role in the public sector particularly governments who have a responsibility to invest the taxpayers money wisely. Governments are the largest purchasers of ICT systems and are therefore an extremely valid source of information when coming to evaluate what is used in industry and what isn't.

Waring.T & Maddocks.T (2005). Björgvinsson.T & Thorbergsson.H (2007) confirm that governments benefit as they are in a position to 'lock in' a whole nation by choosing certain products. They also identify that there is a significant issue with customising preparatory software, particularly to smaller countries. The benefit of OS software and the community aspect of sharing is that local developers can fuel rapid development of a satisfactory result. This obviously is dependent on the scale of customisation (which typically is of Western heritage); however this is still an issue for preparatory software developers. The nature of governments getting open software correct is it has a cascading effect throughout the entire nation which is its biggest advantage, however choosing incorrectly will have the opposite effect.

We will now have a look at a real world case study at a hospital and then look at individual case studies for each criteria identified in the introduction.

3.2 The Beaumont Hospital

Fitzgerald.B & Kenny.T (2003)

An example of a successful implementation is that of Beaumont Hospital in Ireland. It is cited that the three factors that instigated the move from proprietary based software (which was heavily subsidised) were principles, pragmatism and practicality. Both principles and pragmatisms focused mainly on the cost goal of OS software having very little face value, whilst the practicality element was focused on the exact likeness to some of the major proprietary software used, e.g. the move from Microsoft Excel to Star Office. In terms of cost the hospital is estimated to have saved up to thirteen million Euros over a span of five years with four point seven million saved in the first year of the switch. A further advantage to the hospital is the flexibility of the OS software as it can be distributed around the network easily. This was a problem for the hospital as when people are moved from one department to another they tend to feel to have entitlement to the proprietary software.

It is wrong to think that OS software is always compatible and has little or no cost. Beaumont Hospital found that there can be progressive costs in terms of support. In this particular case study, although twenty thousand pounds were spent on support by OpenApp (a software company) this was still a lot cheaper than implementing a proprietary version of the software on a whole. When purchasing software off the shelf it is common to see built-in support for a period of time which is usually infinite. However with OS, companies run the risk of being solely dependent on the community to support problems with a particular version or incompatibility. Even to a certain degree if we have a hot desk or development team the flexible nature of OS will still pull them away from their primary roles.

The Beaumont Hospital highlighted particularly having problems with spending a huge amount of their budget on

arranging training sessions which were constantly postponed or cancelled due to lack of attendance. In order to solve this problem they reduced their need for staff time with the introduction of an OS e-learning application called “caraloine”, which allowed many employees to choose when most convenient and also allow those who required training who were on the road on the companies behalf an opportunity to engage in the training. The saving given from hiring a professional to undertake the training a number of times has further benefitted the hospital.

3.3 Cost

Suffolk College – OpenSource Academy

Many public sectors rely on money from people, whether it is in the form of tax or charitable donations. Therefore, they have a responsibility to utilise it effectively. The first advantage that is perhaps seen as obvious when mentioning OS software is the huge financial benefit. Proprietary software will require you to purchase the software in some form; either purchasing the physical disk, or an agreed licence which can have conditions attached, usually the period of time and the number of machines. This can be a real benefit particularly for the larger organisations and the public sector as the numbers tend to multiply based on the number of machines that need the software. There are many examples of the savings that can be made in the education sector with the need for more access to computers and specialised software with an increasingly lower budget. Suffolk College are one example of an institution that utilised this when needing to upgrade their current email system, with Microsoft Express the prime target to distribute to over eighteen thousand people. With involvement of System Integration (Total Solution Computing) and IT service suppliers (GBdirect) they focused on a number of OS software (Spamassassin – spam filter, SuSE Linux Enterprise Server – Server

Software) that could work around many of the features offered by an all inclusive package such as Microsoft Exchange. If Microsoft Exchange had been offered on this scale it is estimated it would have cost in the range of one hundred and ten thousand pounds.

3.4 Maintenance

Koponen.T, Hotti.V (2005)

Part of the strength of OS software is the community that continues progressively to reduce the amount of defects and develop the code further. However, it can be said that one underlying flaw with opening up solutions and maintenance to the masses is how you manage it, and how or whether it is going to be solved. Koponen.T, Hotti.V (2005) introduces a case study on the similarities of the ‘standard’ framework for maintenance process stealing many elements from the ISO/IEC Maintenance process activities. Both Mozilla and Apache satisfy these activities with CVS as a version management system and Buzgilla as their defect management system. Any member of the community can submit a modification to the project with those accepted passed to the version control management system CVS. The final decision comes from the core set of programmers allocated to its development who do not only validate the integrity of the solution, but must access the quality if more than one solution is put forth.

3.5 Training

Open Office – Migrant Helpline

Training is one of the essential costs throughout all organisations, particularly in the IT industry as the pace is especially vigorous. Migrant Helpline is a charitable organisation helping asylum seekers access support. Like any charitable organisation any saving that can be made is of great significance. Therefore their decision to move from a Microsoft Office suite to

Open Office had long projected savings by not having to pay for Microsoft upgrades, services and of course the licence itself. Even though the User Interface (UI) is of great similarity, it is found that even slight changes can throw employees. Therefore it was suggested that a portion of the saving would initially be used to bring in a consultant to manage the switchover which mainly focused on training the current employees. This is usually very expensive as not only do they have to deal with entire organisations worth of people but the training is usually specialised, requires full commitment for an unknown period of time and is well known to be of high value as knowledge is power. The consideration was weighed up against the overall cost of the proprietary licence, with the longer term saving preferred.

3.6 Security

Hoepman.J.H & Jacobs.B (2007)

Schryen.G & Kadura.R, (2009) drawing on work from Messmer, E. (2005) identify that comparisons between open/closed software are fraught with danger due to the amount of “religion” that exists between the two. It is important to deal in fact, therefore as what we have tried to do by looking at the majority of case studies and literature. The community nature of OS software does not have any strict eligibility of who is able to introduce solutions that potentially could have security loopholes that may stutter the process. Hoepman.J.H & Jacobs.B (2007) inform that developers of Linux kernel discovered a backdoor in what appeared a harmless error checking facility. With the huge inconveniences of computer systems it is quite understandable why both public/private organisations are hesitant towards OS. Hoepman.J.H & Jacobs.B (2007) provides a counter argument towards OS stating that eventually even closed source software will eventually leak its way onto the internet, with a specific example being the Windows NT operating system.

Unfortunately due to the closed model it is far less responsive to these problems with the development time of patches/fixes being months later. Proprietary software companies ultimately also have different end goals to the OS environment, with a limited workforce; most of the maintenance is placed in parallel with development of new software. There is therefore a risk that companies stop supporting, which requires users to seek a newer version.

3.7 Support

Becta – Open Source Software in Schools

Becta’s introduced eight case studies on fifteen schools all of which have OS elements which have three outline aims. These aims individually touch upon on the performance, cost and successful implementation respectively. The final results that were collected from these eight case studies were varied but some common themes towards compatibility and unfamiliarity appeared. Whilst many commented on the advantages to OS software mentioned in previous case studies, some came up with some surprising disadvantages. More than a few of the case studies indicated that there was no support from Microsoft to make some of its products available on OS platforms such as Linux. Furthermore it appears that Microsoft have developed their niche in academia as most institutes have a need for one or more of their products as the students are so accustomed to the look and feel of it. There was a common theme that because OS software appeared to be more stable and of a superior build quality there was less need for support. One institution following on from this point stated that OSS did seem more reliable and because of their excellent support from people around them with expertise they received excellent support. Of course this is not applicable to all and if this knowledge is not in house then it may end up quite

expensive. Usually in terms of Microsoft and other proprietary software a support service is built into the cost of the software.

4. Proprietary Fights Back

4.1 Benefits of Proprietary

Microsoft – Newham

The case studies looked at so far have all been from selected businesses who have attempted to use OS, so based on the evidence of the case studies analysed thus far why is proprietary software needed? Leading the fight for preparatory systems Microsoft amongst others can justify in their own case studies that they can provide the many advantages identified in OS and more. We will now look at a case study from Microsoft implementing a new ICT system in Newman Council in the United Kingdom. In similar fashion to many other companies their aims were to reduce cost, provide business advantage and establish a long term solution.

It is hard to imagine how preparatory software can actually lower the costs of OS due to the vastly expensive fees and maintenance associated particularly with Microsoft products. However it was calculated that a Microsoft based system could prove up to 13.5% less expensive than the current system. The software, as generally expected, was more expensive however the cost of implementing the solution was cheaper with the OS solution only providing half the saving (7%). In addition, the migration costs for the Microsoft solution were estimated to be 68% lower than the switching costs of migrating to an OS platform.

Microsoft state that their solutions are more secure than any OS alternative and make emphasis towards their continuous investment in the Trustworthy Computing Initiative.

In a closing statement one of the main advantages of preparatory software is the need for companies to depend on their software. Implementing Microsoft based software doesn't even involve a great deal of risk as it has been extensively tested and stable, it is in essence how Microsoft made their name in the first place. Perhaps the biggest contrast in the final statement was the idea of consistency and predictability that companies seek, which would most likely justify a purchase of preparatory software. After all ICT is a very expensive field and whilst constantly changing there is a necessity to place the right foundations in order to realistic deal with change. While in the OS community we can bunny hop from one piece to another, this is fraught with danger due to retraining, compatibility and teething issues which ultimately take the most precious element in business, time.

5. The Future of Open Source and Standards

5.1 Open Source

Reflecting on the literature, many authors have their own theories on what the factors are that will influence whether OS Software will take over. Having looked at the likes of Iceland Thorbergsson.H, Bjorgvinsson.T & Valfells.A (2007) and particularly the recent UK Government Watson.T (2009) stance on open software, we know that both show OS is breaking down more and more barriers and is actually becoming the preference over proprietary software rather than being considered the cheapest option. Action point four of the document states where there is no significant price difference between the two, OS will be the preferred option due to its flexibility. With this extended freedom it provides a platform for OS to grow. It is likely therefore that more OS software will climb its way to the top of other domains mimicking those that have gone before them such as Apache.

Campbell-Kelly.M (2008) identifies that “the most prominent software development methods are those that seem to work best with the contemporary technological and economic constraints.” This debate is then applied towards the next software phenomenon, software as a service (SaaS) with both open and closed software having equally competent representation. A key benefit of a SaaS over closed software is that it is a diversion away from being locked in to a particular piece of software (which was one of its main disadvantages) to software that can be purchased on a pay while you play basis. Regardless of the particular application Trumba (2007) identifies five key benefits of SaaS including cost/time savings, diverting budget away from IT infrastructure and immediate access to the latest innovations. The OS model also has to adapt as identified by Asay.M (2007) that the benefit of switching no longer applies to OS software as once you leave the service there is a need to get the data in order to use it for another.

From the authors’ viewpoint, unless something drastic happens in the next five years it is predicted that growth will continue its current direction. It is expected, with governments becoming more amenable towards OS, for them to have new opportunities and hence get a few more OS products at the top of everyday used software. According to Appelbe.B (2003) the winner of the battle will be decided by two domains, word processing and operating systems. A critical battle will be between Open Office and Microsoft Office, with the emphasis on Open Office to see how far they can go to surpassing the facilities on offer by Microsoft and breaking down its commonality stigma.

In order for them to compete there is a possibility that licensing could be reduced to combat the open software moment initially. However the platform in which users choose to do everyday tasks is likely

to migrate and change and both will again vie again to be at the top of pile. As identified this is likely to be SaaS.

5.2 Open Standards

The keyword interoperability highlighted during our definition of open standards is likely to underpin the future building of software. Models with increased interoperability such as SaaS are rapidly evolving with Bill Gates describing the software “services wave” as the “next sea of change [that] is upon us” Trumba (2007). This has, and will, continue to accelerate the need for open standards as many consumers wish to take advantage of being able to perform multiple tasks of a similar nature across a number of different services within the same browser. A current example of this ‘grouping’ is Tweetdeck which provides simultaneous use of social networking sites such as Facebook and Twitter Orlando.D (2009). For Tweetdeck to achieve this it requires all sites to have an Open Application Programme Interface (API) that allows these individual services to be accessed and treated in the same manner.

At the time of writing the downloading of Tweetdeck is provided for both personal computers (PC) and iPhones in ‘app’ form. This trend has provided an extra complexity for software vendors to have a mobile phone ‘app’ representative, as many wish to access such social apps on the move. Due to mobile devices not containing open standards, software vendors have quickly realised the impracticality of developing such apps for different mobile devices i.e. iPhone and Blackberry. It is due to the lack of an open standard that, as more devices seek to take a slice of this ‘app’ trend, the number of scenarios to cater for will grow. The future of mobile apps is two-fold, migrate over to open standards or be superseded by a more amenable service already present via PC’s.

The popularity of multi running services does not just provide benefit of having all applications on a single screen. Social networking such as Tweetdeck and email are two of the strongest forms of communication over the internet with similar facilities but many still require more than one due to nature of usage, personal taste or compatibility with someone else. This in essence particularly fits the SaaS model as there is a need to see the event requiring focusing across the spectrum of accounts rather than the screen being religiously tied to one.

When the demand becomes strong and it is of benefit for individual services to be part of a more exposed popular ‘container’ this will have a domino effect on all services within that particular domain to have an interface that the container can use. Effectively this container will not only become the market leader instantly; it is in effect representing ‘open standards’. There is therefore a potential for open standards to fall into decline as SaaS grows.

6. Knowledge House Information System

6.1 Introduction

KHIS is a web-based CRM/project management system designed in-house to meet the specific requirements of universities and interactions between them. KHIS handles the full project life cycle and supports the sharing of project and client information across disparate teams to allow them to collaborate and be aware of other activities related to their work. KHIS is jointly owned by the five universities in North East England, each of which pays an annual subscription for the service. As part of the fee universities receive free technical support and benefit from continuous patches and functionality upgrades.

Written in proprietary Adobe ColdFusion and Microsoft SQL Server, KHIS was designed to satisfy four key requirements: security, extensibility, customisability and robustness. The system architecture is such that it could potentially be adapted for use by non-universities to manage client interactions.

6.2 Open Source Potential of KHIS

Perhaps the most obvious precursor to releasing an OS version of KHIS would be the need to gain approval from all five North East Universities. Clearly security of confidential data would be a prime concern. However, this could be avoided by removing university-specific customisations and releasing a vanilla version, whilst maintaining the university-specific KHIS as a standalone system.

Having identified five key OS criteria we will now consider the merits of KHIS against each.

Cost

KHIS is a web-based system and therefore a license is not needed for every computer which accesses it. However KHIS was built using proprietary software and server licenses are currently needed for SQL Server and ColdFusion. To avoid such costs the authors propose migrating to OS alternatives MySQL and Railo Open Source, Railo (2010).

It is anticipated that with tweaks to both code and SQL and thorough testing the OS alternative platform could support KHIS, although a thorough technical investigation is recommended to confirm this.

Hosting costs are another consideration. KHIS is hosted on the JANET network, JANET (2010) which is governed by strict rules prohibiting commercial use. Even if migrated to Railo and MySQL, unless hosted in-house an installation of KHIS

would incur hosting costs if used commercially.

Maintenance

KH utilises an established CVS and bug reporting system Trac (2010). Were an OS version of KHIS released a separate code branch could be created and made available to external developers. It would be necessary to maintain a separate branch as the core university version of KHIS is heavily customised to meet Higher Education (HE) requirements. Even taking into account the CVS merge functionality, maintaining separate branches could potentially lead to a divergence of code given the heavy customisation of the HE version of KHIS.

As recognised in 3.4 a potential difficulty is that externally submitted code updates may have to be reviewed by the KHIS Development Team which would inevitably detract from their day job.

Applying global code updates from the CVS could also corrupt local KHIS installations as such updates could potentially overwrite customisations.

Support

KH have an established KHIS support mechanism in the form of a wiki (user guide) and bug reporting system (tickets). The authors anticipate this system could be expanded to cater for additional users of an OS version of KHIS.

A potential pitfall is that technical knowledge of the system is limited to the core KHIS Development Team. Were KHIS to be released as OS it would be necessary to publish design and best practice guides, and technical specifications to the wiki.

As highlighted in 3.6 thought would also have to be given as to how to support an OS KHIS community as KH staff are

employed solely to respond to the needs of the subscribing universities.

Training

At the time of writing the KHIS Training Manual was in the process of being transferred from Microsoft Word to the KHIS wiki to ensure that users have access to the most up-to-date material. As any wiki facilitates, it could be opened up to allow an OS community to contribute content.

As recognised in 3.5 thought would have to be given as to how to train potential OS users. Currently only limited personnel have sufficient knowledge of KHIS to be able to train new users and their time is consumed by day-to-day duties. This places extra importance on the wiki containing comprehensive training material.

Security

A key design feature of KHIS is the separation of security logic from code, enabling developers to write new plug-ins without fear of compromising security. In this sense the code is suitable for release to external developers.

However the security logic is engrained in the database so external developers would have to exercise extreme caution if working with stored procedures. The authors recommend database updates be strictly regulated and reviewed by the core KHIS Development Team via the CVS.

KHIS Mobile

5.2 drew attention to the parallel development of OS applications for mobile devices.

During the JISC ‘Trialling Collaborative Online Tools for BCE’ project a demand emerged for a mobile version of KHIS.

However as reported, due to the lack of an open standard for mobiles considerable

overhead would be incurred in catering for the different types of Smartphone. Whilst possible, development of KHIS mobile apps should perhaps be considered as an independent OS project.

6.3 Software as a Service Potential of KHIS

Although beyond the remit of this paper, having touched upon SaaS in 5.1 it is worth briefly highlighting the potential candidature of KHIS for release as SaaS.

The core difference between an OS and SaaS KHIS would be a single centrally managed installation with multiple customised accounts, instead of multiple local installations. This model would negate several of the problems identified with releasing KHIS as OS.

Firstly a single centrally managed installation would require only one ColdFusion license and one SQL Server license plus hosting fee.

Secondly such a setup would be easier to manage and remove the risk of code diversions from multiple customised installations.

Thirdly patches and new functionality could be easily applied and immediately available to all users.

However on a cautionary note several drawbacks spring to the authors' mind.

Careful consideration would have to be given as to whether the service was licensed as free or subscription-based. Either way a Service Level Agreement (SLA) would be required placing more responsibility/liability on the host.

A centrally managed service with multiple accounts would also introduce data protection issues; it would be necessary to exercise extreme care when creating new accounts and applying permissions to ensure sensitive data was not exposed.

The authors recommend a more comprehensive study into KHIS and SaaS as the subject of a separate paper.

7. Conclusions

The findings of this paper illustrate the difficulty in determining what is meant by both open source and open standards and how they are commonly misinterpreted. The collection of case studies suggested that open source software has many other benefits to the user other than strictly financial with the benefits for open standards being guaranteeing interoperability.

The future of both open source and standards is dependent on its adaptation to new software models and its ability to take over several of the more popular software domains currently held by proprietary software vendors.

In considering the candidature of releasing KHIS as OS this paper has identified both positives and negatives.

The biggest positive is that the architecture of the system is such that it can be rapidly customised and extended. A training wiki, CVS and bug reporting system are also established.

The biggest drawbacks are time and cost. Technical and user knowledge of KHIS is limited to a small subset of people who are occupied by day-to-day duties. The release of an OS version would require a significant investment of time initially and licensing and hosting costs could deter interested parties.

Time and money permitting, however, it can be concluded that the potential exists to release an OS vanilla version of KHIS.

References

Journals

Appelbe.B (2003) The Future of Open Source Software, *Journal of Research and Practice in Information Technology*, Vol 35. No.4

Asiri.S (2003) Open Source Software, *ACM Special Interest Group on Computers and Society (SIGCAS)*, Vol.23. Issue 1

Ceri.D, Fuggetta.A (2007) Open standards, open formats, and open source, *The Journal of Systems and Software*, Vol 80. Issue 11. Pp 1930–1937

Dalziel.J (2003) Open Standards versus Open Source in E-Learning, *Educause Quarterley*, Vol 4. Pp 4 - 7

Hoepman.J.H & Jacobs.B (2007) Increased security through open source, *Communications of the ACM*, Vol 50. Issue 1. Pp 79 - 83

Kesan.J & Shan.R (2008) Open Standards in Electronic Governance: Promises and Pitfalls, *ACM International Conference Proceeding Series*, Vol. 351. Pp 179-182

Koponen.T, Hotti.V (2005) Open source software maintenance process framework. *Proceeding of the fifth workshop on Open Source software engineering*, Pp 1-5

Krechmer.K (2006) Open Standards Requirements *the International Journal of IT Standards and Standardization Research*, Vol 4. No. 1

Morelli.R & de Lanerolle.T (2009) FOSS 101: Engaging Introductory Students in the Open Source Movement. *ACM SIGCSE Bulletin*. Vol 41, Issue 1, Pp 311-315

Osterloh.M & Rota.S (2007) Open Source Software Development – Just Another Case of Collection Invention, Vol 36, pp 157 – 171

Scacchi.W (2007) Free/Open Source Software Development: Recent Research Results and Emerging Opportunities, *The 6th Joint Meeting on European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering: companion papers*. pp 459 - 468

Schryen.G & Kadura.R, (2009) Open source vs. closed source software: towards measuring security, *Proceedings of the 2009 ACM symposium on Applied Computing*. Pp 2016 - 2023

Throrbergesson.H, Bjorgvinsson.T & Valfellis.A (2007) Economic Benefits of Free Open Source Software In Electronic Governance *ACM International Conference Proceeding Series*. Vol 232.

Tiemann.M (2006) An objective definition of open standards *Computer Standards & Interfaces* Vol 28, pp 495-507

Waring.T & Maddocks.P (2005) Open Source Software implementation in the UK public sector: Evidence from the field and implications for the future *International Journal of Information Management* Vol 25, pp 411–428

Case Studies

British Educational Communication and Technology Agency (2005) *Open Source Software in Schools – A case study report*. Retrieved 5th May 2010 from: publications.becta.org.uk/download.cfm?resID=25908

Campbell-Kelly.M (2008) Will The Future Be Open Source? – Tracing the course of influential computing development and consider possible paths to new paradigm. *Communications of the ACM*, Vol 51. Issue 10. Pp 21-23

Fitzgerald.B & Kenny.T (2003) *Open Source Software can improve the Health of the bank balance – The Beaumont Hospital Experience*. Retrieved 5th May 2010 from: www.netproject.com/docs/Beaumont.pdf

Microsoft (2004) *London Borough of Newham Chooses Microsoft Solution Over Open Source as Best Overall Value Options*. Retrieved 9th May 2010 from: www.egovmonitor.com/reports/rep9455.doc

Open Source Academy. *Case Study regarding the project to install a bespoke email solution at Suffolk (affiliated to the University of East Anglia)*. Retrieved 6th May 2010 from: www.opensourceacademy.org.uk/solutions/casestudies/suffolk-college/file

Open Office – *OpenOffice.org 1 software meets finance company’s needs for Microsoft Office compatibility and cuts costs*. Retrieved 7th May 2010 from: www.openoffice.org/product/docs/skilldeal.pdf

Watson.T (2009) *Open Source, Open Standards and Re-Use: Government Action Plan*. Retrieved 9th May 2010 from: ww.cabinetoffice.gov.uk/media/141716/090224opensource.pdf

Websites

Down.K (201), Business and Community Engagement Programme. Retrieved 29th April 2010 from: www.jisc.ac.uk/whatwedo/programmes/bce.aspx

Perens.B, Open Standards Principles and Practice. Retrieved 3rd May 2010 from: perens.com/OpenStandards/Definition.html

Stallman.R (2007) Why Open Source misses the point of Free Software. Retrieved 5th May 2010 from: www.gnu.org/philosophy/open-source-misses-the-point.html

Asay.M (2007) Applying the principles of open source to Software as a Service. Retrieved 6th May 2010 from: news.cnet.com/8301-13505_3-9793559-16.html

Trumba (2007) Five Benefits of Software as a service. Retrieved 7th May 2010 from: www.trumba.com/connect/knowledgecenter/pdf/Saas_paper_WP-001.pdf

Orlando.D (2009) Top 10 tips for writing successful Software as a Service. Retrieved 5th January 2010 from: www.ibm.com/developerworks/opensource/library/os-cloud-saas/index.html

JANET (2010) JANET, the UK's education and research network. Retrieved 3rd June 2010 from: www.ja.net/services/index.html

Railo (2010) Railo Open Source. Retrieved 27th May 2010 from: www.getrailo.com/com/index.cfm/products/railo-open-source/

Stewart.A (2010) Collaborative Tools 4 BCE. Retrieved 3rd June 2010 from: collaborativetools4bce.jiscinvolve.org/wp/about-2/

Trac (2010) The Trac Project. Retrieved 9th June 2010 from: trac.edgewall.org/

Wilson.J (2010) OSS Watch – About Open Source. Retrieved 9th June 2010 from: www.oss-watch.ac.uk/about/faq.xml

Appendix A

Number	Criteria
1	Free Redistribution
2	Source Code
3	Derived Works
4	Integrity of The Author's Source Code
5	No Discrimination Against Persons or Groups
6	No Discrimination Against Fields Of Endeavour
7	Distribution Of Licence
8	Licence Must Be Specific to a Product
9	Licence Must Not Restrict Other Software
10	Licence Must Be Technology-Neutral

Source manipulated from Open Source Initiative (OSI). Retrieved 5th June 2010 from: www.opensource.org/docs/osd

Appendix B

1	Linux kernel
2	GNU utilities and compilers
3	Ubuntu
4	Three BSDs (FreeBSD, NetBSD, OpenBSD)
5	Samba
6	MySQL
7	BIND
8	Sendmail

9	OpenSSH and OpenSSL
10	Apache

Dineley.D, Mobley.H (2009) Top 10 Open Source Hall of Famers – InfoWorld’s top pick for the most indispensable open source software of all time. Retrieved 7th June 2010 from www.infoworld.com/d/open-source/top-10-open-source-hall-famers-848

Appendix C

	Principle	Practice
Availability	Open Standards are available for all to read and implement.	Open Standards are available for all to read and implement. Thus: <ol style="list-style-type: none"> 1. The best practice is for the standards text and reference implementation to be available for free download via the Internet. 2. Any software project should be able to afford a copy without undue hardship. The cost should not far exceed the cost of a college textbook. 3. Licenses attached to the standards documentation must not restrict any party from implementing the standard using any form of software license. 4. The best practice is for software reference platforms to be licensed in a way that is compatible with all forms of software licensing, both Free Software (Open Source) and proprietary. However, see <i>Predatory Practices</i> regarding license restrictions that may be appropriate for a software reference platform.
Maximum End-User Choice	Open Standards create a fair, competitive market for implementations of the standard. They do not lock the customer in to a particular vendor or group.	Open Standards create a fair, competitive market for implementations of the standard. Thus: <ol style="list-style-type: none"> 1. They must allow a wide range of implementations, by businesses, academia, and public projects. 2. They must support a range of pricing from very expensive to zero-price.
No Royalty	Open Standards are free for all to <i>implement</i> , with no royalty or fee. <i>Certification</i> of compliance by the standards organization may involve a fee.	Open Standards are free for all to <i>implement</i> , with no royalty or fee. <i>Certification</i> of compliance by the standards organization may have a fee. Thus: <ol style="list-style-type: none"> 1. Patents embedded in standards must be licensed royalty-free, with non-discriminatory terms. 2. Certification programs should include a low or zero cost self-certification, but may include higher-cost programs with enhanced branding.
No Discrimination	Open Standards and the organizations that administer them do not favour one	Open Standards and the organizations that administer them do not favour one

	<p>implementer over another for any reason other than the technical standards compliance of a vendor's implementation. Certification organizations must provide a path for low and zero-cost implementations to be validated, but may also provide enhanced certification services.</p>	<p>implementer over another for any reason other than the technical standards compliance of a vendor's implementation. Certification organizations must provide a path for low and zero-cost implementations to be validated, but may also provide enhanced certification services. Thus:</p> <ol style="list-style-type: none"> 1. A standards organization that wishes to support itself through certification branding should establish a premium track and a low-cost or zero-cost track. Generally, the premium track will provide a certification lab outside of the vendor's facility to verify a vendor's implementation and enhanced branding: a certification mark that indicates a greater certainty of verification and financial support of the standard. The low or zero-cost track would provide self-certification by the vendor and baseline branding.
Extension or Subset	<p>Implementations of Open Standards may be extended, or offered in subset form. However, certification organizations may decline to certify subset implementations, and may place requirements upon extensions (see <i>Predatory Practices</i>).</p>	<p>Implementations of Open Standards may be extended, or offered in subset form. However, certification organizations may decline to certify subset implementations, and may place requirements upon extensions (see <i>Predatory Practices</i>).</p>
Predatory Practices	<p>Open Standards may employ license terms that protect against subversion of the standard by <i>embrace-and-extend</i> tactics. The licenses attached to the standard may require the publication of reference information for extensions, and a license for all others to create, distribute, and sell software that is compatible with the extensions. An Open Standard may not otherwise prohibit extensions.</p>	<p>Open Standards may employ license terms that protect against subversion of the standard by <i>embrace-and-extend</i> tactics. The license may require the publication of reference information and a license to create and redistribute software compatible with the extensions. It may not prohibit the implementation of extensions.</p> <ol style="list-style-type: none"> 1. The standards organization may wish to apply an agreement similar to the <i>Sun Industry Standards Source License</i> to the standard documentation and its accompanying reference implementation. The Sun agreement requires publication of a reference implementation (not the actual commercial implementation) for any extensions to the standard. This makes it possible for a standards organization to actively preserve interoperability without stifling innovation.

B.Perens, Open Standards – Principles and Practice. Retrieved 12th May 2010 from perens.com/OpenStandards/Definition.html