DRUG DISCOVERY TODAY BIOSILICO

The case for open-source software in drug discovery

Warren L. DeLano

Widespread adoption of open-source software for network infrastructure, web servers, code development, and operating systems leads one to ask how far it can go. Will 'open source' spread broadly, or will it be restricted to niches frequented by hopeful hobbyists and midnight hackers? Here we identify reasons for the success of open-source software and predict how consumers in drug discovery will benefit from new open-source products that address their needs with increased flexibility and in ways complementary to proprietary options.

It is the nature of innovation to make obsolete current technology, current business, and current law. Nowhere is this more apparent today than in the battle over internet music downloads, where the music recording industry is struggling against a starkly changed reality [http://www.riaa.org, http://www.itunes.com]. Similar impacts of internet technology are either expected or already being felt in publishing, movies, retail, television, advertising, and software itself. It is changes in this last area that will have the greatest near-term impact on drug discovery, as traditional approaches to development, sales, and support of scientific software are supplanted by modern ones, including open-source methods.

What is open source?

Although now a familiar term, 'open source' still means different things to different people. To the rebellious hacker, open source represents the ultimate expression of creative freedom and defiance of established ways. To the frugal consumer, open source bears a striking resemblance to the elusive 'free lunch'. To traditional software publishers, open source is a serious threat to licensing revenue and current business models. But to the pharmaceutical manager, tasked with delivery of robust information systems, open source is simply a way to gain increased flexibility and lower upfront costs in exchange for assuming greater internal responsibility over acquired software.

Much existing open-source software today has been written by teams of volunteers, each with personal motivations for contributing to a given project. On a global scale, such development only recently became practical owing to the advent of the Internet and adoption of distributed development tools such as Concurrent Versions System (CVS) [http://www.cvshome.org] and SourceForge [http://www.sourceforge.net]. These factors suddenly made it possible for diverse teams of like-minded individuals with common interests to collaborate on the creation of software, irrespective of time zone or geographic location. Nowadays, there are at least tens of thousands of open-source projects underway involving the efforts of hundreds of thousands of developers.

According to emerging standards [http://www.opensource.org], to be considered open-source, a software product must be available in source-code form and licensed in a manner that permits unrestricted redistribution and creation of royalty-free derived works. Furthermore, the software license must not limit how, where, or by whom the software can be used. Because these requirements are fundamentally incompatible with traditional

Warren L. DeLano
DeLano Scientific LLC,
400 Oyster Point Blvd.,
Suite 213,
San Francisco,
CA 94080, USA
e-mail: warren@delsci.com

1359-6446/04/$ – see front matter PII: S1359-6446(04)03363-X

license-centric software business models, commercialization of open-source has been slow. But now, with industry leaders backing open source [http://www.ibm.com/developerworks/opensource; http://www.apple.com/opensource], and smaller companies succeeding with it [http://www.mysql.com; http://www.sleepycat.com], that appears to be changing. Consumers and vendors alike have realized that there are benefits to be had from open-source software that are the worth the trouble of obtaining, even to the point of adopting new business practices.

What are the benefits of open-source software?

Generally speaking, open source represents a shift in the balance of power, away from the traditional mode where a software vendor has complete control over its code, over to a mode where that power is effectively shared between vendors and consumers. The consumer benefits from this new arrangement in the following ways:

(i) Immediate product availability: Open-source software can be obtained right away, simply by downloading from the Internet. There are no upfront cash expenditures, counter-signed agreements, or high-level approvals necessary.

(ii) Try before you buy: The only initial costs of open-source software are those expended in investigating and trying a new tool. How often have companies purchased expensive software systems only to discover months after acquisition that the product cannot effectively perform the intended function? With open source, if a tool is discovered to be unsuitable, losses are minimized and the company is free to move on to an alternative.

(iii) Vendor independence: With open-source software, there are several options for obtaining support. Once the source code is obtained, a customer can elect to self-support using an internal team. Alternatively, a customer can contract with the open-source vendor to provide needed support or customizations. A third option is to use third-party developers to maintain existing systems or build on top of open-source infrastructure.

(iv) No mandatory license fees: Even after deciding to commit to an open-source solution, a customer is under no obligation to pay a large sum of money for the license. Although an open-source vendor might ask for an upfront fee to establish a support relationship, the magnitude of fees charged is tempered by the fact that the customer can go elsewhere for similar services.

(v) No black boxes: As customers have source code, when problems arise they can be diagnosed and resolved by company staff equipped with a full description and understanding of the underlying processes. Gone are the incomprehensible bugs, black-box workarounds, and unhelpful limits on internal information that typify integration work performed with closed-source components.

(vi) Flexibility: Because open-source software systems can be changed by anyone with a reason to change them, there is always the possibility of customizing code to better suit current needs. Although such customizations do consume resources and should not be attempted half-heartedly, there is no gate-keeping vendor with 'veto power' over the implementation or prioritization of desired changes. A consumer can always maintain their own customized version of the code if an open-source vendor is unwilling to incorporate changes into the official version.

(vii) Open standards: Unlike with proprietary software, there is no market pressure to create proprietary or exclusive file formats. In fact, there is an opposite pressure to adopt existing standards for data interchange, because doing so can often mean a substantial reduction in the amount of development work required to create a new tool.

(viii) Competition: Open source can create new low-cost options in markets monopolized by one or two major vendors. Even when a consumer continues to buy from a dominant vendor, they still might benefit indirectly from the competitive pressures applied by a credible open-source alternative in a market otherwise lacking meaningful competition.

(ix) Collaboration: As there are no secrets within open-source software, the barriers to informal collaboration between and among users and developers are very low. Open communication about usage or improvement of such tools can take place without the careful legal agreements that often surround proprietary software. As a result, usage and development of open-source software often becomes a community effort, where involved parties help each other to accomplish related goals.

(x) Consolidation: Open source is the software equivalent of scientific methods publications, and it has many analogous benefits for drug discovery, including the shared progress resulting from the impact of numerous complementary advances. Once a problem is solved in open-source form, everyone benefits from not having to incur the cost of solving that problem again before making forward progress.

Although there are ways to achieve some of these same benefits with proprietary code, such as Microsoft's Shared Source Initiative [http://www.microsoft.com/resources/sharedsource/default.mspx], only true open-source software that permits unrestricted redistribution and royalty-free creation of derived works can provide all of these benefits. For example, if a product does not allow for unrestricted redistribution, then a consumer might be subjected to additional fees each time the manner, extent, or location of their software usage changes. If a product does not allow for royalty-free derived works, then its users are dependent exclusively on that single vendor for improvements or changes. There is far less choice and flexibility in that kind of software market.

What is the global impetus behind open-source software?

At the societal level, open-source is an innovative solution to a deep-rooted systemic problem: most legal systems

currently deem software programs as stand-alone independent works instead of recognizing them as productive inventions optimally built on previous inventions. Traditional inventions are protected by patent law, and it is widely understood that patents serve two purposes: (i) to provide a means of encouraging innovation, through granting of lucrative time-limited monopolies on new inventions, and (ii) to provide a means of increasing common capacity, through immediate publication of patents and their timely expiration. All patented inventions, such as novel chemical entities, medical devices, and manufacturing processes, eventually become practicable by all, and there is a tremendous cumulative effect as the previous generation of patented technologies serves as a royalty-free foundation for the next.

Current intellectual property law prevents this from happening with software because computer programming is not yet recognized as a new and unique form of invention requiring special legal treatment. Software is still treated like a work of art instead of as a useful tool. Although software now serves an analogous productive role in information economies to mechanical inventions in manufacturing economies, computer software programs, unlike other inventions, are protected under copyright law, which affords the owner exclusive control for almost a century from the date of publication [http://www.copyright.gov]. Although this treatment has had the positive effect of enabling the software industry to enjoy sustained growth, such excessive protections have hindered the cumulative development of a common software capacity that would best enable new software technologies and thereby provide numerous downstream benefits to society.

As the law currently stands, software innovators are rewarded for their efforts many times over, but little cumulative progress is being made. Computer programmers starting new 'from-scratch' projects today that are not based on costly licensing of existing code still struggle with many of the same challenges that plagued software development efforts twenty years ago, such as unreliable memory management, clumsy parsing, and insecure string handling. Almost all creation of a royalty-free common capacity has been through open-source development efforts, such as Linux [http://www.linux.org], the GNU developer tools [http://www.fsf.org], and the C++ standard template library [http://www.stlport.org].

Perhaps the best solution to this systemic problem would be the creation of special short-term copyrights for works of software that would be effective for only a decade or so and would require copyright owners to place machine-readable source code in escrow to benefit from government-enforced protections against unauthorized use or copying of their products. At the end of the term, the code would then enter the public domain and become available to enable the next generation of software to be developed by other parties. Such a system would both reward innovation and promote cumulative progress in software, just as patents do today for non-virtual inventions. Nevertheless, established interests in the software industry will probably mean that such legal changes will not be implemented soon.

Thankfully, there is an alternative: open-source software provides an effective way of obviating software copyrights via creation of common capacity, through a means that is not easily subverted by powerful economic players, despite some recent attempts along these lines [http://sco.iwethey.org]. With leading open-source products such as Linux, Apache [http://www.apache.org], OpenOffice [http://www.openoffice.org], and MySQL [http://www.mysql.org] now competing well against established proprietary offerings, the question is no longer, 'Will open-source software succeed?' – it already has. The question instead becomes, 'What are its natural limits?' and specifically to this article, 'What are the potential impacts of open-source software on drug discovery?' They could be far-reaching.

What makes open-source software suitable for drug discovery?

Drug discovery is already enabled by open academic research, open publications, and open databases, and there are good reasons why the industry could likewise benefit from open source as well as other open technologies. Chief among these reasons is the fact that unlike in other mature sectors of the economy, such as banking or insurance, software needs in drug discovery are neither static nor well defined. Therapeutic discovery is a dynamic activity that will continue to evolve, exploiting new technologies and scientific discoveries as it does so. Indeed, the recent emergence of information-intensive activities, such as high-throughput screening, genomics, combinatorial chemistry, rapid structure determination and informatics, has made drug-discovery software more of a moving target than ever. Companies engaged in therapeutic discovery cannot afford to wait for proprietary application vendors to catch up with their needs.

Instead, companies need software tools and applications than can be rolled out immediately and then modified or extended in-place, with implementation cycles of days to weeks, rather than months to years. Research-driven companies need software that is adaptable and will not fail because of modifications made to one part of a larger system. This kind of robustness and modularity is an innate characteristic of successful open-source projects, as the only way distributed development can work is by developers adopting architectures that allow independent changes to be made without destabilizing the overall system. Reflecting this requirement, many open-source projects consist either of numerous stand-alone programs, as in BSD/Unix [http://www.freebsd.org] or GNU/Linux, or of many small modular components, such as those found in Apache, Perl [http://www.perl.org], and Python [http://www.python.org].

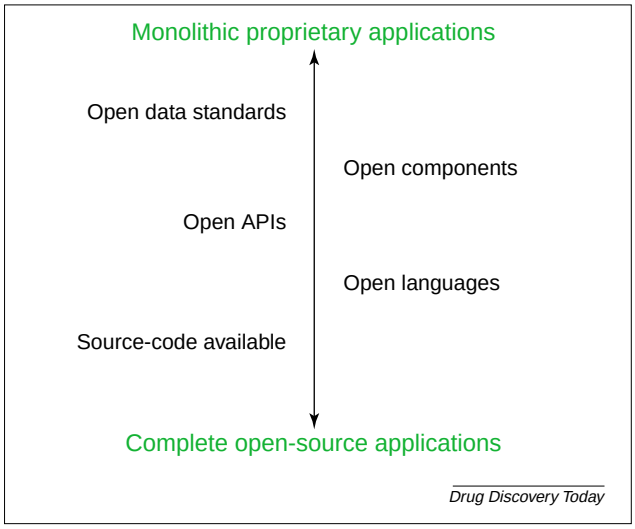

FIGURE 1

The spectrum of openness. Traditional drug-discovery software systems consisted of large proprietary applications found near the top of the spectrum. Over time, however, software systems have become increasingly open, with movement down the spectrum. If this trend continues, open-source applications will eventually become common in drug discovery. (APIs: Application Programming Interfaces)

The notion that a single third-party software company could create a monolithic, all-encompassing software system capable of meeting the needs of a modern drug-discovery operation is highly suspect, particularly because most vendors do not have access to the proprietary inner-workings of drug-discovery research. Without such information, a vendor could not hope to develop accurately targeted proprietary software solutions, even it they had enough time – which they do not. Instead of proprietary mega-applications, customers need small inexpensive modular software components that do one task well and can be integrated rapidly by in-house staff to meet specific research needs as they are encountered. Open components best enable customers to stitch together best-of-breed tools to accomplish difficult tasks in optimal ways.

What advantages does open-source software offer for integration?

Generally speaking, integration is best enabled through openness, and the more open a component is, the greater its potential utility. As illustrated in Figure 1, Openness can actually take many forms: adherence to open data standards, use of open and modular components, publication of open application programming interfaces (APIs), exposure of open languages, release of source-code, and adherence to full open-source criteria. From this perspective, open source is thus one extreme on a 'spectrum of openness' that includes a whole range of approaches with open characteristics.

Some recent successful commercial examples of openness include the Molecular Operating Environment (Chemical Computing Group, Inc.), a computational chemistry system based around an open language [http://www.ccg.com], PipelinePilot (SciTegic, a subsidiary of Accelrys, Inc.), an informatics integration environment based on an open component architecture [http://www.scitegic.com], and the expanding portfolio of chemistry and physics software tools of OpenEye, Inc., which are largely based on open-application programming interfaces and the Python open-source language [http://www.eyesopen.com].

Clearly, all such open technologies have value in enabling integration. Nevertheless, it is our belief that open-source software is the superior choice and will provide the greatest long-term benefit to consumers. Given two equivalent commercially-supported software options, one more open and one less so, we believe that shrewd informatics consumers will come to prefer the more open product. Why? Because commercially-supported open-source software occupies an optimum middle ground between proprietary commercial software and scratch-built, in-house software. It can provide lower upfront costs than either option, with the accountability of commercial software, but with the flexibility and consumer autonomy of in-house code.

Retention of control is a key reason to prefer open-source products. Only open source keeps power in the hands of the consumer. Although closed-source, open-architecture components can meet the same infrastructure needs as open-source equivalents, heavy reliance on proprietary components means giving up control of critical systems to outside vendors who do not have the company's best interest at heart. When customers develop significant reliance upon closed-source components, they can naturally expect vendors to use that leverage to squeeze them for as much revenue as they can get, just short of driving them to switch products. Unfortunately that pattern is part of the history of drug-discovery software: consumers who developed heavy reliance upon closed-source third-party systems lacking meaningful competition have sometimes regretted their decision after vendors raised prices to bolster revenues, or worse still, abandoned those products in search of greener pastures. Given the limited size of the drug-discovery software industry and the limited competition present within it, open-source software could be the only sure way around these problems, short of developing all the critical code internally.

Will open source emerge spontaneously in drug discovery?

To some extent, open source has already begun to emerge in drug discovery. Perl is widely used in bioinfomatics, and Python likewise in cheminformatics and structural biology. Some notable examples include RasMOL [http://www.openrasmol.org], Babel [http://www.openbabel.sf.net], and PyMOL [http://www.pymol.org] (see Figure 2). However, most existing open-source scientific software has originated in academic settings.

It is important to recognize that open-source products have most often been created by the primary users of the software, and not by a disinterested third party. In drug

discovery, unlike with operating systems and development tools, there is no large population of self-motivated programmers working to develop pharmaceutical code in their spare time and for their own use. It would therefore be unreasonable to expect spontaneous rapid emergence of the numerous open-source systems that would be required to support drug discovery fully.

Rather, open-source software will only become a major factor in this industry if pharmaceutical and biotech companies take proactive steps to ferment its creation, by requesting open-source solutions from existing vendors, by directly participating in open-source development efforts themselves, or by choosing to work preferentially with vendors that are prepared to succeed via open-source business models, such as open-source software companies and consulting groups willing to develop open-source implementations. Without direct consumer pressure, the emergence of open-source solutions in drug discovery will continue to be slow – given a choice, vendors would prefer to develop proprietary software to retain maximum leverage over their customers.

How and why should software vendors become more open-source friendly?

Ultimately, vendors may not have a choice in this matter. If open-source is in fact better than proprietary software for drug discovery, and if consumers continue to pursue increased openness in the products they buy, then market forces alone will eventually transform this environment into one where open-source products are common. Vendors that embrace this trend by evolving their business methods are likely to gain market share; those that cling exclusively to proprietary ways might lose out.

Some significant changes in thinking are required for vendors to succeed via open-source: whereas proprietary software vendors tend to view industry-wide usage of their software without payment as lost revenue, opportunity for competition, and a form of stealing, open-source vendors view such free usage of their code as a prerequisite for success. Widespread free usage of their code is the platform upon which open-source vendors deliver value-added commercial products and services to their customers. Proprietary vendors have traditionally viewed licensing as the climax of a business relationship, and support as a long-term liability or constraint. By contrast, open-source vendors see licensing as just the beginning of long-term relationships in which parties work together to increase product effectiveness and enable customer successes. Proprietary vendors fear and discourage communication or collaboration between customers, whereas open-source vendors support and encourage such interactions, as they reduce support costs and increase overall product utility.

Nevertheless, to succeed ultimately, open-source vendors, like all vendors, must provide customers with clear value beyond what is available elsewhere. For the pharmaceutical and biotech industries, this primarily means providing solid accountability on top of open-source offerings. As with proprietary software vendors, open-source vendors must answer for the development, maintenance and support of their products. All that open-source vendors really need to do, however, is to provide the equivalent benefits and services on top of open-source software platforms that proprietary software vendors currently do, and then the intrinsic consumer advantages of open-source products will do the rest of the work, insuring broad market acceptance. Much of the successful deployment of Linux and MySQL in commercial settings over the past five years is attributable to the fact that Red Hat, Inc., and MySQL AB have been (respectively) leading and accountable vendors for these open-source products during that time.

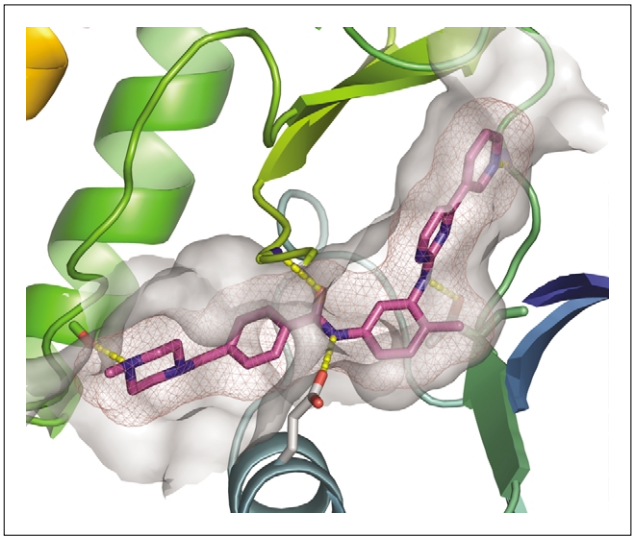

FIGURE 2

Some open-source tools already serve useful functions in drug discovery, as illustrated by this PyMOL-generated image of Novartis's drug Gleevec in complex with the cAbl proto-oncogene (PDB code: 1iep). PyMOL [http://www.pymol.org] is an open-source molecular graphics system developed, supported and maintained by DeLano Scientific LLC [http://www.delanoscientific.com].

Conclusion

Open source is here to stay and will continue to spread. Its success can be attributed to both consumer-driven market forces and an innate human desire to build efficiently upon previously existing technology. Open source and other open technologies have already found their way into a variety of commercial products in drug discovery, and a clear trend toward openness is apparent. Nevertheless, it is the consumers of drug-discovery software who stand to benefit most from increased openness, and it is their market choices that will primarily determine the growth of open-source software in drug discovery. With this in mind, pharmaceutical managers can now decide whether simply to wait for open-source options to emerge, or to take an active role in fostering open-source tools tailored to meet their company's specific needs.