Open Source Trends and Business Models

Guidelines for Key Irish Stakeholders

July 2006

Forfás

Table of Contents

1 Introduction and Background

Revenue estimates and forecasts based on Linux and other Open Source (OS) environments indicate that the OS environment is growing rapidly worldwide. Revenues for 2005 are expected to show a 48 percent increase over those of 2004 and a compound annual growth rate (CAGR) of 36 percent is forecast between 2004 and 2008. Major software developing companies such as IBM and HP that traditionally used the proprietary model of software development have increasingly adopted OS. Indeed, companies may find that embracing Open Source Software (OSS) is not an option, but an imperative, in order to maintain market share. So it is not necessarily a question of either/or, as both models continue to be operated successfully in the commercial world.

1.1 Objectives, Scope and Methodology

As Ireland is a key location for software development internationally, Irish Independent Software Vendors (ISVs) and Higher Education Institutions (HEIs) cannot afford to ignore the increased importance of the OS approach. The objective of this report is to provide guidelines to ISVs and HEIs on the adoption of open source models of developing software.

The objectives of this report are to:

- Provide an overview of those OS commercialisation models most suited to the Irish context; and
- Provide an overview of the legal implications resulting from the use of different licensing models in an OS context.

This document is intended to be read:

- As a primer for those new to open source, and it covers the basic issues for consideration; and
- As a source of more detailed considerations for those in advanced stages of open source software development.

To illustrate the many complex issues involved, a number of case studies of software firms currently using open source in a variety of ways is provided to illustrate the commercial avenues for open source.

It is not the purpose of this document to advise ISVs or HEIs on whether or not to adopt open source software development. The broad issues are too complex for this, and each organisation must make its own considered judgement based on its individual circumstances.

However, the document does identify the key considerations to be taken into account, and provides guidelines on when and how to consider each of these.

Table 1 on pages three and four summarises the key considerations for ISVs and HEIs.

The material in the report is based on a combination of desk based research and case study interviews carried out by IDC. Forfás established a steering group to oversee the project, comprising of representatives from IDA Ireland, Enterprise Ireland (EI), Science Foundation Ireland (SFI), as well as industry and academic participants.

How to use this report

The purpose of this report is to provide guidelines to ISVs and HEIs on the adoption of open source software. It focuses on the key commercial considerations in adopting open source, and provides specific guidance to HEIs on how to commercialise open source software. Background information to the open source movement is also provided.

Several case studies of organisations commercialising software via open source are provided. These case studies are denoted by the use of grey boxes:

Case studies are denoted by grey boxes.

We also include some detailed technical information on commercial models and licensing for the more advanced reader. Such information is denoted by the use of double-lined boxes:

Technical details are denoted by double-lined boxes.

Table 1: Open Source – Key Considerations for ISVs and HEIs

Key questions for HEIs and ISVs	Considerations	Actions
Does code exist?	If this is a new project, then now is the time to consider commercialisation. It is much easier to build commercialisation plans into a project if this is done at the outset.	You need to consider your business model and licence regimes as soon as you can, preferably before code is created.
If code exists, was the code developed by you?	If code already exists but was developed by you, then it is safe to proceed with commercialisation considerations.	You need to consider your business model and licence regimes. You may have to retrofit code to meet licence and business model assumptions. All documentation should also be up-to-date.
Is the work derived from the wider open source community?	If the code was originated in the open source domain, then it is probably already subject to licence terms. Find out what these terms are. You must avoid "contaminating" your code with derived work under licence regimes incompatible with your business model.	You must establish the basis of these terms before proceeding. Existing licence terms may preclude you from using the business models you are planning to use. Can you ask the originator to supply a version of the code with a more appropriate licence?
Do you have a vision or strategy for commercialisation of open source software?	Open source commercialisation is perhaps one of the few areas where a strategy can be a bad idea. There can be no overarching framework for commercialising open source software. Each project needs its own terms of reference, especially if the work involves collaboration between HEIs and ISVs.	Establish terms of reference and a contract for each collaborative project. Do not attempt to create an all-encompassing contract for every project, since the project parameters (objective, timescales, etc) will vary, and collaboration will not be appropriate for all commercialisation efforts.
Are IP concepts and ownership concepts understood by developers and managers/supervisors?	Developers, especially (though not exclusively) in HEIs, may be unaware of the licensing constraints of open source software. Importantly, developers' managers and supervisors must also be familiar with the many issues that surround open source licensing and commercialisation.	All parties must become familiar with IP and licensing issues tied in to open source.
What is the primary goal of the project?	Is the primary goal to create IP, to commercialise IP, or to create open source software for wider community benefit?	Determine the overall goals for the project. EU-funded projects are typically well-defined, but have an overriding objective to benefit society. Commercialisation may be a side benefit. Although IP created using open source software can be commercialised using traditional licensing methods, consider whether or not open source is the most appropriate mechanism to realise your objectives.

Key questions for HEIs and ISVs	Considerations	Actions
Is the project a collaborative venture between HEIs and commercial organisations?	HEIs have a "publish or perish" mindset. While this is entirely appropriate in an academic context, it places some constraints when considering commercialising software. Academics may not know (or care about) the provenance of open source code that they are using on an internal project, but once a project is to be commercialised this issue is critical. ISVs, on the other hand, pursue a "demo or die" mentality, where the pressures to release code drive a compacted development and delivery schedule – typically 12 months. In many ways, open source projects compound the differences between ISVs and HEIs – they highlight the cultural differences in approach to software commercialisation, rather than inherent software development issues.	Decide whether collaboration is, in fact, the best approach for the project. This is more than an issue of funding. The aims of the project must benefit both parties, and must account for the differences in priorities of ISVs and HEIs. Any collaborative work between ISVs and HEIs must first resolve these differing positions. Importantly, these must be discussed early in the project set-up, so that all parties understand their responsibilities, commitments and timescales. Collaborative projects work best where all parties share the interest and potential benefit in the project. The value created through IP and its commercialisation could also be shared proportionate to the resources expended.

Source: IDC, 2006

2 Why Go Open Source?

2.1 A Primer in Open Source, Free Software and Proprietary Software

2.1.1 Introduction

Software Intellectual Property (IP) requires and produces two types of product:

- Source code suitable for compilation and running on computer systems; and
- Works and tools such as data formats, documentation or development frameworks.

IP of this sort is subject to copyright law. Through copyright law, the owner of the copyright can assign rights to other parties to use the IP under conditions set by the copyright owner. For commercially distributed software this is usually achieved through the electronic distribution of the compiled binary application (with or without the source code) subject to a software licence. This licence specifies the rights and duties of those using the code or compiled binary application.

2.1.2 Free software

Free software, or open source software, typically refers to software licensed under a licence that allows the licensee to examine, modify and distribute the code in binary or source code form *without* paying a royalty to the licensor. The Free Software Foundation (FSF) sets out the following principles for code to qualify as free software:

- The freedom to run the program for any purpose;
- The freedom to study and modify the program;
- The freedom to copy the program; and
- The freedom to redistribute modified versions of the program.

2.1.3 Proprietary software

By contrast, proprietary software usually requires payment of a royalty to the licensor, disallows examination of the source code and restricts or prohibits modification and distribution of the code. For example, Microsoft retains ownership of the source code and provides a licence to use a binary version of its code.

For the purposes of this report, the term open source is used to refer to IP licensed using a FSF compatible licence and the term proprietary to refer to code that is not compatible with these principles. We accept that there are different uses for these terms within the broad IT community[1].

2.1.4 What is open source?

Open source software refers to software that is created by a development community rather than a single vendor. It is typically programmed by volunteers from many organisations, and proponents of OSS claim that a broad group of programmers produces a more useful and more bug-free product. As discussed above, OSS differs from traditional proprietary software development in the sense that the intellectual property developed is released under an OS licence.

Revenue streams for software developers and companies operating in the OS environment are not based on royalty payments. This has resulted in the development of a number of alternative models which are being used effectively in the business world today to generate income.

2.2 Commercialisation and the Intellectual Property Portfolio

A software vendor's long-term viability depends on its ability to maximise returns from core IP assets. The licensing choices outlined in section 6 provide vendors with a path to translate intellectual property into revenue. However, many vendors are now discovering that overall revenue maximisation requires that they address the broader economic question of how to increase the entire IP portfolio value, rather than examine isolated situations of generating revenue from the IP associated with a single product or offering.

There are many approaches to managing an IP portfolio, such as a more traditional market focused approach, assessing new/existing product and/or new/existing market opportunities for diversification.

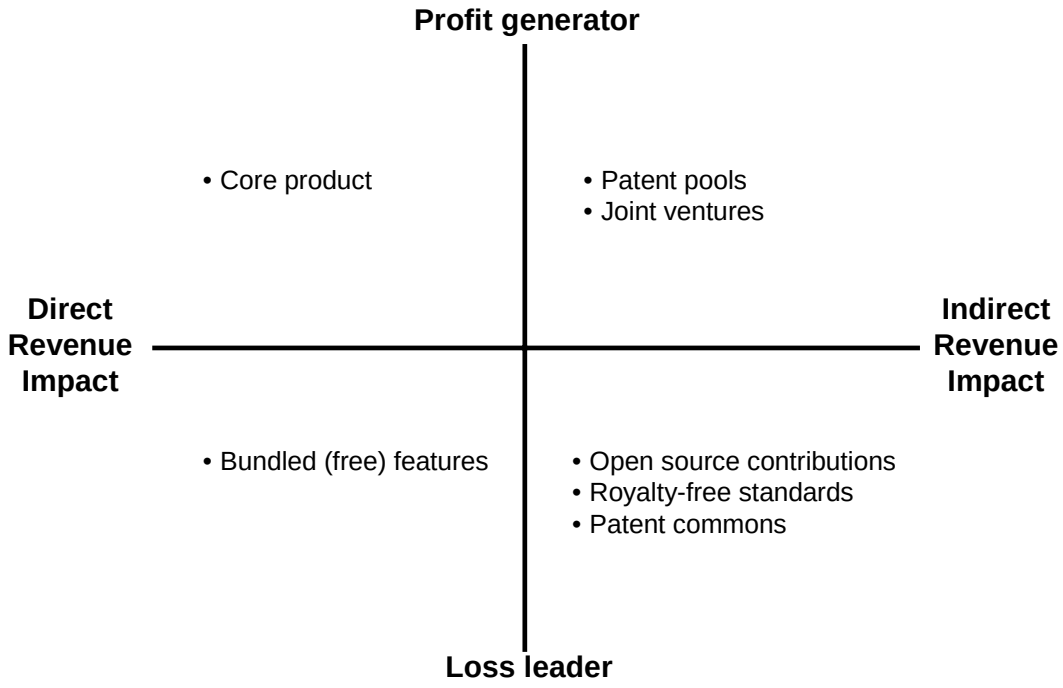
OS can be used as part of a wider IP portfolio management by ISVs. Taking a portfolio approach to IP, rather than looking to generating revenue on a product by product basis, confers a distinct advantage. This approach requires that a company looks at how it can increase market share, and ultimately profits, through mechanisms such as providing bundled (free) features and/or contributing to the OS community to build a customer base or create new markets, and not only at

[1] In this paper we focus on the licence considerations of open source, since this determines the commercialisation aspects for those owning rights to the code. However there are two other relevant and important senses in which the term open source is used:
- As a software development methodology;
- As a set of products.

how revenues are generated from each individual product. This approach also requires a different mind-set from the more traditional IP ownership/royalty based model and has been selectively adopted by firms such as IBM and Nokia.

A possible approach to managing the overall IP portfolio is shown in Figure 1 overleaf. In fact, IBM, the world's largest IT patenting firm in 2005, was very active in many facets of the IP portfolio described in this figure – such as actively supporting the open source community, launching a patent commons[2] and enabling the use of its patents in specified vertical market open standards.

Figure 1: Possible Approach to Managing the Overall IP Portfolio



Source: IDC, 2006

Although the open source business model is often portrayed as the bi-polar opposite of commercial software, there are many reasons a commercial vendor would (and should) engage in an open source

[2] IBM pledged that it would not pursue litigation with any company that infringed on the patents provided their products were open source.

business model to support its business objectives. Any new technology is hampered by the lack of a pre-existing installed customer base, and establishing de facto standards through open source is critical to overcome barriers to entry and the existence of natural monopolies (see the Iona case study on page 10 for further insight into this issue).

2.3 Why Go Open Source?

There are three main reasons why an ISV may choose to take a project open source:

- Developer motivation: developer interest in the project and support for free software aims;
- Development economics: reducing the cost of developing the product;
- Market adoption: increasing the utility of the product to customers.

2.3.1 Developer motivation

The urge to create something that satisfies an individual developer's intellectual hunger or natural curiosity is a primary reason why developers choose to work on any project. Open source harnesses this force by providing developers with the source code they require and there are many ways through which people with different skill sets can engage. Among the most critical are:

- Understanding the need and demand for particular features and how these can be integrated into the project;
- Development of new features and writing code to support these features;
- Patching the existing code base;
- Porting and testing hardware and middleware platforms;
- Testing patches and new features; and
- Documentation and support.

It is clear that there is wide scope for contribution from people with different skills and successful projects will need to engage each of these skill types. One of the aims of commercialising open source is to provide the resources necessary to build the full team needed for a complete project.

2.3.2 Development economics and cost

The open source approach to developing software is focused on reducing the cost of development in that the cost is spread across multiple parties and individuals. Since all contributors can use the results of the coding effort, they benefit from the work done by others.

While the most obvious benefit of this approach is a reduction in risk and cost to any single entity, other benefits include:

- Quality: Some studies have found that open source code has the same or higher quality – using measures such as memory allocation and de-allocation – compared to proprietary equivalents[3].
- Speed of development: Depending on the scale and model for development used, it is arguable that open source projects can evolve more quickly and introduce new features more quickly. This is based on the premise that the OS development model relies on mailing lists for code sharing, discussion and rapid iterative builds and tests of the code base and therefore results in a lower management overhead.
- Integration: One of the touted virtues of the proprietary model is that it allows the controlling company to fully integrate its products so that both new and existing code is integrated and interoperable with other code. It is also arguable that the open source model promotes integration because those contributing to the code have access to all of the components that need to be integrated.

Case study: IONA

Iona is a company firmly founded in the business of providing proprietary software and services to major enterprises. Founded in 1991, it was at the forefront of infrastructure software with its CORBA based Orbix product that enabled the interoperability of large-scale distributed computing systems. It continues to provide Orbix as a core, mature product and currently provides two versions, Orbix 3.3 and Orbix 6, to service the full spectrum of demand for CORBA based interoperability. The latest version (Orbix 6) is offered in three versions (Standard, Enterprise and Mainframe) to meet the varying development and deployment requirements of the corporate marketplace.

In addition to Orbix, it also offers Service Oriented Architecture (SOA) based integration software and services for enterprise IT organisations with multiple generations of business applications, technologies and architectures. This product, Artix, is an Enterprise Service Bus (ESB) that provides interoperability characteristics that form an excellent basis for a corporate SOA. Both of these products are licensed with a proprietary software licence and are targeted primarily at the larger enterprise.

Iona also has two complementary products that are aimed at smaller end user organisations. These products are:

- Orbacus, a source available CORBA 2.5 implementation, is targeted at organisations requiring rapid development and deployment with a small footprint; and
- Celtix, an open source Enterprise Service Bus, is a newly written Java implementation designed

[3] This was found, for example in a study published in late 2003 conducted by research firm Reasoning Inc. on MySQL and Tomcat, which is summarized in: Reasoning Study Reveals Code Quality of MySQL Open Source Database Ranks Higher than Commercial Equivalents and Reasoning Finds Tomcat Software Code Quality on Par with Commercial Code;

to drive the adoption of SOA in the enterprise.

For architects and developers both products have separate community based web sites to host the products with the Celtix site being located on the ObjectWeb Consortium web site and Orbacus hosted at www.orbacus.com. The ObjectWeb Consortium is an open source software community aimed at developing component-based middleware for large-scale distributed systems.

In addition to its role as lead sponsor of the Celtix open source product, Iona is also leading an open source SOA Tools Project (STP). The STP is a top-level project hosted by the Eclipse Foundation to provide best-of-breed, open source SOA tooling. Eclipse is an open source community whose projects are focused on providing a vendor-neutral open development platform and application frameworks for building software.

Although Celtix is not yet core to the overall commercial activities of Iona, it provides an excellent way of providing sophisticated capabilities to users in the SOA space, plus it provides an effective marketing tool to promote the expertise of Iona in areas that are critical to the IT departments of both major enterprises and small to medium sized businesses (SMEs).

In addition, Orbacus and Celtix provide a revenue and profit earning opportunity to Iona through the provision of service and support packages. Those that are currently available are:

Orbacus

There are four levels of support offered with Orbacus. Briefly, these provide an escalating set of services starting with a basic service providing support through a mailing list system through to a "Gold Service" providing:

- Unlimited access to a Technical Support Centre for 24 hours a day, 7 days a week;
- Four levels of response based on the severity of the problem;
- Access to the Iona knowledge base of solutions to technical problems;
- Discounts on major product updates and free minor updates;
- Individual application architecture reviews; and
- A personal technical account manager charged with supporting the contract holder.

Celtix

Iona offers Services Packages that provide:

- Training and e-Learning – Training in Celtix and its deployment that is available as:
    - Free online access to course materials and demos; and
    - On-site classroom training programs.

[4] A service-oriented architecture is the current best practice for architecting enterprise systems. It provides a collection of applications that communicate with each other, using each others "services". SOA facilitates cost savings because of reduced integration costs and re-use. It works well within a distributed systems architecture.

- Consulting – Consulting is offered in a variety of programmes that include:
  - Customised sessions based on the customer's specific requirements; and
  - A four-day Celtix FastStart program that combines Celtix training and consulting.
- Four levels of support to fit a typical deployment cycle. Briefly these are:
  - A free support pack giving email response plus access to an online knowledge base;
  - Per incident developer support;
  - Standard support with unlimited incident access and a one business day response time; and
  - Enterprise support with unlimited incident support and 24 X 7 access to a Technical Support Centre.

In summary, Iona is an established supplier of proprietary software and services that does not aim to be a pure "open source company". However, it is providing new open source products as an extension to its core commercial strategy. Although not in any way reliant on this strategic initiative, Iona will benefit in a number of ways from this market positioning. These benefits include:

- It will be seen and respected as an active member of the open source movement; deriving marketing advantages from this position;
- It will gain a source of revenue, benefiting from the economics of high unit volume with relatively low associated costs; and
- It will gain significant brand recognition and marketing impetus through open source products targeted at the low-end volume market.

Iona has invested heavily in its new Service Oriented Architecture (SOA)[4] products targeted at large enterprises. Through this strategy it will protect the lower end of the SOA market against penetration by competition that might distract Iona from its marketing efforts with major enterprises.

Iona illustrates very well the need to build a strategy that not only balances the direct costs of open source code against tangible, measurable benefits, but also takes into account the intangible benefits that are difficult or impossible to quantify.

2.3.3 Market adoption and revenue

The other side of the business model is driving increased market adoption of the product and turning this into revenue. Open source projects compete in sectors where there are well-established, existing proprietary competitors. In this kind of market an incumbent has advantages beyond those of having information on existing customers and brand awareness, including:

- External support: Depending on the vendor's approach to building its channel and support network there will be an existing base of expertise on which an organisation can draw;

- Trained staff: The business will already have staff that can maintain and build on the infrastructure;
- Dependencies: Depending on the nature of the product there is likely to be code linked to the installed product. Replacing the product will involve not only testing the product itself but also these dependencies; and
- Supplier relationships: The cost of switching suppliers can be considerable, both in direct and indirect costs. For example, costs incurred in changing management systems, or those less easily quantified costs involved in managing the relationship.

The classic marketing strategy for entering this kind of environment is to create a new category for the product in the mind of the customer and to price the product slightly below existing alternatives.

Open sourcing a project creates an opportunity to build market share more quickly through the following mechanisms:

- There are no direct costs for the customer in acquiring a version of the product; and
- The purchaser can have the confidence that the code base will survive any failure of a controlling owner.

It is also important for ISVs to realise that open sourcing a project does not in itself result in wide adoption. The practical steps to building a successful project that is widely adopted are beyond the scope of this report, but the following elements are important:

- Naming and packaging of the project in an appealing way to attract developers;
- Supporting individuals and organisations who want to join the project through documentation, encouragement from the existing community and open mailing lists;
- A clear statement of aims for the project so that potential developers can understand what the project is trying to achieve; and
- No competition with existing, successful projects.

Perhaps the central point regarding the economics of development and driving market adoption of a product is that producing code constitutes only a small part of the overall cost of bringing a product to market. Even for a proprietary software company that has to bear all the costs and manage all the development processes related to bringing the product to market, IDC research indicates that development cost tends to be between 10-30 percent of overall cost of bringing a product to market.

3 Guidelines in Considering Whether to Open Source

3.1 Key Considerations

In the past, small Irish software start-ups have faced challenges in the sale and marketing of their products. In many cases, the people who are adept at developing the software do not have the required skills to be equally adept at building a market for their products. This has been particularly true because of the small size of the indigenous Irish software market. Not only do Irish start-ups have to sell and market their products, but they have to do so in the international market at a very early stage. The open source model can provide such start-ups with the opportunity to find an alternative route to build a user base with the focus on revenue gained in other ways, such as from services.

Open source software companies do not tend to have finance available for formal marketing programs, so there tends to be an emphasis on viral marketing (involving, for example, the use of PR communications instead of advertising). Because of its small size, the Irish market is a good environment for building product usage through this marketing mechanism. There are a considerable number of publications focused on IT developments as well as organisations that publicise developments, providing a start-up with many avenues to build awareness of its products at an early stage.

There is considerable evidence from open source ISVs that services provision in the open source environment can be very different to what it would be in the proprietary environment. With the services based model, the primary focus in the open source environment is to build a community of users of whom a proportion will over time sign up as paying services customers. There is generally a time lag of some months between the time when people start using the products to when they become services customers.

In the early stages in particular, much of the service provision can be done online or can be automated, for example in the case of software updates. This means that a start-up can provide much initial support for customers with a very small number of personnel. If and when usage grows and users become customers demand for services will rise. However, the start-up should be able to deal with this as it grows, using usage figures to estimate future services demand and adding personnel as required.

3.2 Revenue, Margin and Cost Implications of Open Source

The use of open source software has grown strongly as an alternative to proprietary products. It has also developed in newer areas of IT infrastructure, such as domain name serving and web serving, where proprietary products have been slower to develop.

Over the last five years, in which the use of open source software has expanded, overall revenue from proprietary software and services has been growing. This infers that there is no inherent

reason why growth in open source software reduces revenue from proprietary software. Understanding the effect of open source on proprietary software margins is a more complex area because of the number of factors that affect both the sub-sector itself and the individual firms operating within it.

Sources of revenue will be different from those dominating the proprietary model. A number of commercialisation models for open source are discussed in section 5 of this report.

There is no simple answer to the question of whether or not open source software has an overall negative impact on software revenue and margins. For companies the situation is even more complex. A decision to use an open source licence rather than a proprietary licence offers both potential advantages and disadvantages which need to be considered in the overall context of an individual firm's strategy (see section 6).

3.3 Marketing Implications of Open Source

ISVs should consider potential advantages or disadvantages of open source in making marketing more effective, or in reducing the cost of marketing, in two distinct areas of activity:

- In marketing the project to other developers to encourage them to work on the project; and
- In marketing to end user companies and other ISVs to encourage them to pay for the commercial offering.

3.3.1 Marketing the project to other developers

ISVs should not fool themselves by thinking that they can open source their project and then developers will automatically work on the project. There are many open source projects that do not obtain widespread (or even any) support from other developers and ISVs. Such projects are either maintained by a dedicated individual or ISV or development withers away.

3.3.2 Marketing the product to end users or ISVs

ISVs should look at the broad distribution of software as a primary benefit of open source. Many of the challenges faced by Irish ISVs are those of scale. Lacking a large home market, Irish ISVs have to move overseas very early in their development without necessarily having built up a business strong enough to support this move. ISVs could promote their use of open source for increased marketing power in the following ways:

- Convince corporate purchasers that the product is backed by more than a small organisation that may be perceived as unable to serve the wider needs of these customers; and
- Demonstrate that the code in which the customer is investing will continue to exist even if the ISV fails.

3.3.3 Service Delivery Model

The software market is increasingly moving away from using traditional licensing models to generate revenue towards a service delivery model, which has at its centre:

- Support;
- Updates;
- Custom development; and
- Proprietary tools based on open source.

ISVs are likely to have concerns that their size and business model will not allow them to develop support offerings. However, the case studies given in this report show that this is precisely the model that many open source ISVs have successfully used to develop their business from the start-up stage.

3.4 Development Implications of Open Source

It is important to distinguish between developers using open source tools to carry out development work and those actually working on open source projects or considering taking their work open source.

3.4.1 Using open source development tools

Using open source development tools is well established practice with developers: tools such as Concurrent Versions System (CVS) are used to manage version control or tools such as Eclipse are used as a development environment. No specific issues arise in relation to using these tools - unless a developer is working on the source code for the tool or proposing to incorporate that code in their own code base. In these cases consideration of derived work licensing should be taken (see section 6.5).

3.4.2 Working on open source projects

For developers working on an open source project the situation is more complex and requires consideration of potential advantages and disadvantages. Developers are likely to come to open source from one of three backgrounds:

1. As an individual developer;
2. As part of an ISV organisation; and
3. As an owner of code.

As a freelance developer some of the advantages of open source are in:

- Being exposed to a wide variety of applications and coding styles;

- Having access to some types of code (such as middleware) that would otherwise not be available to the developer;
- For less experienced developers, being able to increase ones market value by showing code experience gained on open source projects.

As an owner of code or as a developer in an ISV the primary implications of open source are in:

- Code ownership - The developer needs to ensure that he does not introduce GPL (General Public Licence) code into his (or his employer's) application where he (or his employer) wants to redistribute this application under a licence other than the GPL; and
- Skills – The developer has the opportunity to gain skills and get recognition for those skills in a peer reviewed fashion.

3.5 Implications of Open Source for HEIs

This section examines the specific issues pertaining to the use of open source for commercialisation in HEIs (Higher Education Institutes). Note that all of the above considerations for ISVs apply equally to HEIs. HEIs are a special case, and require additional guidelines.

3.5.1 Typical HEI situation

Many HEIs around the world have ambitions to commercialise IP created under their auspices. The potential financial returns from commercialisation are significant, not to mention the academic kudos and consequential indirect funding opportunities (from grants, sponsored research etc.).

It is important to note, however, that historically the number of cases of successful commercialisation of academic IP is small. There are four main reasons why HEIs may find IP commercialisation difficult:

- Academic prowess, not commercial success, is the raison d'être of HEIs.
- HEIs have limited resources to invest in the capital expenditure required to support a commercialisation model.
- There is a lack of experience in HEIs, both of general business exposure and specific commercialisation skills.
- The relationship between HEIs and industry is sometimes difficult, primarily because of the difference in underlying objectives (creating knowledge versus creating wealth).

Many countries, including Ireland, are attempting to address these issues, through the creation of intermediary organisations that bridge the commercial and academic worlds. Scotland's Intermediary Technology Institutes (ITIs) are examples of this, modelled on similar agencies in Finland, Canada and in South East Asia.

The basic concept of these agencies is to channel public funding (e.g. from Scottish Enterprise) into the ITI. The ITI awards funding to selected academic projects, and provides liaison with industry organisations such as HP and IBM. Additional support in the form of premises, business model development and marketing is also available.

The gestation period of these agencies' ventures is long, typically three or more years. In addition, the ITIs retain a majority stake in IP ownership.

It is therefore generally the case that HEIs' success rate in commercialising software is low, despite considerable attempts to improve the situation.

HEIs and IP commercialisation – In more detail

Academic prowess, not commercial success, is the raison d'être of HEIs. Many HEIs receive funding from government and other sources based on the extent and quality of research. Those HEIs unable to secure such funding must implement outreach programmes, such as public training courses and services-for-fee (industrial design is a common example). But in such cases, any IP that is generated belongs to the commissioning agency, not the HEI.

HEIs have limited resources to invest in the capital expenditure required to support a commercialisation model. The network and hardware resources needed to sustain a commercial enterprise, as well as ongoing product development, demand substantial resources, typically beyond the means of the usual academic funding sources. Efforts must therefore be made to attract large investments from public and private sources.

There is a general lack of experience in HEIs, both of general business exposure and specific commercialisation skills. Even in universities with commercialisation departments, such as in the US, efforts can be constrained by a lack of understanding of what is required to develop software that can be commercialised. Consultants at IDC have substantial experience in assessing the likelihood of commercial success in software developed in HEIs and in enterprises. Few HEIs understand the difference between developing software that works and software that is productised. The amount of effort required in making software robust and usable, plus creating quality documentation and a support infrastructure, is usually underestimated.

The relationship between HEIs and industry can be strained, primarily because of the difference in underlying objectives (creating knowledge versus creating wealth). In IDC's experience, where an IT firm is interested in academic IP those responsible for creating the IP are typically hired by the firm, taking the IP and knowledge with them. Often when an entrepreneurial individual creates their own firm to commercialise IP, the IP goes to

[5] However, for publicly funded research in Ireland the HEI owns the IP. This is the case with Science Foundation Ireland, the Health Research Board, Enterprise Ireland and the Higher Education Authority.

the firm and is not retained by the HEI.[5] Indeed, in those instances where HEIs retain ownership of IP generated by its academic staff and students, it is found that entrepreneurship declines. Industry also moves at a faster speed than HEIs. For example, firms will look for a 12-month time-to-market, whereas HEI projects often align with postgraduate theses lasting three years.

4 General Commercialisation Considerations

4.1 Commercialisation Background and Scoping Issues

There is no single model for bringing open source software to market and to revenue. There is therefore considerable scope for innovation in all areas of commercialisation. However, it is also clear that certain factors are key for commercial success and this report outlines the options that are most likely to succeed in current and forecast market conditions.

For the purpose of this report software commercialisation is defined as the activity of building an ongoing stream of revenue and profit from either:

- An existing base of software code that has been written, but is not yet released outside the organisation that created the code; or
- A proposed programme of work to create code for commercial exploitation.

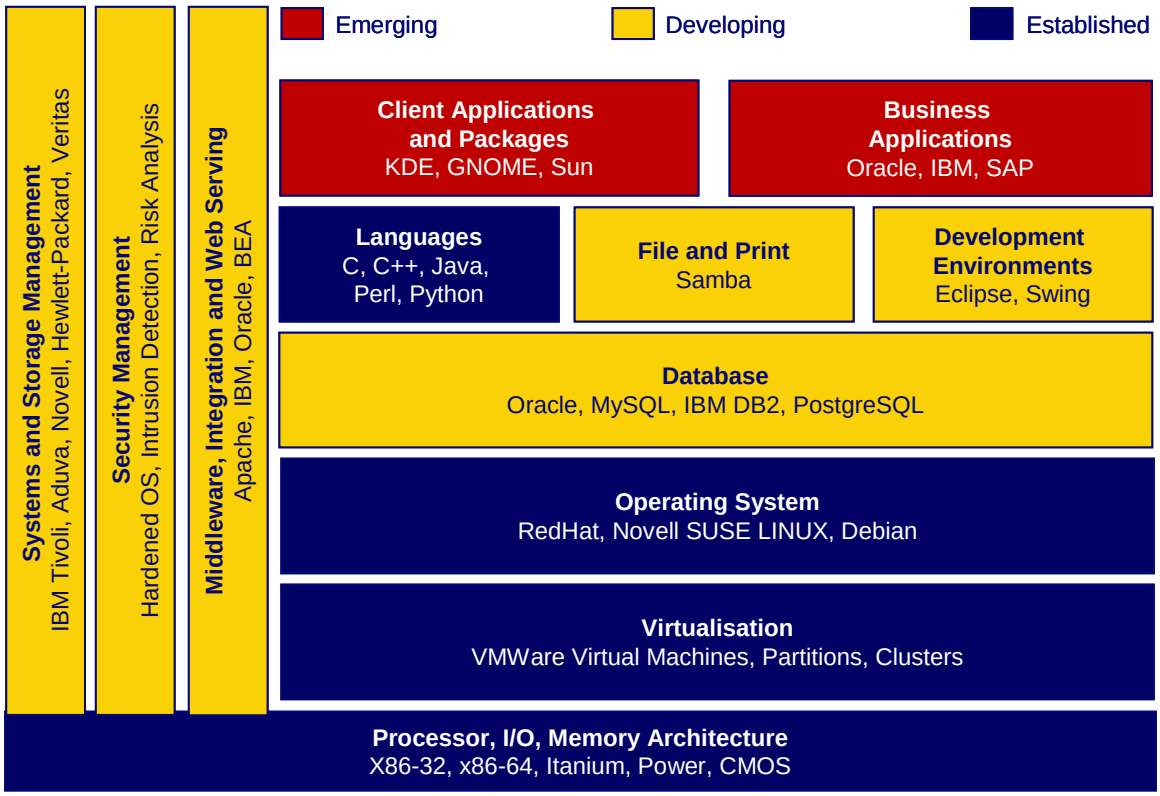
4.2 Open Source Development Opportunities

Development opportunities for Irish ISVs are apparent in a number of sectors within the overall software ecosystem. However, various sectors are in different stages of maturity and the more mature sectors offer less opportunity compared to those that are either:

- Developing as a sector - where there are open source products that have already gained acceptance in customer environments, with Original Equipment Manufacturers (OEMs) or with other ISVs;
- Emerging as a potentially viable sector - where open source adoption is currently more limited.

Using the notion of established, developing and emerging software development opportunities for open source, Figure 2 overleaf provides our assessment of the current state of the market.

Figure 2: Building the Stack - Development Opportunities



Source: IDC, 2006

Established sectors such as operating systems offer a reduced opportunity to Irish ISVs when compared with other available options.

In contrast, developing sectors offer a number of interesting opportunities to Irish ISVs, particularly in the areas of:

- Application life-cycle management;
- Application deployment software;
- System and network management software; and
- Security software.

Emerging sectors offer the most opportunity and serving the demand for business applications software is likely to provide the most significant potential for Irish ISVs. The case study of SugarCRM on page 23 provides an excellent example of a newly formed company exploiting this demand.

With regard to business applications software, research indicates that there will be increased demand across the board. Demand for vertical requirements will become an increasingly significant factor combining with the current demand for horizontal products. Despite the maturity of some business applications (e.g. CRM and SCM), future software demand is likely to be in the areas of:

- Enterprise resource management (ERM);
- Engineering applications/product life-cycle management (PLM);
- Supply chain management (SCM);
- Operations and manufacturing; and
- Customer relationship management (CRM).

4.3 Geographic and Vertical Sector Considerations

4.3.1 Geographic sectors

IDC's 2004 estimate for worldwide software revenue based on Linux and other open source environments indicates that there is little difference in terms of market adoption rates between countries. IDC does not currently forecast this demand based on geographic sector. However, its worldwide forecast for all software revenue indicates that countries outside North America, Western Europe and Asia/Pacific will grow the fastest with a 2004 –2009 CAGR of 10.6 percent. This is compared with:

- North America 6.2 percent
- Western Europe 5.8 percent
- Asia/Pacific 7.9 percent

Despite the faster growth rates forecast for other regions, we recommend that Irish ISVs initially concentrate on the Western European and North American markets. This recommendation is based on the similarity and familiarity of business practices in these areas and the relatively low cost of accessing these markets. However, if an ISV has already established a presence and contacts in one or more of the Far East countries, then these country options should also be considered.

4.3.2 Vertical sectors

Growth in total worldwide software spending across all business verticals from 2004 to 2009 is forecast to be six percent. The three fastest growing vertical sectors are:

- Healthcare 8.8 percent
- Government 7.1 percent
- Financial markets 6.7 percent

However, these minor differences notwithstanding, the increase in global spending on software is relatively uniform across vertical sectors with compound annual growth rates through to 2009 varying only between five and nine percent. No sectors show exceptional growth opportunities and IDC's view is that there is no increased propensity to adopt open source in any specific vertical sector. Accordingly, Irish ISVs should use their own expertise and specialisation to define their vertical sector concentration.

ISVs should note that the demand for vertical market specialisation is growing, as enterprises seek to get more from their IT suppliers. While there are no indicators to suggest particularly strong verticals, IDC recommend that ISVs concentrate on providing tailored offerings to one or two targeted vertical sectors. In other words: avoid generic horizontal applications.

4.3.3 Overriding purchasing drivers

Irish ISVs should always take into consideration three primary drivers with regard to the purchasing behaviour of user organisations worldwide. These drivers directly influence the success of any software development once the resultant software package is taken to market, be it through direct sales or through channel partners. These drivers are:

- User organisations buy software and services based on the derivation of business value, not based on the features associated with the particular piece of software or service.
- User organisations now consider the particular features of software licensing models as a source of business value (e.g. lower cost) and this will be an increasing area of interest over the next few years.
- Cost is a major factor when considering open source, but flexibility and return on investment (ROI and especially payback time) become the critical factors over the lifetime of a project.

In summary, suppliers should always sell their products and services based on business value.

Case study: SugarCRM

SugarCRM provides a good example of open source moving to line of business applications.

SugarCRM is a fairly new company providing customer relationship management (CRM) software to the enterprise and mid markets. It was founded in April 2004 and raised significant venture capital support of $26 million. It provides both OS and commercial versions of its CRM software product and boasts more than 600 customers for the commercial versions with 5,000 community developers and 125 business partners. Over 500,000 copies of the OS version of its products have been downloaded free from its web site.

SugarCRM offers one OS product, Sugar Open Source, which is free and contains basic CRM functionality such as campaigns, contacts, opportunities, accounts, cases, bugs, email, collaboration and dashboards. It has two commercial products, which are built as extensions to Sugar Open Source, offering access to the full source code:

- Sugar Professional – This product contains more advanced functionality such as reporting, user and team management, workflow and access control functionality; and
- Sugar Enterprise – This product is aimed at larger, more complex deployments, with features such as offline client synchronisation and support for multiple databases.

SugarCRM operates a mixed open source/commercial model as well as selling services. The core product on which all others are based continues to be developed by the SugarCRM staff and is available free under a version of the Mozilla Public Licence 1.1. Self support for this product is provided free of charge. Extensions, translations, and other modules are contributed by the SugarCRM community. The development environment is known as SugarForge and it currently hosts over 200 projects associated with the basic OS software.

SugarCRM generates revenue by charging for Sugar Professional and Sugar Enterprise and through the provision of a range of support services. The revenue generation for commercial products is accomplished through three routes:

- The products are made available to the customer in a hosted environment (Sugar On-Demand);
- The products are packaged as a turnkey solution providing both hardware and software (Sugar Cube); and
- The products are simply downloaded in a packaged format, but with the added advantage that the source code (written in PHP) is made available to the user. (Sugar On-Premise).

For an appropriate charge, customers can move between these deployments options depending on their needs. The basic premise for pricing of the products is to charge on a "per seat, per annum" basis i.e. the end user of the product is effectively paying a subscription for or renting the product from SugarCRM.

There are three sets of services offered for revenue generation. These are:

- Support Services – standard support is chargeable for the open source version of the product, but packaged with the subscription for the proprietary versions. Extended support is available at a further cost;
- Implementation Services – a range of some ten implementation service offerings are available from a one-day starter package to full installation and customisation; and
- Professional Services – a range of 12 services covering most things that a user would want to do as part of an overall CRM implementation initiative.

5 Commercialisation Models

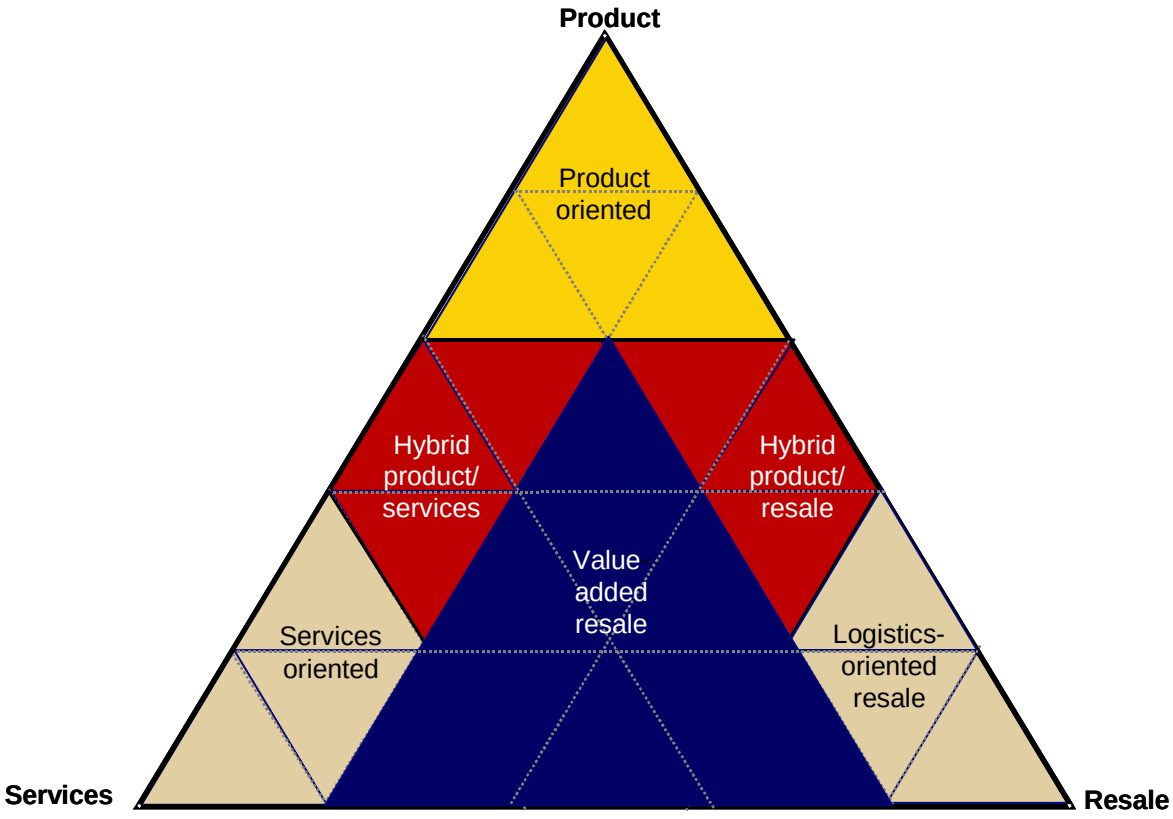
5.1 Introduction to Commercialising Software

There are a number of ways in which a company can make money out of activities related to software. Owners or prospective developers of software intellectual property can:

- Charge a royalty fee for use of the software product. This is the traditional proprietary software model where the customer pays for the right to use the application.
- Charge for services. These services include a broad range of activities that are discussed below.
- Charge a margin for reselling other companies' IP. As is normally the case, the margin the ISV can charge will tend to reflect the perceived value added by the ISV's other activities.

These are shown in Figure 3 below:

Figure 3: Commercialisation Models Using the IDC Partner Segmentation Model



Source: IDC, 2006

Companies or individuals considering open source should note the following:

Open source software companies (and many proprietary software companies) should consider moving their business model towards providing update and support services for their products and towards providing custom software development based on their products. Customers have a need for such support and the developer of the software has an obvious advantage over rivals in providing this support.

Revenue forecasting should be made easier through a move towards services models. The traditional software licensing model of taking single or multiple payments for a licence to use the company's software means that future revenue is both unknown once the payment period ends as well as cyclical, in that it is affected by upgrades and new release cycles. A move to support and subscription upgrade services allows for a more predictable and less cyclical revenue stream.

It is clear from the case studies in this report that revenue streams can be built on open source products and that ISVs are currently using a range of models to do this. However, ISVs considering open source should in the first instance investigate the potential for cost reduction offered by the open source model.

A number of commercialisation models are discussed in the following sections.

5.2 Commercialisation of Open Source

Although commercialisation is not an inherent goal of open source software, there is evidence of current OSS business models operating successfully. IDC believe that the following models offer the greatest scope for commercialisation:

- Support base commercialisation;
- Software as a service; and
- Proprietary development on open source.

The next three sections explore these high-level models in detail, as well as considering relevant sub-model variants.

5.3 Support-Based Commercialisation

This model does not necessarily involve ownership of the code itself, but instead relies on creating value for services around the project. Typical services offered by companies using this model are:

- Integration of an OS project into a larger code base and subsequent testing;
- Extensions to the OS code base to address specific customer requirements or to integrate the code within the customer's existing infrastructure;

- Provision of telephone, web, email and on-site support for the project at the customer's site;
- Supply of patches and updates; and
- Indemnification services against legal action arising from use of the product.

The most prominent company using this model is almost certain to be Red Hat, which has been at the forefront in the popularisation of open systems. A case history of Red Hat giving details of its operational characteristics is shown on page 27.

The need for support is a feature of almost all software, and within the OS context this is a good model for applications that are less integrated into other products and where features are customised to particular customer needs.

5.3.1 Advantages and disadvantages

The principal advantages of the Support-based model are:

- Large businesses typically require vendor support for their software. Hence, if this segment uses an OS product then there is likely to be a market for the associated support services;
- It provides a predictable and dependable revenue stream;
- The reliability of the revenue stream leads to opportunities for lower cost of borrowing;
- Subscription renewal rates for some vendors can be very high (greater than eighty five percent), reducing the overall marketing and sales effort required to generate repeat business; and
- It can be used for any type of software since all software is likely to need some form of support.

The chief disadvantages are:

- There is a feeling among customers that since the product is open source and plenty of information is openly available on popular projects, there should be no need to pay for support;
- It requires an existing installed base of customers to support; and
- Given that the product is open source, third parties can also provide support and can provide significant competition for the ISV. Smaller ISVs with popular projects could consider linking with third party support providers and entering into a revenue sharing arrangement.

Case study: Red Hat

Red Hat uses a support-based subscription model for its open source software. Customers pay for Red Hat Enterprise Linux through a maintenance/service and support subscription, while there is also a community-developed free product called Fedora (See later in this case study). The paid version is a "hardened, tested and certified version of the free product" that ensures stability, certification, scalability, and security.

A user subscription comprises of:

- A complete open source solution with thoroughly tested software that includes enhanced performance and security. The API/ABI stability guarantees of Red Hat Enterprise Linux result in it being a practical

platform for commercial application certification and deployment. There is also an extensive ISV and OEM certification ecosystem, which means that a customer can construct a complete IT solution (servers, storage, network, operating system, applications, management tools, security services, etc) that is fully supported by all the vendors, is commercially viable, and is able to deliver mission critical capabilities.

- Red Hat Network – a systems management platform providing tools to manage the systems on a user's network, making Linux deployable, scalable and manageable, and adding a greater level of privacy and security.
- Unlimited Access to Support – includes customer service calls to speak with Red Hat Certified Engineers, Support Guides, Knowledge base, Documentation, Hardware Compatibility List, Security and Errata and Product Support guides.

User support can be obtained via a one or three year subscription and is global 24X7 with centres in the US (east coast), the UK and Australia. A user can also get technical account management, developer support and premium developer support packages, as well as security and bug fixes available for the long haul. These services are provided at an additional cost to the basic service platform. Red Hat obtains revenue streams from:

- Subscriptions from Red Hat Enterprise Linux (RHEL) on a per system basis (prime profit driver);
- Subscriptions from additional layers of open source technology on a per system basis;
- System management services; and
- Support services.

Red Hat utilises the GNU General Public Licence and similar open source licences and considers that their most valuable intellectual property is its collection of trademarks.

Fedora

Red Hat has supported a free Linux distribution for over ten years. The Fedora Project is a Red Hat sponsored and community supported open source project for Fedora Core and other open source content projects. Fedora Core is an operating system and platform based on Linux, and is free for anyone to use, modify, distribute, or develop. Fedora Core is not tested and certified like Red Hat Enterprise Linux, but often development from the community on Fedora Core provides new innovation for features that may eventually make its way into other Red Hat products. The Fedora Project is "an openly-developed project designed by Red Hat, open for general participation, led by a meritocracy, following a set of project objectives." In essence it is Red Hat's roots; 2-3 releases per year, and freely available downloads.

Economic model

Support-based models generally provide for a revenue stream that for most commercial applications is fixed over a period of between one and three years. It differs from the more traditional royalty models that involve a single up-front payment and subsequent annual maintenance contracts.

Pricing strategy for the Support-based model has evolved and includes the following broad options:

- Per support incident;
- Per instance of the software installed;
- Per physical device, e.g. per desktop, per server, per server processor;
- Per user in the customer's organisation;
- Per number of accessing entities (e.g. database connections, web server); and
- Per customer's business performance (typically revenue).

The selection of pricing strategy is relevant in terms of costs of providing the support and in profit margin. Therefore, most support business models will have thresholds or tiers for support.

Major considerations in increasing profitability

Custom development work is a natural source of revenue for a company with a leading product and with the in-house skills to engage in such development work. It is, however, a commercial model that is hard to scale and is labour intensive. Hence most organisations offering customised development will tend to do so where there is an opportunity to build the results into their core products.

If an ISV moves from the more traditional based revenue model to the support based one it needs to consider the implications in terms of cash flow, revenue recognition, competitive positioning and the approach of the sales force.

The choice of infrastructure and resources for support is critical to profitability. In particular, on-site support is often requested by very large organisations, but is very expensive for an ISV to offer without access to third party resources. On the other hand, this provides an opportunity to increase the price accordingly.

Product distribution, licensing and delivery

ISVs that currently use this model successfully are those that combine the following:

- A wide distribution of product, as the potential growth for support services is contingent upon the installed customer base for the product. Therefore the barriers to potential customers need to be as low as possible.
- At the same time there needs to be strong differentiation between unsupported and supported products. This can be achieved through several mechanisms based on marketing, product packaging, access to updates and others. For example, vendors looking to differentiate an unsupported product will often call it

the "bleeding edge" release and restrict downloads of older stable products.

- Making a product "hard to access" unless accompanied by support. However, there is a balance here, as too restrictive a mechanism from the project maintainer will lead to complaints and people either not using the product or getting it from another source.
- Collecting customer information at install time, which enables the ISV to target customers directly. Converting customers who have downloaded the product into customers paying for support is easier if one can market to them directly. Therefore mechanisms for getting hold of customer data, such as forms before downloading, data collection at install time and others, will be a key element of commercialisation.

Sub-model recommendations

These sub-models are suitable for ISVs who can use open source to build an installed base into which they can sell these services.

Pre-tested set of applications, technology-based subscription

A technology-based subscription that provides maintenance services such as patch management and updates, or information such as product knowledge, security alerts and other documentation, is a viable model for Irish ISVs. This model is most suited to those ISVs who are using the open source model to drive adoption of a project where the ISV has considerable development knowledge.

This model suggests that the ISV provides a service offering a pre-tested set of applications that reduces the risks businesses face in deploying open source software, together with support in deploying and maintaining the applications. This is a low cost model when compared with more labour intensive forms of support.

A good example of its use is the company SpikeSource, which combines this support-based sub-model with the migration sub-model defined below. A case history of SpikeSource giving details of its operational characteristics is given on page 31.

Services for aiding migration

A more tactical approach to selling the support model is to focus on the technical and business aspects of migrating software, from older to newer versions, whether applications software, operating system, middleware or databases.

The ISV can sell its knowledge of how these migrations can be achieved in a cost and time effective manner. These services can take several forms. The lowest cost form involves the use of web-based tools, including for example, configuration file audit tools, to carry out the migration. The ISV can scale this from telephone support through to consulting and implementation services.

Case study: SpikeSource

SpikeSource's business model is essentially the provision of automated operational services to enterprise class users of a range of well-established OS software. It has attained immediate market recognition and publicity through its founding advisory and executive management team who are all well-known names within the industry from their previous executive roles.

The software for which they provide services comprises 50 plus OS components including Apache, MySQL, PHP, JBoss, Tomcat, Axis and Hibernate. An enterprise can take one of a choice of standard software component combinations or else can define their own combination of components. The standard combinations, or stacks, are made up of OS components integrated, preconfigured and tested by SpikeSource into commonly used stack types such as an application server, web server and database server.

The software is delivered in an Enterprise Starter Kit that also contains:

- Spike Configuration Manager – Enables a user to tune the configuration of each component through that component's individual user interface;
- SpikeNet – A software tool that delivers an ongoing stream of fully tested component updates; and
- SpikeSource Technical Support – An incident based set of technical support services provided through an annual subscription.

Their basic value proposition is that they remove from an enterprise three of the significant barriers to the enterprise adopting an OS route for its software provision. These are:

- Interoperability Testing – SpikeSource has automated its test framework so that it can undertake over 30,000 tests across six platforms and runtime languages on component stacks.
- Deployment Management – SpikeSource provides an automated tool that can install an entire set of OS components – a stack or a stack and application – in one action. The built in configuration manager runs after each installation to automatically configure all selected components, whether they are a SpikeSource pre-defined stack or a custom stack chosen by the enterprise.
- Software Component Updating – A tool called SpikeNet automatically provides available updates for the OS components that the enterprise has chosen to use. It also provides information about each update that enables the enterprise to focus on the most relevant updates. All updates are tested against the complete stack, not just individual components of that stack.

SpikeSource derives its revenue stream from support services, which it offers to both enterprises and to ISVs. The enterprise or ISV can buy support packs on a per incident or an unlimited basis with varying levels of support hours, response times and electronic/voice interaction. It also provides support for a number of open source technology providers that can be acquired as a complete package in combination with its core service provision. In particular, SpikeSource is a JBoss Authorised Service Partner for a number of JBoss products. The company also offers additional specialised services to the ISV community based on the standard SpikeSource interoperability testing and support capabilities. The services are fee based for ISVs, but free to qualified open source projects.

5.4 Software as a Service (SaaS)

This business model is currently much discussed as an alternative to traditional packaging and licensing strategies. The term is used in three senses, all of which an ISV might adopt:

- A hosted service through which the user accesses the software remotely rather than installing on a local machine;
- Use of certain standards for application programming interfaces (APIs) that allow other developers to use the functionality of the application in their own applications and data formats; and
- A payment model that is more variable and less based on capital expense, for example paying per transaction rather than paying per server licence.

As a commercialisation model this report will focus on offering SaaS as a hosted service since APIs and data exchange are not commercialisation models and variable payment is a possibility with any software.

It is likely that SaaS will be successful where the ISV has unique data as well as IP that can be easily incorporated into existing customer infrastructure or applications. The market with the most potential is also more likely to be consumer and small businesses rather than large business.

5.4.1 Advantages and disadvantages

The principal advantages of the SaaS model are:

- It gives the ISV access to a regular, renewable revenue stream without the overhead of monitoring software usage;
- It can combine open source and closed source in a single offer to the customer. It also gives the customer choice that is informed by the clear difference between the costs and benefits of service provision versus downloaded product; and
- The open source model offers the customer some comfort that in the event of the failure of the ISV, at least some part of their infrastructure (i.e. the OS product) remains intact.

The chief disadvantages are:

- Last time it was tried it didn't work. The application service provider (ASP) craze of 1997-1999 wasn't successful. However, there are indications that the market has moved on since then. A Forfás study undertaken on Internet Data Centres in 2005 (unpublished) indicated that there is a move toward higher value managed services that include the hosting of data storage through to the management of more complex IT applications (e.g. HR systems, supply chain management systems, etc.). This move toward managed services augurs well for the ASP model. IDC Consultants project that the managed services market is set to reach almost €10.2 billion in Western Europe in 2008, which represents a compound annual growth rate of 7.3 percent;

- Hosted software is likely to be less customisable than non-hosted software built on the open source platform; and
- The customer is highly reliant on the hosting company's processes, which may make for a difficult selling proposition.

Economic model

Depending on the pricing options and delivery model, the ISV's economic model will differ. Offering a hosted service is a clear advantage particularly if an ISV's target market is small and midsized businesses that may not have the skills or inclination to install and maintain software. For a hosted service the advantage for the customer is in ease of use, while the advantage for the ISV is in having a regular, dependable revenue stream without the difficulties inherent in monitoring the use of the software.

Major considerations in increasing profitability

Hosting open source software is an intriguing business model. Because of the growth in broadband capacity over recent years, businesses that utilise bandwidth intensive software may reconsider the hosted model as an attractive model compared with building their own infrastructures.

From the ISV perspective, the costs of distribution will involve the leasing or ownership of bandwidth, servers and storage. In addition, the ISV will need to reassure potential customers of the data integrity, security and fault tolerance of its systems. The ISV may consider collaborating with a better-known networking or telecoms partner, rather than relying on its own branded efforts, in order to increase the potential for gaining market traction.

The barriers to entry into this type of service offering are relatively low, so that it is preferable for the ISV to apply closed source IP to the hosted offer (refer to licensing models chapter 6). Otherwise the ISV will need to rely on its brand or on a lack of interest from potential competitors for its competitive advantage.

Product distribution, licensing and delivery

The application itself must be suitable for the SaaS model taking into account constraints such as the bandwidth of TCP/IP versus local memory and disk.

In the case of business software, it will almost certainly need a degree of integration with existing systems. This might be achieved through data exchange protocols or through hosting customer data on the ISV's systems.

There are proposals in the next revision of the GPL (GPL 3) that will provide more restrictions on the uses of hosted software in order to prevent hosting companies from using this delivery model to get round the redistribution requirement of the GPL.

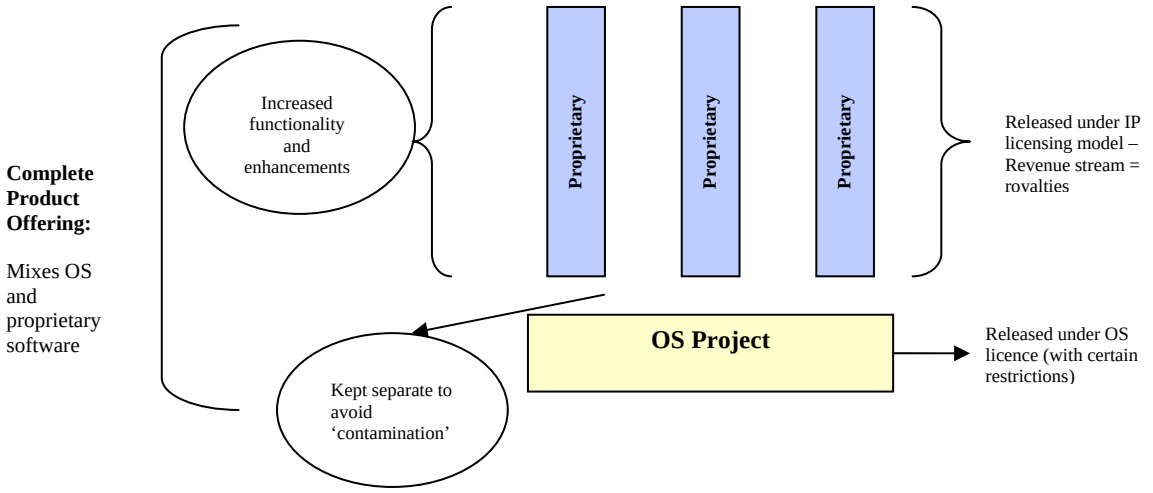
5.5 Proprietary Development on Open Source

The third commercial model recommended is a hybrid approach, which combines both OS and proprietary elements. IP for an OS project is released under an open source licence. Then, using the open sourced project as a base, the commercial organisation develops proprietary products which enhance the functionality of the open sourced project (Figure 4). This proprietary element is released under the more traditional IP licensing model. A good example of its use is the company XenSource, a young company currently coming to prominence through its advanced virtualisation software (see XenSource case study on page 35).

In some instances the commercial organisation may not have had involvement in the development of the OS project itself. Many large organisations will develop IP to enhance the functionality of open source projects without necessarily having had much connection with those projects. This is largely a defensive strategy, aimed at protecting market share.

The proprietary development model is the ideal model for components in the stack where there is the additional value necessary to make a complete product.

Figure 4: The Hybrid Model



Source: Forfás, 2006

5.5.1 Advantages and disadvantages

The principal advantages of the proprietary development model are:

- It preserves familiar economic models for software by allowing for the high profit margins associated with royalty payments, while at the same time providing a greater installed base into which the product can be sold;
- If the installed base of product is sufficiently large, then other proprietary vendors are likely to target the product that is closed source. This can have the advantage of improving the quality of the code as well-resourced organisations join the development community; and

- The ISV that releases the open source code which is taken up by a growing installed base is likely to be the de facto maintainer of that code (or is likely to have the significant maintainers working for them). Either way, this provides control over the direction of the OS development, which can ultimately benefit the proprietary product.

The chief disadvantages are:

- Standing still. With the source code open, both individuals and competitors are able to replicate features in the closed source product more quickly than would be the case with a pure closed source approach. ISVs will need to remain in line with the community on which their business relies;
- Risk of forking, where other developers take the open source code and develop a product that is not compatible with the parent code base. If this new version is widely adopted, the ISV loses its competitive advantage; and
- Losing a key developer is a problem for any ISV. However, it is likely to be more problematic for a hybrid open and closed source ISV since the developer who leaves the project possesses not only an understanding of the code but also access to the code-base.

Case study: XenSource

XenSource is a good example of company using the Proprietary development on open source model.

XenSource is in its formative stage of company development. It was set up in January 2005 and raised $23.5 million in first two rounds of venture capital funding. It currently employs about 65 people based in Palo Alto, California with offices in, Cambridge, UK and Redmond, Washington, USA.

XenSource's open source software, Xen, is a hypervisor. Xen is a very low level piece of system software that allows a single machine, typically a server, to simultaneously host multiple different operating systems and to share the resources between them, providing resource guarantees to each virtual server. It's of strategic importance to the server manufacturers because it offers an idealized hardware abstraction to all operating systems, removing the need for platform (server) specific support in each virtual server. It also allows OEMs to embed additional hardware-specific software into the system, outside the virtual servers. It is strategically important to the chip manufacturers because all of the new features that the chip manufacturer brings out can be expressed in the hypervisor without the need to change the operating system. In particular, legacy operating systems can take advantage of the features of new hardware, without needing to modify the operating system. It is strategically important to customers because it allows each server to host multiple virtual servers, increasing utilization and dramatically reducing the need for additional physical servers.

Although it could be argued that their business model is still untested, XenSource plans to generate revenue through:

- Enabling operating system vendors (to date Sun Microsystems, Red Hat, Novell, Mandriva) to deliver Xen to market completely unencumbered as a free component of their operating system; XenSource partners with

those vendors to up-sell their customer base with additional proprietary, potentially closed source components.

- The sale of mixed-source, proprietary virtualization products that incorporates the OS hypervisor and additional components. The first proprietary product (XenEnterprise) was announced in April 2006 and is based on Xen 3.0. XenEnterprise offers enterprises a fully packaged, installable virtualization platform, with bundled deployment and virtualization management capabilities to take advantage of the availability of Xen within a large data centre.
- The business model is based on the premise that as a piece of open source software, Xen can be considered and endorsed by the market, leading to it becoming a de facto, open standard. XenSource believe that this process is critical to the overall success of the business model that they are adopting. Specifically, XenSource seeks ubiquitous deployment of Xen, both through the operating system vendors and its own XenEnterprise, and leverages the footprint of both to deliver solutions to its enterprise customers.
- XenSource offers subscription based pricing for its own products, but does not directly derive revenue from the distribution of Xen by its operating system vendor partners.

Economic model

This model is essentially the proprietary model for software but with a strong connection between the closed and open sourced products. The advantage of this approach is that the ISV can get the advantages of market adoption of open source combined with the royalty payment for closed source.

Major considerations in increasing profitability

The ISV that operates the more traditional proprietary model is likely to find this the most immediately attractive model, because it has familiar revenue elements. The royalty payment covers the risk profile and sunk costs of the project as well as providing profit.

Open source models allow for the selective release of code under an open source licence to grow the installed base while retaining ownership of the IP in the code base.

The developers with the greatest understanding of the project are also likely to be most able to develop compatible existing features. Therefore, it is likely to be more profitable that these developers devote a portion of their time both to the production of open source features as well as to proprietary development, rather than for a separate team with less knowledge of the project to do the proprietary development.

Product distribution, licensing and delivery

For the proprietary development model to be successful it is necessary to have a wide distribution for the product upon which new features are built. Therefore the analysis of market adoption is important in the first instance.

Although the legal framework and mechanisms for licensing under open source seem fundamentally the same as those used for closed source licences, there are potential grey areas around some of the concepts. For example, with derivative works, there may be some challenges if a GPL type licence is used for the open source piece of work.

Any of the delivery and distribution models can be used for the closed source project. There is fundamentally no distinction between this and marketing other closed source products. However, given that one is aiming for wide distribution for the open source project, ISVs should consider methods such as freely distributed, limited functionality versions of the closed source product.

Finally this model allows for OEM and other cross licensing schemes where the OEM can consolidate the open and closed components and sell these as part of their product.

Sub-model recommendations

Using open source to leverage sales of proprietary software is challenging in terms of:

- Maintaining the technological advantage in the proprietary product;
- Building a community around the project and continuing to understand the needs of that community;
- Developing the support ecosystem.

The most realistic options in this area are in increased functionality built on an open project platform and dual licensing, discussed below, with an open source option and then optimised or packaged options for different customer sets. An interesting variation on this is MySQL, which uses the product licence as the differentiating factor (see MySQL case study on page 38).

Increasing Functionality

While the functionality required is dependent on the platform (e.g. operating system, middleware, database, language, etc.) there is ample scope for an ISV to use the open source product as a base upon which additional functionality can be sold. The advantages of this model are in driving wide adoption and lowering marketing costs through use of open source and in the expertise and understanding the ISV has of the application.

To date, ISVs have generally looked at management frameworks building on open source as a model. However, as open source moves more into business applications it is likely that adding modules such as analytical functionality to supply chain management systems will become a rich area for exploration. This is the most useful model for business applications.

The ISV will probably want to ensure that it can reuse open source code without redistributing its source code and therefore needs to choose a licence compatible with this aim.

Dual Licensing For Different Customer Sets

With dual licensing, the ISV provides the software with two licences: an open source licence and a proprietary licence.

ISVs seek to add competitive advantage based on the use of the software. However, the open source licence will require them to redistribute source code and hence lose this advantage. By providing a proprietary licence redistribution is prevented.

Companies wanting to go down this route need to have software that is appropriate to this form of licensing. To date it has been used for applications that are embedded in other systems such as database and development environments, and it is difficult to see how this can be applied to business applications where there is no embedding in other systems.

As well as not being suitable for all types of applications, there is the possibility of customers exposing the application as a service without incorporating the code into their work. The customer could then use the ISV's open source product (without buying the commercial product) and still redistribute its own application under a closed source licence.

This has proven an effective model for companies like MySQL and Trolltech. It allows the ISV to use the open source model for market share and software development while preserving a revenue stream from other customers.

Case study: MySQL

MySQL is a Swedish based organisation that promotes a range of products, some free and some paid for. The flagship product is MySQL database and it comes in a free, Community edition and a paid-for, Pro-Certified Server edition. There are other versions such as MaxDB, an open source database certified for use with SAP R3 and a Cluster version for fault tolerant, mission-critical environments. MySQL operates under a business model that involves mixed closed/open source software.

The MySQL Community Edition is available under the open source GPL licence and is:

- Released early;
- Released often; and
- Includes the latest bleeding-edge features that are under development.

MySQL Community Edition is the foundation for MySQL Network. However, it has not been certified and is not considered ready for enterprise production use. In this way the Community Edition acts very much like the Fedora Project at Red Hat, in that it is the development base for the company.

MySQL obtains its primary revenue stream from selling customers a commercial licence that allows them to use the product without it being covered by the GPL. Consequently, these customers can include the product in their own products that they sell on. MySQL charges these OEM customers a fee based on the number of licences they will sell and the level of "freedom" they want regarding their use of the product.

The licence is designed for organisations that do not want to release the source code for their applications as open source / free software; in other words they do not want to comply with the GNU GPL.

Typical examples of MySQL distribution include:

- Selling software that includes MySQL to customers who install the software on their own machines;
- Selling software that requires customers to install MySQL themselves on their own machines; and
- Building a hardware system that includes MySQL and selling that hardware system to customers for installation at their own locations.

MySQL also gains revenue from support and other services for its commercial software and, on a less frequent basis, for its OS version.

5.6 What HEIs Should Do About Open Source

IDC believes that all three main commercialisation models for open source are appropriate for HEIs, but with important caveats. Whichever commercialisation model HEIs adopt, they cannot abdicate their responsibilities to customers. Therefore they must offer product support, either themselves or by training partners to act on their behalf.

5.6.1 Support-based commercialisation

There is considerable evidence from open source ISVs that support services provision in the open source environment can be very different to that in the proprietary environment. With the support-based model, the primary focus in the open source environment is to build a community of users of whom a proportion will over time sign up as support services customers.

As discussed earlier, there is generally a time lag of some months between the time when people start using the products to when they become services customers, and in these early stages much of the support provision can be done online or can be automated. If and when usage grows and users become customers demand for support will rise.

However, HEIs are not in the business of providing support services. Substantial expertise and infrastructure is needed to make the revenue model effective, and in general HEIs are neither inclined nor able to invest in such a structure.

Scale is usually an issue: Do HEIs want to take on substantial volumes of support or would they rather pursue new R&D opportunities? Based on their primary roles and responsibilities, HEIs should continue to focus on research. However, HEIs can partner with commercial ISVs or services firms, where the HEI receives a proportion of revenues resulting from services related to supporting software developed by the HEI.

HEIs could consider training ISVs or services firms to offer first and second line support, retaining third level technical support within the HEI itself. It must be recognised that offloading support onto a third party still embodies a substantial commitment from the HEI. It requires the HEI to invest in documenting the software in order to enable transfer of support liabilities. It also requires a close and structured relationship with selected partners. HEIs may find a better response from services firms (such as resellers) that are used to selling third party products.

5.6.2 Software as a Service

The economic model for software as a service (SaaS) is based on a pay-per-use concept, rather than an IP royalty model. Therefore the standard SaaS model applies equally to proprietary and open source software.

SaaS is increasingly accepted as a standard model for software distribution. Major ISVs are in discussions with customer groups on how to migrate licence-plus-maintenance revenue streams into pay-per-use charges. The most nervous participants in the debate are the CFOs of ISVs, who worry about impacts on the predictability of revenue streams.

Moving from traditional licence-based regimes to SaaS presents a risk for ISVs, in that it challenges their primary cash generation mechanisms (licence fees and maintenance charges). In this sense SaaS presents less risk to HEIs as HEIs have no licence revenue stream to protect. Indeed, HEIs could pre-empt the proprietary ISVs in migrating to the SaaS model, gaining competitive advantage for any commercial venture in the process.

The major downside to SaaS is that substantial investments must be made in hosting infrastructure, primarily hardware and bandwidth. Salesforce.com, the most successful proponent of SaaS, outsources its hosting infrastructure completely, in order to achieve scalability and cost control. IDC recommend that HEIs deploying a SaaS model do likewise.

5.6.3 Proprietary development on open source

The significant advantage to HEIs of proprietary development on open source is that it is closely related to the traditional IP-based model they are familiar with. It is essentially a mechanism for creating a market by distributing loss-leading (in a commercial sense at least) software upon which subsequent proprietary licence revenue can be derived.

It is possible that the model is actually better for HEIs than the traditional IP model. It allows free distribution of a basic product and then charges for upgrades and add-on components. There is a good chance that free distribution will reach a wider audience and is more likely to appeal to the SME market, making it a strong model for Ireland.

The skill required is in converting enough free users to paying customers in order to sustain further development. This is in part a marketing exercise, to extol the benefits of the upgrade, and in part a product design requirement. Sufficient free functionality must be offered in order to attract demand and make the product useful, but there must be scope to add substantial revenue-earning components. Getting the balance right is important.

Using this model, IDC recommend that HEIs develop the open source basic code and the revenue-generating code, rather than reusing pre-existing open source code from a third party. Although IP rights are waived on the basic code, the knowledge contained in software design and original development gives the HEI an advantage over any other organisation that takes the basic code and tries to enhance it itself.

This model should therefore be attractive to HEIs, assuming they develop the basic software first.

6 Licence Selection

There are at least 50 licences that comply with the Open Source Initiative (OSI)[6] definition of an open source licence and many other licences are derivatives of these. Typically these variations are to respect local laws with regard to trademarks, copyright and patents.

License selection can therefore be a complex area, and the following are the most critical issues that need to be considered:

- For those projects with an existing code base whether the existing licensing of the code base prevents the ISV from making the product open source;
- Whether the licence requires or permits distribution of the source code and whether one can charge for distribution of the source code and an executable version;
- Whether the licence is compatible with the General Public Licence (GPL), the importance of which is discussed below;
- The treatment of derivative works under the licence;
- The licensing of documentation, images, sounds, fonts and other works that are not computer program source code.

6.1 Licence Universe

A large number of licences are compatible with OSI principles, and a larger number still may be compatible but have not been certified as being so. However, the reality is that there is concentration on a limited number of licences. Table 2 shows the share of each licence in November 2005 of the 69,036 projects hosted on SourceForge. (SourceForge is the world's largest development and download repository of open source software).[7]

There are two important recommendations for ISVs regarding licence selection.

- Choosing a popular OSI certified licence is an important step towards removing a barrier to developers joining the project. Individuals donating their time and companies willing to have their employees work on an open source project will be more comfortable with existing, common open source licences.
- The use of an existing licence reduces the legal costs associated with validating the licence and may also provide the licensee with access to legal funds in the event of a challenge to the validity of their licence.

[6] For information on OSI please see http://www.opensource.org/
[7] For information on SourceForge please see http://sourceforge.net/

6.2 Existing Code

As discussed earlier, there may be reasons for an ISV to convert its proprietary code to open source. However, many ISVs considering this option have found that cross licensing of code in their application prevents them from doing this, unless they are prepared to rewrite significant portions of the application.

If the ISV's examination of its source code reveals a cross licensing problem, the ISV could either work with the owner of the code to open source the whole application or with the development community to replace this code.

Table 2: Open Source Licence Share of Projects

Licence	Share
GNU General Public Licence	69.26%
GNU Library or Lesser General Public Licence	11.46%
BSD Licence	7.29%
MIT Licence	1.86%
Artistic Licence	1.85%
Apache Software Licence	1.43%
Mozilla Public Licence 1.1	1.31%
Apache Licence V2.0	0.86%
Common Public Licence	0.79%
zlib/libpng Licence	0.52%
Open Software Licence	0.48%
Academic Free Licence	0.36%
Mozilla Public Licence 1.0	0.36%
Qt Public Licence	0.33%
PHP Licence	0.23%

Source: SourceForge, 2005

6.3 Redistribution and Charging

After auditing existing code and ensuring that it can be open sourced, the second major decision the ISV needs to make relates to redistribution. There are three options based on the following questions:

- Does the ISV want users of the code to be able to redistribute both the source code as well as the compiled application within the users' redistributed product?
- Does the ISV want to require that the user does so?
- Does the ISV want this to be optional?

This ties into the business model decision discussed earlier: If the ISV wishes to follow the dual licensing model, then it has to require the user to distribute its source code. The GPL fulfils this aim while for example the BSD (Berkeley Software Distribution) licence does not.

As well as conditions on redistribution, the ISV can apply conditions on whether other parties distributing the code can charge above the cost of transferring that code. Typically this should be free for online download or the cost of materials for other methods. The GPL prevents the user from making a higher charge while the BSD licence does not.

6.4 GPL Compatibility

There are clear benefits to GPL compatibility as it is a well-understood licence that has been widely adopted. However, it can be quite restrictive and alternative licences may provide greater freedom for the ISV in using the code in other applications.

ISVs should study the proposed provisions of GPL 3 to see if these are compatible with their business models. This is particularly true for ISVs considering the software as a service model.

6.5 Derivative Works

A derivative work is a new piece of work based on original work where the original work has been modified. Typically, the derivative work will be obtained by elaborating or revising the original work. Copyright law differs from country to country and ISVs should take advice on their own circumstances.

However, the question of how the licence treats works that incorporate the ISVs source code is probably the most critical question facing an ISV in licence selection. Again, this relates to the business model as some ISVs will wish to restrict this use, while others may want to make it obligatory.

6.6 Licensing Outside of Source Code

The final important area for the ISV to consider is how to licence works that are not source code since it is likely that the application will require documentation, fonts, content and other materials. Although the open source licences discussed in this report can be applied to these works, other licences exist that may be more appropriate, particularly for documentation.

However, the same principles of audit, redistribution and charging and derivative works apply to these works.

6.7 Licence Selection

For ease of summarising the available options Table 3, below, lists the main considerations in licence selection together with the status of each of the most common licences in respect of these considerations.

In short, a critical question in deciding which licence to employ is whether the licensed work is either original or derived from a third party source. If work is original, the choice of licence is in the hands of the originator. If the work is derived, the licensing constraints of that work apply.

The importance of this question cannot be overstated. Many commercialisation efforts become unstuck because of open source "contamination" from derived work.

There are several other questions to be navigated, as shown in the table. The footnotes to the table explain these considerations in more detail. Further explanation of each licence's parameters requires analysis of and reference to the licence itself.

As an example, Table 3 shows that a main difference between GPL v2 and BSD is that GPL mandates distribution of source code for original work, whereas BSD does not. Similarly, the table shows that Artistic is the only licence that does not mandate a copyright notice to be distributed with derived work.

Table 3: Licence Selection - Key Considerations for Major Licences

Licence	Original Work[8] – Provide Source Code on Distribution?[9]	Original Work – Distribute Executable Only – Higher Fee?[10]	Original Work – Distribute Source and Executable – Higher Fee?	Derivative Work[11] – Must be Distributed Under Licence for Original Work?	Derivative Work – Must Source Code be Distributed?	Derivative Work – Must Include Copyright Notice?[12]	Derivative Work – Must Include Documentation?
GPL v2	Yes	No	Yes	Yes	Yes	Yes	Yes
LGPL v2.1	Yes	No	Yes	Yes (if based on library) No (if linked to a library)	Yes (if based on library) No (if linked to a library)	Yes	Yes
BSD	No	Yes	Yes	No	No	Yes	No
MIT	No	Yes	Yes	No	No	Yes	No
Artistic	Yes	Not Applicable – Must Distribute Source	No	No	No	No	Yes

[8] "Original Work" refers to the program (source code and binary). See section 6.2
[9] The requirement to provide the source code refers to whether the licence requires a distributor of the executable file to distribute the source code or simply provide access to it. Licences differ in how this is interpreted.
[10] The "Higher Fee" sections refer to whether or not the distributor can charge a higher fee than the cost of physical transfer, as discussed in section 6.3.
[11] "Derivative Work" refers to a work based on the original work's source code either repeated verbatim or modified. See section 6.5.
[12] How the copyright notice can be displayed varies from licence to licence.

6.8 Guidelines on Licensing

Unless there is some factor important to the ISV, IDC believe that the GPL should form the core of open source strategy in Ireland.

The licence chosen by an ISV is less important than the initial decision to go open source. Essentially the key decision in licensing is whether one's business model is compatible with the GPL or not.

Many other licences have been developed over time to address unique circumstances and this is reflected in the fact that about 80 percent of active projects use the GPL (or Lesser General Public Licence), around ten percent use the BSD licence (or variants) and the remaining ten percent address unique circumstances.

Except in the case of the hybrid model, the limitations of which are noted in section 5.5, what the GPL does not do is allow for revenue from straightforward licensing and cross licensing.

The BSD licence is closer to a "commercial" licence than the GPL. Certainly many observers have made the point that the BSD licence has less restrictive redistribution provisions than the GPL.

Use of BSD style licences will allow other parties to distribute the software, with attaching the BSD copyright notice being the only requirement. However, there are reasons why this may not be suited to Irish ISVs:

- It is not clear how allowing redistribution of an ISV's work under less open provisions than the GPL will benefit the ISV. In addition, it actively prevents the ISV from using the hybrid model;
- While there are undoubtedly successful projects not released under the GPL, most examples of successful commercialisation use the GPL or a similar licence.

In addition, it is clear that the GPL and similar licences are at the core of open source software in terms of market knowledge of the licence and the number of projects using these licences.

Case study: DERI

The Digital Enterprise Research Institute (DERI) is based at the National University of Ireland, Galway (NUIG). DERI is a research institute which aims to make the Semantic web vision a reality.

DERI was initially funded by a 12M euro grant from Science Foundation Ireland (SFI), as a new Centre for Science, Engineering & Technology (CSET) in Ireland, with Hewlett-Packard (HP) as the major industrial partner. The DERI CSET is initially a five year project, which started in 2003. Since foundation DERI has expanded rapidly, securing significant funding from other sources including Enterprise Ireland, The European Union (Framework 5 & 6 funding) and industry. HP has been a significant stakeholder in this expansion, including joint participation in EU funded projects. Today, DERI has over 70 researchers based at NUIG and has

also established relations with other research institutes in Europe, USA and the Far East which are affiliated to DERI under the DERI umbrella and with other industrial partners. HP provides management support and industry grounding for the research programmes of DERI including joint participation in other funded research projects and shaping projects for commercialisation. Today, DERI is a flagship for collaboration between traditional IT vendors and HEIs.

As a HEI NUIG is primarily an education and research intensive university and is not positioned to directly commercialise IP, unlike an industrial partner. During the early formation years of DERI different perspectives emerged as to the needs of commercial and academic institutions in the protection and commercial exploitation of IP, and to large degree open source software only compounded the cultural differences that exist. HEIs need to publish papers and ISVs need to attract revenues. While the two are not mutually exclusive they can pull a project in different directions and at varying speeds.

Open source projects within DERI have encountered many of the challenges in commercialising software in the OS context. Paramount in importance is the clear and early setting of commercial objectives (if any) for each project. In this regard it has been very beneficial for HP to communicate the particular requirements of commercial ventures based on open source and for NUIG to put management structures in place to ensure such objectives are maintained.

In developing new IP, NUIG has an eye as to how best to commercialise such IP, be it through licensing or other means and recognize the balance required to protect IP and to enable industrial partners to commercialize such IP in a timely manner. In this regard NUIG has put guideline policies in place which are designed to facilitate IP commercialisation through licensing or collaboration with industrial partners, while at the same time putting management structures in place to facilitate IP commercialisation and protect the interests of industrial partners and software developers.

HP and NUIG have established 3 types of projects in order to clarify scoping and ownership issues:

- Projects owned and delivered solely by HP
- Projects owned and delivered solely by NUIG
- Collaborative projects, in which ownership and scoping are shared.

The project type structure mitigates against the risk of confusion and purposes of projects. It also means that there is no room in DERI for a single contract type, since each project must have the appropriate terms of reference based on ownership and scope of the project and recognizing that different industrial partners have different needs.

The decision on which project follows which type is driven by the degree of shared interest; projects that interest both HP and NUIG tend to result in collaborative projects with shared value to both parties. The other determining factor for collaborative projects is the joint bidding for EU funds, which are well-scoped and defined prior to the project commencing.

7 Appendix - Indicative Market Data

7.1 Potential Worldwide Software Revenue Opportunity

Global software and associated services revenues based on open source environments represents a relatively small, but fast growing, segment of the total available market for software products and services worldwide. Although the current market for open source is relatively small compared with the total world market for software, it still represents a market measured in the low billions of US dollars. This market size is easily large enough to accommodate the aspirations of Irish ISVs and associated HEIs and will not constrain their potential expansion in any way. (For detailed data, including a four-year forecast, see table four below).

The potential for open source software is expanding very quickly with 2005 revenues expected to show a 48 percent increase over those of 2004 and a compound annual growth rate between 2004 and 2009 of 36 percent. These growth rates compare very favourably with the underlying growth of the total software market of some six percent.

IDC research also indicates that the total revenues derived from software will be larger still. For example, service revenue associated with sales of software will grow from just over US$6 billion during 2004 to US$17 billion during 2008, a CAGR of just under 30 percent.

IDC believe that this immature, but as yet unconstrained, market provides a good opportunity for Irish ISVs and their HEI partners.

7.2 Data Sources and Detailed Forecasts

All market related figures come from the following IDC sources:

- Ongoing IDC research undertaken to respond to the needs of its service clients;
- A recent IDC multi-client study in the USA with three focus groups and a survey of 507 organisations in 4 verticals conducted with the specific aim of understanding the move towards Linux;
- The IDC Systems Survey 2005 for Western Europe (1,000 interviews with senior IT managers);
- The IDC European Server Workloads Survey, 2004 (Investigating end user view of workloads running on servers); and
- Analysis of the results of the above by IDC analysts.

To provide an assessment of the size and growth rate of the worldwide market for open source software, forecasts are based on worldwide revenues for Linux and other open source environments. IDC believe this forecast to be an excellent indicator of market size, expansion and potential for

Irish ISVs considering open source development. It is a standard part of IDC's forecasting system and is considered to be reliable[13].

Using 2004 as a base year and forecasting through to 2009 the forecasts by software type are (in US$ millions):

Table 4: Forecast for worldwide Software Revenue Based on Linux and Other Open Source Environments ($M)

	2004	2005	2006	2007	2008	2009	2004-2009 CAGR (%)
Application Development and Deployment Software	1,963	3,062	4,298	5,726	7,040	8,497	34.1
Applications Software	629	907	1,411	2,037	2,905	4,535	48.5
System Infrastructure Software	1,055	1,437	1,906	2,436	3,068	3,847	29.5
Total	3,647	5,406	7,615	10,199	13,013	16,879	35.9
Growth (%)		48.2	40.9	33.9	27.6	29.7	

Source: IDC, 2006

These numbers might, on first sight, seem somewhat small for a world market opportunity, but this is likely to be the case given the nature of the business models for open source software. However, it is IDC's belief that the revenue derived from associated revenue streams will be equal to or larger than these base numbers.

7.3 Current State in Development Organisations

Currently, most open source software development is taking place within the very large, multi-national companies (Intel, HP, IBM, etc) or by a small number of specialist developers. Development in open source is not yet considered commonplace across the ISV community.

There is custom open source software development within the IT departments of business organisations. To a large extent this forms the major part of the software development effort that is taking place within the industry.

It is apparent that within a large number of developers of all types, open source software tools are increasingly used as development platforms (e.g. Eclipse).

[13] It should be noted that in IDC's forecast methodology Linux and other open source environments include all operating systems deployed aboard servers, workstations, minicomputers, and clients that are based on Linux or other Unix-like open source operating systems.

7.4 Current State in Channel Players

Although obviously at a relatively early stage in development, an increasingly significant factor in the development of the open source market is the increased activity from the IT services channel players. IDC have noted that:

- Corporate resellers are starting to build up open source practices;
- Leading systems integrators are increasingly doing open source work.

There are a large number of skilled but typically small specialists who are becoming very active within the market.

Forfás Board Members

Eoin O'Driscoll (Chairman)
Managing Director, Aderra

Pat Barry
Communications Adviser

Martin Cronin
Chief Executive, Forfás

Sean Dorgan
Chief Executive, IDA Ireland

Sean Gorman
Secretary General, Department of Enterprise, Trade and Employment

Dr William Harris
Director General, Science Foundation Ireland

Anne Heraty
Chief Executive, Computer Placement Resources (cpl) plc

Dr Rosheen McGuckian
Chief Executive Officer, GE Money

Rody Molloy
Director General, FÁS

William O'Brien
Managing Director, William O'Brien Plant Hire Ltd.

Frank Ryan
Chief Executive Officer, Enterprise Ireland

Jane Williams
Managing Director, The Sia Group

Dr Don Thornhill
Chairman, National Competitiveness Council

Michael O'Leary
Secretary to the Board, Forfás

Forfás Publications 2005/2006

Forfás Annual Report 2005	June 2006
Waste Management Benchmarking Study, A Baseline Assessment	June 2006
Skills at Regional Level in Ireland Expert Group on Future Skills Needs	May 2006
Small Business is Big Business Report of the Small Business Forum	May 2006
SME Management Development in Ireland Expert Group on Future Skills Needs	May 2006
A Baseline Assessment of Ireland's Oil Dependence	April 2006
International Trade & Investment Report 2005	February 2006
Research and Development in Ireland, 2005 – at a glance	February 2006
State Expenditure on Science and Technology and Research and Development 2004 and 2005	February 2006
Survey of Research and Development in the Higher Education Sector 2004	January 2006
Benchmarking Ireland's Broadband Performance	December 2005
Data Analysis of In-Employment Education and Training in Ireland Expert Group on Future Skills Needs	December 2005
Science Foundation Ireland: The First Years 2001-2005 Report of an International Evaluation Panel	December 2005
National Code of Practice for Managing and Commercialising Intellectual Property from Public-Private Collaborative Research Advisory Council for Science Technology and Innovation	November 2005
Competitiveness Challenge National Competitiveness Council	November 2005

Forfás Websites

The publications of Forfás and the independent advisory councils to which it provides administrative and research support are available on the Forfás website www.forfas.ie.

Email notifications direct to your inbox are available on the latest announcements, press releases and publications. To sign up for our email alerts contact us at info@forfas.ie or through the website.

The National Policy and Advisory Board
for Enterprise, Trade, Science,
Technology and Innovation

Wilton Park House, Wilton Place,
Dublin 2, Ireland

Tel +353 1 607 3000
Fax +353 1 607 3030

www.forfas.ie

Forfás