# Free software 2.0

## Commercial class software with free license flexibility

Nathaniel Palmer

Free software (and open source) license models have become the most influential force in business IT to date. The first part of this article presents a brief history of free software, combined with the findings from an analysis of the attitudes and expectations, across several hundred large and medium-sized businesses, relating to free software. The second part of this article presents Delphi Group's vision for the next wave of commercial free software, where demand is driven not by cost alone, but foremost by quality of service and increased agility.

### Free software 1.0 - a brief history of free software

The first commercial computers came with "free" software, including the source code, which could be freely shared. It wasn't until the 1970s that independent commercial software became widely available. By this time, competitive forces had lead to increasingly closed-source architectures and restrictions on redistribution.

The decades that followed saw both explosive growth in software development and rapid declines in the cost of computing power.

Hardware moved rapidly toward commodity status, while software became increasingly proprietary and redistribution restrictions were enforced at an increasingly aggressive rate. This trend led to two sets of circumstances whose repercussions now hold the potential to redefine the software industry:

1. communities of programmers have ready access to hardware horsepower, but find the latest and greatest software tools out of reach;
2. commercial developers seeking to differentiate their applications developed increasingly closed and proprietary software.

By the late 1970s, the growing cost of software first inspired the early seeds of today's free software movement, including the GNU Project and the Free Software Foundation. For decades to follow, the free software movement grew within communities of hackers who viewed commercial software as a cultural anathema. Yet the innovations these communities produced were largely relegated to command line UNIX, with little impact on end user computing or commercial software sales.

In the 1990s however, the free software trend line hit an inflection point. As a result of maturing standards (such as HTML and XML), the open orientation of the internet, the evolution of Java and J2EE, and the success of initiatives such as *Apache*, *Linux*, and *MySQL*, the terms "open source" and "free software" have become part of the modern business vocabulary. Today, free software represents one of the greatest opportunities for both buyers and sellers of software, offering what is increasingly viewed as a viable exit strategy away from the trappings of the software oligarchy and the rising cost of proprietary licensing.

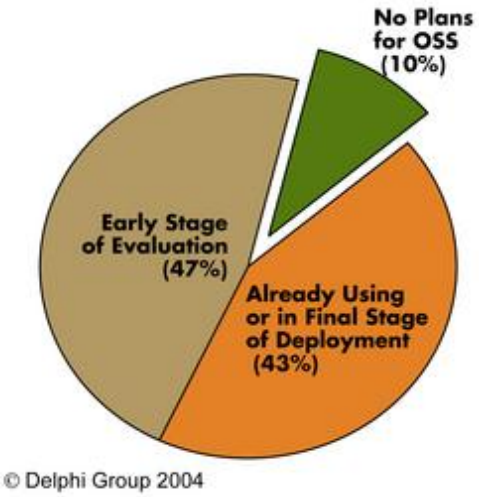
## Shifting sands across the software landscape

In a Delphi Group survey of several hundred software consumers, nearly half of respondents agreed with the statement that free software represented *"an emerging area about to revolutionize the software industry,"* while 10% cited free software as *"already the best way to go for software development and procurement."* Interestingly, when the same questions were put to software developers, the responses proved even more favorable to free software.

In March of 2004, Delphi Group surveyed several hundred large and medium-sized businesses, as well as government agencies and large universities, regarding their use of and attitudes toward free software. The organizations examined included large consulting firms such as EDS, Accenture, and CSC; large medical services firms such as the Mayo Clinic and Carle Clinic; engineering and manufacturing firms such as Alcoa, CDM, and Elekta; firms in the energy sector such as SPL WorldGroup and Santos; United Stationers Supply Company, North America's largest business products wholesaler; pharmaceutical firms including Lattelekom and Novartis; publisher Penton Media; as well as non-profit firms such as Mercy Ships. Also surveyed were several universities and government agencies including the Federal Aviation Administration, the U.S. Congress, and the State of Nevada's Department of IT. The responses presented in this article reflect only those of users and internal application developers (e.g., software consumers). Responses from commercial software developers, resellers, consultants, and integrators have not been included in the analysis illustrated in this article.

The free software licensing model allows software to be freely shared, shifting competitive differentiation among software publishers from proprietary code to the quality of support and services. By shifting the point of competitive differentiation from proprietary code to openness and adherence to standards, free software holds the potential to radically alter the economic equation that has defined the software industry for the last 3 decades. There are few, if any, developments in commercial computing, which have evolved as quickly as the adoption of free software. This is driven by a combination of internet-delivered software (i.e., downloadable code), maturing standards, and of course, a price point that is either free or virtually free compared to closed-source equivalents.

The stage of deployment for free software

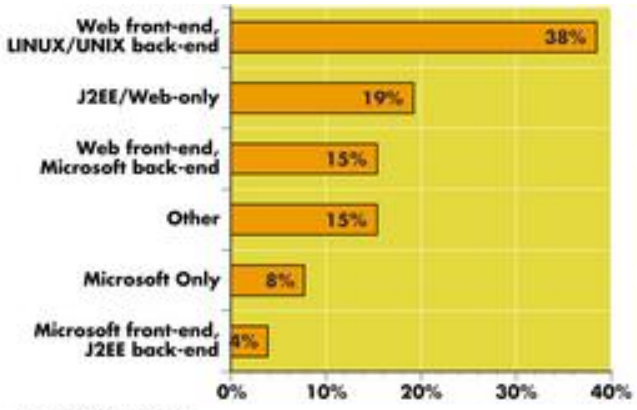



The first wave of free software to impact commercial computing was at the server level, notably the explosive rollout of Linux since the late 1990s, and the less visible but equally pervasive adoption of the Apache Web server family. It wasn't, however, simply the low (or arguably nonexistent) price point that paved the way for these and subsequent free software initiates. It was the democratization of application development by J2EE and its constituent set of community-driven standards. For this reason, most commercial adoption of free software has been on the J2EE and/or Linux/UNIX platforms.

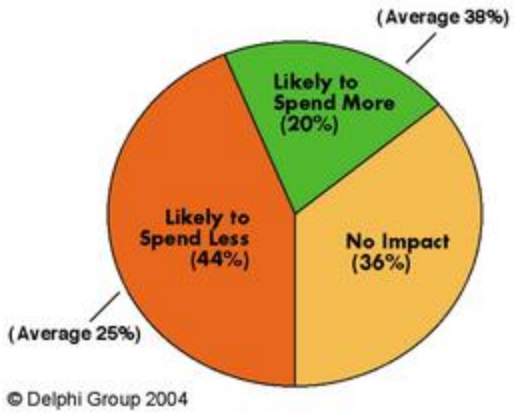
The most commonly deployed free software today includes the *Apache Jakarta* projects (Java-based server-side tools), the *Apache Tomcat* servlet engine, the *Eclipse* Java-based development environment, the JBoss J2EE application server, and tools such as the PHP scripting language and the *Samba* Windows-to-Linux integration software. The most widely deployed free software is Linux, which is today also the most widely installed UNIX variant and by far the one with the most rapidly growing market share.

The object-oriented, component-oriented, standards-based architecture of J2EE has hastened the development of multiple free software project initiatives, such as those under the *Apache Software Foundation (ASF)*. The ASF community represents over 1,000 core developers and over 50 active projects. Organized as a nonprofit association, ASF is a meritocratic community governed by a core set of laws and principals (known as *Foundation Bylaws*). Projects are

The deployment environments where free software has or will be used




What impact will the availability of free software options have on 2004 IT spending and budgets?




vetted and approved through a peer-review process involving both users and developers, comparable to what would be found in the QA process of commercial software development.

## Spending impact: economic consequences of free software

One of the fundamental arguments forwarded by free software detractors, is that without commercial incentives based on proprietary intellectual property, software innovation will cease and ultimately software consumers will suffer.

The economics behind ASF and other community-based free software initiatives are that of reciprocity. Specifically, the contributions to the development of free software will be rewarded with access to works provided by other developers.

The argument that free software removes economic incentive, however, is contradicted in part by the fact that 20% of the firms surveyed indicated they would spend more on IT as a result of available free software applications. Expectations for increased spending are likely the result of the lower entry point offered by free software, rather than the expectation that free software will be more expensive than proprietary alternatives.

Twice as many firms indicated they expect to spend less, which might be taken as fodder for free software detractors. This statistic likely ignores the fact that with a lower entry point, more projects will begin and ultimately these projects will lead to more software-based business activity. What is indisputable, however, is the fact that free software is already resulting in major changes on both sides of IT spending, with only 20% of firms citing "no impact" as a result of the availability of free software options and alternatives.

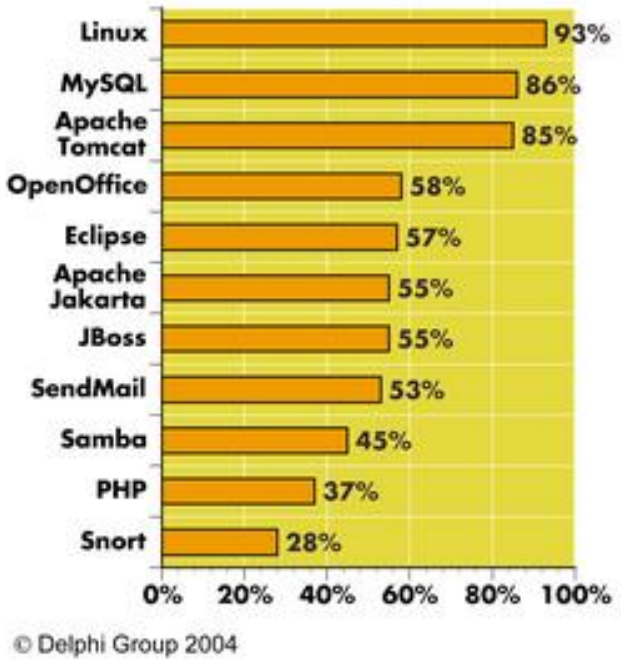
## Free software 1.5 – free software; commercial services

While the first few decades of the free software movement saw little commercial traction, the first real commercial opportunities evolved through the offering of free software (what I will refer to as "free software 1.5") with billable services around maintenance, support and custom configurations. The commercial model behind this generation of free software is more about services than software. Software is the result of a community of developers, not a single commercial vendor.

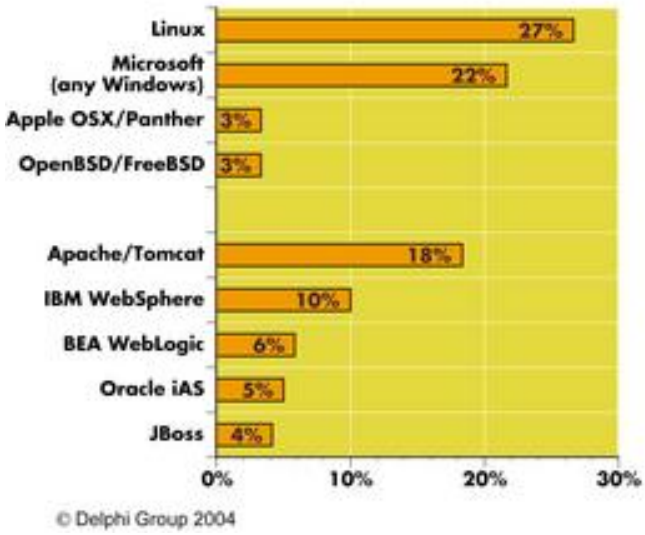
The first wave of commercial free software adoption has been defined largely by server-side tools and infrastructure elements, notably Linux and Java/J2EE-based projects. The business models for vendors in this generation of free software (such as JBoss and MySQL) are based on consulting services and support.

When differentiation and specialization come into play, it's not with regard to software development, but rather the services around a specific free software project, such as Linux. For example, commercial firms engaged in this generation of free software (such as *Redhat* and *Jboss*) provide free ac-

What free software initiatives do you use today or are likely to use tomorrow?




The anticipated platform for the next free software deployment




cess to free software and charge fees for services such as technical support, software customization, and application development.

## The free software infrastructure "stack"

Adoption of the first generation of commercial free software has been defined less by packaged applications than by piecemeal components as part of a larger Web infrastructure. The continued development of both standards and prepackaged components has followed the growing demand for a complete infrastructure "stack" or set of foundational services and capabilities. Perhaps the first available free software stack was the Internet Application Platform (IAP) comprised of a core set of components commonly referred to by the initials "LAMP": *Linux*, *Apache*, *MySQL*, *PostgreSQL*, *PHP* and *Perl*.

LAMP offers a baseline of capabilities to rival proprietary alternatives, such as *SunONE*, *IBM WebSphere*, *BEA WebLogic*, or *Oracle iAS* (each of which leverages free software components, yet do not pass on to consumers the benefits of free software licensing). What it provides is a basic set of foundation services rather than addressing the more sophisticated requirements for today's organizations seeking to develop composite applications.

Enabling dynamic business applications requires moving "up the stack" with additional capabilities focused on a more robust presentation layer, as well as access provisioning, process execution and information management tools. Delphi defines the *free software infrastructure* stack as consisting of a core set of components loosely grouped into two sets of capabilities:

*Value-Added Capabilities*

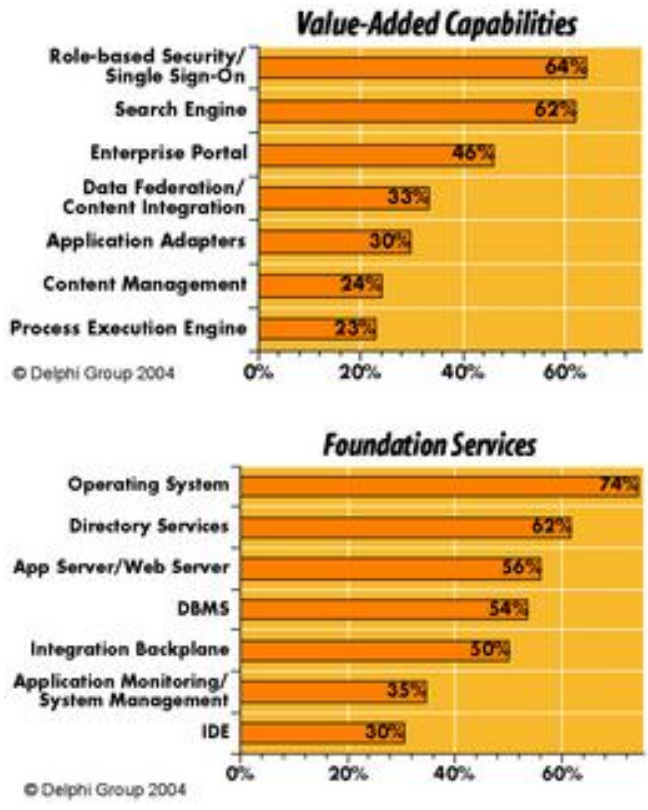
Application Adapters, Content Management, Content Integration and Data Federation, Enterprise Portal (presentation layer and UI), Process Execution Engine, Role-based Security/Single-Sign On, Search Engine

*Foundation Services:*

App Server/Web Server, Application Monitoring/System Management, Development Environment (IDE), DBMS, Directory Services, Integration Backplane, Operating System

Illustrated in the charts below, respondents were asked to rank the free software capabilities and components listed, based on what they deemed as necessary for a "complete" free software infrastructure stack. The foundation services at the bottom of the stack provide the basic building blocks and server-side resources for running, managing, and inte-

Components deemed either ”absolutely necessary” or ”significantly important” in the free software infrastructure capabilities “stack”




grating applications. These capabilities provide the “horsepower” for a free software infrastructure stack. For these capabilities, the commercial drivers are largely rooted in cost reductions – they’re free to acquire and arguably cheaper to run and manage. These were largely developed and supported through the system of reciprocity described earlier in this article. For commercial providers of free software infrastructure stack foundation services, the business model is based on value-added services, ranging from support and customization to management and delivery of software updates.

As these capabilities further commoditize the market for server-side software (such as operating systems and application servers), the point of competitive differentiation, among both traditional software “stack” suppliers *(IBM, Sun, BEA Systems, et al.)* and free software infrastructure projects, continues to move up the stack into value-added capabilities.

Free software has reached the point where it provides an equivalent to every proprietary package on the market today. It is not surprising, however, that the evolution of enterprise free software packages have followed a different route than closed source. ERP packages, for example, represent a space where free software tools have been slow to evolve, particularly compared to the much faster evolution of free software development tools and infrastructure. This is no doubt partly cultural and partly economic, as the value proposition of ERP has long been rooted in a compromise between prepackaged functions and best-of-breed alternatives (i.e., having everything already in one package rather than spending the time required to separately source and integrate best-of-breed alternatives).

In contrast, the value of free software has been firstly cost and secondly the flexibility gained through transparent source code. For these reasons, free software has arguably represented the antithesis (and perhaps for some the antidote) of ERP. So it is not surprising ERP itself has been slow to emerge within the free software community. Yet, the upside to ERP packages, notably the opportunity for pre-integrated, commercially vetted software modules, is not entirely antithetical to free software. Rather, I believe, this will in fact represent the next wave of free software adoption – free software of a commercial quality delivered through a subscription-based business model.

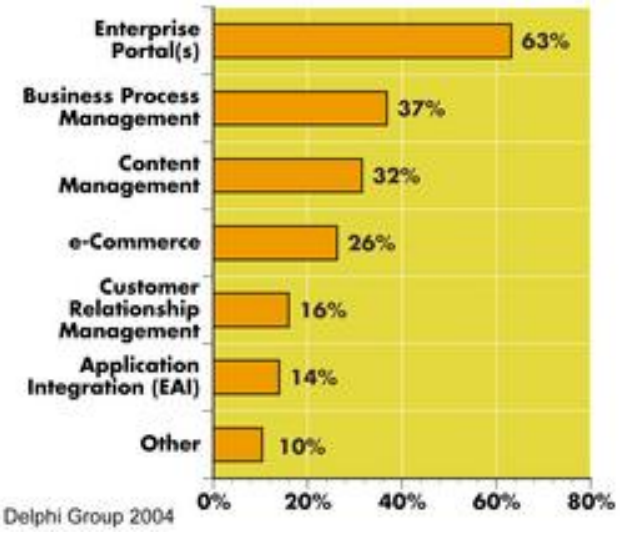
## Shifting the target from “cheaper” to “better”

As indicated by the chart below, firms deploying free software are targeting the enterprise portals by an overwhelming margin (better than 2-to-1 over any other application area). “Managing business process” was cited half as frequently, but also twice as often with integration middleware (EAI).

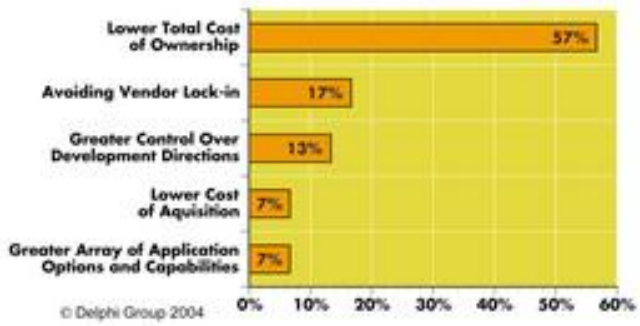
These data points help validate that firms increasingly see free software licensing and free software infrastructure stack investments as an environment for building, managing and maintaining collaborative web applications, rather than simply running server-side software (as is executed by the foundation services in the lower half of the free software infrastructure stack).

Firms embracing free software and infrastructure today are most often doing so in order to build collaborative, web-

The application areas where free software will be applied




The single greatest benefit motivating the adoption of free software




based applications. These firms are pursuing free software options primarily to realize a lower total cost of ownership.

For this reason, the manageability of free software has become a key consideration for commercial free software investments. Decisions are not made simply on the basis of what is cheaper to acquire, but more so on the long-term financial consequences and benefits of ownership. This point is validated by the fact that more respondents cited "*lower total cost of ownership*" by a ratio of nearly 10-to-1 over "*lower cost of acquisition*" when asked to identify the single greatest driver for free software investments.

Just as the bottom of the stack has benefited from standardized J2EE protocols such *as JAAS, JTS, JMS,* and *JNDI,* standards represent core drivers for value-added capabilities and enabling a lower cost of ownership for free software infrastructure. Recently, the coalescence of a set of both J2EE-derived and service-oriented standards have emerged which greatly enhance the ability to build and deploy both user-facing and distributed applications on the free software infrastructure stack.

These standards help to simplify the integration overhead required with combining individual free software projects, into a more unified framework or facilitate connectivity between dedicated software functions, such as process execution and security/access provisioning. Examples include *WSRP (Web Services for Remote Portlets),* which allows users to customize portal environments through point-and-click integration of standards-based portlets from any WSRP-compliant portal server.

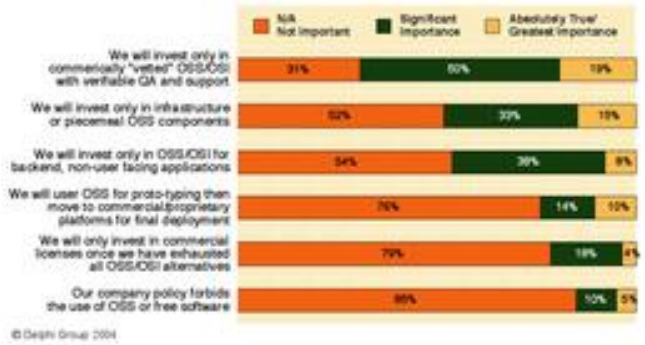
WSRP and distributed, standards-based, access control mechanisms such as *XACML (eXtensible Access Control Markup Language)* enables more administration functions to be delegated to users, while still maintaining centralized governance policies.

Leveraging these standards to delegate administration and empower users, allows for a decreased reliance on system administrators and other IT staff, without comprising content or data security. In many organizations, this should open the door for significant advantages in the reduction of software ownership costs. For these reasons, as well as increased portability of development efforts, standards will remain central to driving free software adoption. Neither standards alone or free software licenses, however, fully guarantee reduced ownership costs. Given that the greatest component of ownership costs comes from the cost of labor (both the personnel and services required for customization and support), free software that is free to obtain but labor-intensive to manage, presents no economic advantage over proprietary alternatives subject to more rigorous QA processes.

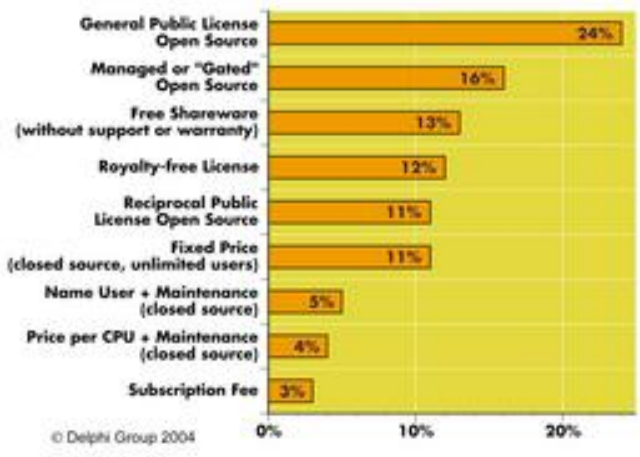
## Free software 2.0 - commercial class quality, free software advantages

For free software to be readily adopted by a majority of commercial software consumers, offerings must be able to demonstrate a standard of quality consistent with proprietary, commercial alternatives. This arguably exists already

The ranking of attitudes towards free software adoption




The preferred licensing model for the next enterprise software investment




at the component or project level, but must also be demonstrable up the entire stack of free software infrastructure.

One of the early barriers to commercial free software adoption was organizational policies that forbid the use of free software. This issue has largely waned, while acknowledgment of the need for commercially vetted free software dominates adoption criteria. Commercial software buyers today seek free software solutions with demonstrable quality assurance processes, and confirmation that integration complexity otherwise involved with combining piecemeal software components has been successfully addressed.

An overwhelming majority of responses (70%) cited the need for the commercial vetting of free software offerings as the leading requirement for investment and adoption. This requirement is consistent with the 10-to-1 favoring of ownership costs over acquisition costs in the identification of perceived free software benefits.

The ability to provide commercial class quality within free software is a function of both the licensing model and delivery mechanism for bringing software to market and into the hands of commercial buyers.

## Free software licensing models: myths and realities

For many, "open source" and "free software" are understood to be synonymous with "no cost". As a result it is often erroneously assumed that free software cannot be sold. However, free (and open source) software mustn't be confused with "shareware" and "royalty-free" licenses. "Shareware" software is normally freely distributed software with voluntary registration fees. Sometimes, you pay to get extra functionalities. This model does restrict the salability of derivative works, and the source code typically is not distributed and cannot be modified or embedded. A similar model is "royalty-free" licensing, which provides for a one-time right to integrate code with other applications (e.g., an embeddable software engine) and often does provide the right to resell derivative works, but typically does not provide access to improvements made to the original source code. In contrast with these two models, however, are free and open source software license models that allow free access and distribution to source code, while also supporting a spectrum of viable software business models.

Without limitation to specifically targeted free software investments, by a significant margin firms indicated a preference for "free software" and "open source software" licensing models for their next enterprise software purchases. Although the implied endorsement of free licensing may belie the more frequently stated requirement for stable, tested, and commercial class software, these findings underscore the growing momentum away from proprietary, named-user license models.

The defining characteristic for free software licensing is not the inability to charge a fee for software, but access to source code.

Each free software project family (e.g., ASF, GNU, BSD, etc.) has developed its own specific licensing policy, but all are based on a variation of one of the following:

- *Reciprocal Public License* - provides reciprocal rights to all changes, but does not allow for resale or licensing of derivative works;
- *General Public License (GPL)* - free to install and share, but does not allow for proprietary change or licensing of derivative works;
- *Lesser General Public License (LGPL)* - allows components of free software to be embedded in proprietary licensed applications;
- *Managed free software licensing* - combines support and maintenance benefits of licensed software, as well as ownership and licensing of derivative works.

The model which most closely follows the requirements for commercial software consumers today is: *managed free software*.

## Free software 2.0 = "managed free software" licensing and delivery

Managed free software licensing offers a combination of support services and software updates to free software. Under this model a commercial vendor assumes the responsibility for testing and validating (either internally developed or community derived) software updates, delivered over network-based web services.

Early variants of this model include the *Red Hat Network*, which offers subscription-based services for delivering updates to the Red Hat Linux platform, and *Gluecode Software*, which offers a model of package applications and infrastructure.

Gluecode focuses on the top layer of the free software infrastructure stack, delivering value-added capabilities (on top core free software foundation services) including process management, security management, and an enterprise portal framework. These sets of capabilities are delivered through a pre-integrated set of software components, delivered as a locally-deployed server *(Gluecode Enterprise Server)*. Gluecode has partnered with the Apache Software Foundation (ASF) to validate, extend and package a set of free software projects (notably those falling *under Cocoon*, *Jakarta*, Portals, web services and XML-specific project families) which address the top layer infrastructure requirements for the development of collaborative web applications. The *Gluecode Managed Free Software* service provides a delivery platform, which combines the Gluecode development methodology with automated software delivery.

## Conclusion

Free software is already shifting the power curve from the software vendor back to commercial software consumers.

The recent research presented in this article illustrate that a majority of firms already view free software as a strategic lever to lower the cost of development and maintenance of enterprise software. This is based not simply on the availability of "cheaper" software, but the expectation of a lower total cost of ownership enabled by the combination of free software flexibility and commercial class capabilities.

## Copyright information



## About the author

Nathaniel Palmer is Delphi Group's Chief Analyst and Vice President. He is also the director of the Business of Technology practice, where for over a decade he has helped define the strategic positioning and market strategy for some of the industry's most visible leaders. Nathaniel shapes much of the Delphi Group's thought leadership, is the co-author of *The X-Economy* (Texere, May 2001) and has authored over 200 studies and published articles. His insights can be found in publications ranging from *Fortune* to *The New York Times*. He is also the Association of Information Management's first recipient of the Workflow Laureate and was recognized as Master of Information Technology.