Open source and private source: foundations

David Ing
Aalto University, Department of Industrial Engineering and Management
151 Booth Avenue, Toronto, Ontario, Canada, M4M 2M5
coevolving@gmail.com , +1-416-459-4915
Supervising professor: Eila Järvenpää

Abstract

This research paper is an excerpt from a forthcoming dissertation titled "Open source with private source: coevolving architectures, styles and subworlds in business". The content has been extracted from the first and second chapters, particularly on foundational definitions. It has being contributed to the *Arctic Workshop 2012* as a research paper as part of a thesis under development.

This thesis, as a complete work, inquires into the question: How do *open source* and *private source* coexist and coevolve as patterns of behaviour in business? The research approach chosen is inductive, from nine cases in which both open source and private source have been in play. Theories built in the fully-developed thesis are placed into pluralistic contexts, as an inductive approach to multiparadigm inquiry.

Coincident with the theme of "Innovation and Sourcing" for the *Arctic Workshop 2012*, this research paper aims to explain the terms "open source" and "private source", mostly as distinct patterns as phenomena in contemporary business. The larger agenda of research into open source **with** private source has been largely precluded due to length.

1. Introduction and outline to the research

The phrase *open source* is commonly understood in the context of software development. Narrowly, it is a form of licensing; broadly, it is the way in which a software development community operates. This understanding can be extended to other types of systems, such as a business as a whole. *Open source* has a meaning more specific than *open*. Open *source* is associated with visibility to system internals, whereas open *interfaces* are associated with external protocols.

A system with *open interfaces* has public protocols on its boundaries that enable external systems to interoperate, producing functions emergent in the containing system of systems.

A system with *private interfaces* has privileged protocols, whereby some external systems will have had access and functionality disclosed and authorized selectively.

A system with *open source* discloses its internals to external parties, enabling a potential for dialogue and/or collaboration on the modification of its behaviour in some future version or release.

A system with *private source* reserves the visibility of its internals with a privileged group, thereby retaining responsibility and authority for maintaining and enhancing behavioural integrity for the containing systems of systems.

Open source does not necessarily connote open interfaces, and private source does not necessarily connote private interfaces. In the more general sense for businesses, the degree of openness and privateness can be seen as relative.

Open source and private source have traditionally been viewed as alternative patterns of behaviour for social systems. As an example, government generally exhibit open source behaviours, with transparency on system operations seen as a virtue. In contrast, businesses commonly exhibit private source behaviours as a means to maintain competitiveness. Open source and private source therefore reflecting contrasting moralities.

A business enterprise can, however, simultaneously operate with both open source and private source behaviours. This study describes and analyzes histories of cases at IBM where (i) open source and private source projects have run in parallel, (ii) private source projects have been disclosed to open source, and (iii) open source projects have been enclosed to private source. The cases in this study are non-exhaustive, in the context of this company and this industry, and may be complemented by other research.

Section 1.1, below, describes the research question and observed phenomenon driving the study. Section 1.2 previews the conclusions of Chapter 9, with findings and suggested extensions of potential interest for researchers and practitioners.

1.1 Question: How do open source and private source coexist and coevolve as patterns of behaviour in business?

This scope of this study is framed with cases where both open source and private source patterns of behaviour have been exhibited within an historical context. The three subsections below describe:

- the phenomenon: Since 2001, open source contributions by corporations has increased at the same time that open source is being embedded into private commercial offerings;
- the impact: Pioneering companies have shaped organizational, industry and social contexts to make open source viable for private commercial businesses; and
- the inquiry: What can we infer from IBM's learning on open source with private source that may be applicable in other situations, or for other organizations?

The path for the study has been to observe practices in situations where both open source and private source have been concurrently in play, and build theories that might be further developed in later research. The large body of research on open source – both in open source licensing, and the operation of open source communities – is acknowledged, with this research study embracing the concurrent operation in private source not generally within the scope of study. Additionally, the body of research on competitive strategies for private source software that excludes open source projects takes an emphasis different from this research scope.

A key question of interest to many parties, “how do I make money with open source?”, is an agenda not covered in this research study[1]. Entrepreneurs have and will find ways to develop customers and revenue streams to enable viable and profitable businesses. This study focuses on the behavioural aspects of businesses operating in a world that includes open source.

1.1.1 Phenomenon: Since 2001, open source contributions by corporations has increased at the same time that open source is being embedded into private commercial offerings

The resources put into open source are not insignificant. In the Linux project, the estimated development value of Linux 2.6.30 was over €1 billion, with an estimated annual R&D of €228 million in 2008 (García-García and Alonso de Magdaleno 2010). The Eclipse Foundation doesn't provide hard numbers, but estimates that its ecosystem generates in the range of a billion dollars.[2] The rise of open source in business is illustrated by (a) the contribution of assets to open source organizations, and (b) the embedding of open source into private commercial offerings.

(a) Contributions of assets to open source organizations have been counted as lines of code. As of 2009, the Linux kernel had received 1.4 million lines of code from Red Hat, and 725 thousand from IBM. Sun had contributed 6.5 million lines of code to Java, 2 million lines of code to Solaris, and 10 million lines of code to OpenOffice. On Eclipse, IBM had contributed 12.5 million lines of code. Google estimated 10 million lines of code for Android, and 2 million lines of code for Chrome (Asay 2009).

(b) The embedding of open source into commercial open source is listed as one of three broad categories of company business models related to open source:

- (i) *Pure open source* models “use only open source software licenses and generate their revenue via services, support (both ad hoc and subscription-based), customisation, and training”.
- (ii) *Hybrid open source/commercial licensing* models with either “*dual licensing* strategies that see proprietary licenses used for ISV and SI partners, as well as wary corporate clients”, or an “*Open-Core* approach, making additional services, features and functionality available to paying customers using SaaS [Software as a Service) or proprietary licensing”.
  - “The term “Open-Core” ... describe[s] the use of proprietary extensions around an open source core, ... [separating] community users from commercial customers enabling vendors to focus on the needs of each”.
- (iii) *Embedded open source* models “[see] open source code embedded in a larger proprietary product – be it hardware or software. Prime examples of the software approach are IBM’s use of Apache within WebSphere and Actuate’s use of BIRT within the Actuate 10 portfolio” (Aslett 2009).[3]

1 One work that could be informative on the question of financial business models is Chris Anderson, *Free: The Future of a Radical Price*, Hyperion, 2009.

2 See “Eclipse: The billion-dollar baby?: Eclipse's Milinkovich talks up the Eclipse ecosystem”, *InfoWorld*, September 18, 2006, http://www.infoworld.com/d/developer-world/eclipse-billion-dollar-baby-838 .

3 The phrase “embedded open source” used in the sense of a business”model” is disambiguated from the use in a technical platform, i.e. open source software as firmware in an embedded device, e.g. mobile smartphones.

Of these categories, the cases selected for this study has focused primarily on embedded open source models where the target customers are commercial enterprises. Open-core approaches may be applicable for pilot projects initiated by communities internal to the company, with commercialization requiring migration to a license whereby the asset is unrestricted in its redistribution. The legality of redistributing assets lies in the differences in the terms and conditions of free software licenses. The Free Software Foundation, which promotes completely free software, recognizes a list of licenses (e.g. Apache License, Eclipse Public License, Mozilla Public License) as free, although incompatible with the GNU General Public License.[4] This philosophy of free and open source software is described in more detail in Chapter 2.

The rise in open source software occurred late in the 1990s, as evident in the report recommending development of a strategy to the IBM Corporate Technology Council in 1999 (Capek et al. 2005). For the purposes of this study, January 2001 is seen as a watershed date, as IBM announced the plan to invest $1 billion in Linux over the following three years, initially deploying 1000 employees (IBM 2001, 21). Ten years in hindsight, the commitment to Linux has been judged as delivered (Gabriel Consulting Group 2008).

1.1.2 Impact: Pioneering companies have shaped organizational, industry and social contexts to make open source viable for private commercial businesses

Beyond the licensing of software code as open source, companies have actively participated and supported open source. In software development, the contribution of human resources is as important as the resulting assets. In 2007, 31% of administrators and users of 409 Sourceforge.net project declared that one or more firms were somehow involved. In 68% of the cases, firms supported non-development activities (e.g. testing, animating, forums, documentation, financial and logistic support), and 30% contributed code) (Capra et al. 2009). In 2009, 48% of Eclipse developers were allowed by their companies to use software and contribute back to open source communities, up from 37% in 2007 (Eclipse Foundation 2009, 18).

Sponsorship of organizations independent of vendors enables open source to grow. The Linux Foundation lists Fujitsu, Hitachi, IBM, Intel, NEC, and Oracle as platinum members; AMD, Cisco, ETRI, Google, HP, Motorola, NetApp, Nokia and Novell as gold members; and 44 additional companies as silver members.[5] The Eclipse Foundation lists Cisco, Motorola and Blackberry as enterprise members; Actuate, Brox, CA, Cloudsmith, Genitec, IBM, Innoopract, Itemis, Nokia, Obeo, Oracle, SAP, Sonatype, and Sopera as strategic members; and 72 organizations as solution members; and 71 organizations as associate members.[6]

4 See http://www.gnu.org/licenses/license-list.html#GPLIncompatibleLicenses . The Open Source Initiative discourages "license proliferation" by encouring developers to adopt one of the more popular existing licenses, at http://www.opensource.org/proliferation-report .

5 See http://www.linuxfoundation.org/about/members . Both platinum and gold members "engage in or support the production, manufacture, use, sale or standardization of Linux or other open source-based technologies", with annual membership dues of $500,000 or $100,000, as specified at http://www.linuxfoundation.org/about/bylaws .

6 See http://www.eclipse.org/membership/exploreMembership.php . Enterprise members "rely heavily on Eclipse technology"; strategic members "view Eclipse as a strategic platform and are investing developer and other resources"; solution members "offer products and services based on, or with, Eclipse"; and associate members "want to show support for, the Eclipse ecosystem".

Legacy procedures for intellectual property ownership have been adapted to protect both contributors and users using assets declared as in the commons. IBM has committed, in a series of pledges, legally binding commitments not to assert patents over a wide range of software.[7] In January 2005, IBM pledged open access to 500 software patents, as a basis for forming a “patent commons”. With Novell, Philips, Red Hat and Sony, it co-founded the Open Innovation Network to acquire patents and offer them royalty-free, provided those patents would not be asserted. In October 2005, IBM pledged royalty-free access to selected open healthcare and educational software standards build around web services, electronic forums and open document formats. In July 2007, IBM pledged patents on 150 core software interoperability standards including a broad assortment of XML-based technologies (e.g. XSL, OpenDocument). In January 2008, IBM, Nokia, Pitney Bowes and Sony partnered to found the Eco-Patent Commons with the World Business Council for Sustainable Development, each pledging environmentally responsible patents to the public domain.

The commitments by companies to the open source movement do not represent spin-off or subsidiary activities. Most employees do not work exclusively either in open source **or** private source; they work in both open source **and** private source. Working both in open source and private source is relatively uncommon in business, and requires shifts in mindsets and practices.

1.1.3 Inquiry: What can we infer from IBM's learning on open source with private source that may be applicable in other situations, or for other organizations?

In the decade after 2000, IBM has generally been regarded as a successful company. At 2009, it declared “Since the dot-com crash of 2002, we have added $12 billion to IBM's pre-tax profit base, increased our pre-tax margin 2.5 times, quadrupled our earnings per share and more than doubled our free cash flow” (IBM 2009, 3). As a reflection, the company has stated that “A decade ago, we saw change coming. [....] A new computing architecture was taking shape. It was build on pervasive instrumentation and interconnectivity, open standards, unprecedented computing power and advanced analytics” (IBM 2009, 10). Part of open standards has been participation in the open source movement.

This study takes a micro approach to understanding the business, focused on nine case studies where both open source and private source behaviours have been exhibited. These cases are applied inductively towards theory building. Full development of a theory requires generalization to a larger set of cases (i) within the company, (ii) across the industries in which the company engages, and/or (iii) across a broader variety of enterprises in for-profit and/or not-for-profit businesses.

With this description of the research question, we jump to the conclusion of the study, with an outline of the findings.

7 The list of pledges, to 2008, are cited by Bob Sutor, “Announcement: The Eco-Patent Commons”, January 14, 2008, at http://www.sutor.com/newsite/blog-open/?p=2016 .

1.2 Conclusion: Open source with private source presents new opportunities for commercial success in business when new ways are learned and become natural

The overall conclusion from this study is that open source does not have to be incompatible with private source in commercial businesses. From a behavioural perspective, operating simultaneously in open source and private source has to be learned as a natural way of working for both practitioners who have contribute to each, and for leaders who set policy and guide behaviours in inter-organizational relationships.

Editorial note: Descriptions of Chapters 3 to 9 have been edited out from this Arctic Workshop submission.

These foundational concepts are explained in more detail in Chapter 2.

2. Foundations: open source and private source

This research study rests on some foundational concepts, where misinterpretations based on common English usage may result in confusion. This chapter attempts to clarify the use of these concepts, in context. The concepts are:

- *open source*, which is opposed with *private source* [section 2.1]

While the origins of the label of open source are from the software development community, the promise of open source transcends that categorization. Open source, with an opposite of private source, can have a wider applicability in business and society, used both in a sense of contrast (i.e. open source versus private source) and in a sense of combination (i.e. open source with private source).

2.1 Open source and private source: (Business) systems can (inter-)operate with visible and/or hidden internals

At the advent of the label of *open source*, the focus was on the nature of software development. Even in those earliest discussions, however, there was recognition that the phenomenon had implications beyond the licensing of computer code. This wider perspective on open source is adopted in this study. The reasoning follows in four sections.

- the history and meaning of the *open source* label [section 2.1.1];
- source reserved as private through information hiding as an option in modular system design [section 2.1.2]; and
- *private source* as an appropriate opposite to open source in the context of commercial business [section 2.1.3].

This section aims to generalize the concept of open source and private source to a world seen either as a networks of systems, or system of systems.[8]

8 A system of systems would have function (potentially purpose), structure and process. A network of systems may only have structure and process, with no purpose in whole.

2.1.1 *Open source,* as a label originating 1998, has meaning beyond software code

Source code is a set of instructions written in human-readable form. Articulated in a programming language, source code is targeted to execute on a computer in one of two ways: (i) at invocation, an interpreter streams the source code to translate into machine-readable instructions (i.e. following line by line, including re-interpreting instructions repeated in a control flow loop); or (ii) build-time is a stage distinct from run-time, with a compiler translating source code into object code executable on a specific machine platform (e.g. the same source code might be translated into Windows object code or Linux object code).

Distributing object code has the benefits that (i) program execution is more efficient, and (ii) the recipient of the program can immediately put it to use. Object code is rarely modified directly, as few people learn machine-level programming, so the logic that clearly readable in higher-level languages becomes obscured in binary code. When computer power is a precious resource (e.g. transactional volumes are high, or processing bandwidth is low) – centralized compilation to object code is preferred.

Editorial note: Section 2.1.1 has been excised to shorten the research paper for Arctic Workshop 2012.

The ideas of source code and object code are as old as writing and mechanization. Musical scores – printed musical notation often known as sheet music – are a form of source code that one or more musician(s) interpret to perform a composer's work. A player piano can execute object code programmed on perforated paper; an electronic synthesizer can execute object code through MIDI (Musical Instrument Digital Interface); a compact disc player can execute object code recorded onto an optical disc.

2.1.1.1 "Open source" was inspired by "The Cathedral and the Bazaar", at the open release of Netscape Communicator source code

In September 1997, Eric Raymond presented "The Cathedral and the Bazaar" at the O'Reilly Perl Conference (Raymond 2000).[9] He described "two fundamentally different development styles, the *cathedral* model of most of the commercial world versus the *bazaar* model of the Linux world". Based on his experience with the *fetchmail* project, Raymond listed lessons on motivations and practices that had been successful in the distributed collaboration. In July 1999, he appended a chapter "On Management and the Maginot Line", where he reflected on the role of project manager and the new context of "cheap PCs and fast Internet links" with volunteers leading to self-selection and self-organization.

Editorial note: Section 2.1.1.1 has been excised to shorten the research paper for Arctic Workshop 2012.

At a strategy session in Palo Alto on February 3, 1998, open source emerged as a business-oriented label that superseded some of the philosophical positions adopted by the free software movement.

9 The history of "The Cathedral and the Bazaar", Netscape's announcement and the strategy session of February 3, 1998 are documented at http://www.opensource.org/history . The revision 1.27 note of February 9, 1998 replaces the original phrase "free software" with "open source"

Editorial note: Section 2.1.1.1 has been excised to shorten the research paper for Arctic Workshop 2012.

Open source is related to innovation. If the original source was never updated or modified, its openness would be irrelevant. The innovation can be related both to technological progress and social progress.

> Technological innovation is more than the production of improved functionality. In open source projects it is easy to see that striving for the common good is one of the reasons why open source developers commit themselves to a development project. Although the common good is evaluated based on the internal values of the community, even a small contribution can become important when it becomes part of a bigger system. [....]
>
> Innovation, therefore, has its deep roots in the processes of individuation. socialization, and meaning construction. We use language, signs. and tools, and integrate them in our thinking and action. In this sense, human beings are technological beings. Fundamentally, technological change, therefore, relates to questions concerning the way we exist in the world. As technologies and technological change become visible in our everyday life, the foundations of technology also will be increasingly in our focus (Tuomi 2002, 219).

The first decade of the 21st century has therefore seen open source not only as a way of developing software code, but more broadly, a way individuals can make contributions towards common interests.

2.1.1.2 The variety of licensing conditions details options available to authors and (re-)distributors

While some altruistic individuals are willing to make contributions towards a common effort without concerns about ownership or liability, the pragmatic are more cautious. On a trajectory parallel to the software development world, the Creative Commons has formalized copyright licenses in a broader variety of domains, inspired from the experiences in the open source movement.[10]

Editorial note: Section 2.1.1.2 has been excised to shorten the research paper for Arctic Workshop 2012.

2.1.2 Private source reserves a privilege through information hiding in a modular system design

While the origins of the label *open source* can be described with a rich history, an explanation of *private source* is more analytical. Although private source has had a meaning in computer science, the relevance of the label has risen since 1999 with open source. Since open source and private source are related to parts of a greater whole, questions of interoperability in larger containing systems are raised. Pragmatic individuals should recognize that private source, just like open source, has its advantages and disadvantages.

2.1.2.1 Private source, in opposition to open source, is an under-appreciated label

Private source is an underused label that can clarify the spirit (rather than legal terms) in collaborative development and innovation. On a web search, one of the earlier

10 The Creative Commons cites the Free/Libre Open Source Software (FLOSS) popularization from a June 2001 letter to the European Commission, combining terms from the Free Software Foundation and Open Source Initiative. See http://creativecommons.ca/index.php?p=cc101 .

public appearances of *private source*, in opposition to open source (after the 1999 definition) is by IBM in August 2006, at the Linux World Conference.[11]

In computer science, the label has also not been popular, but tracks back with a longer history. In 1975, an article on "source statement libraries" depicts a period when computer programming was moving from punch cards to magnetic storage. The use of the label "private source" as "not available to just any user" is an acknowledgement of the obsolescence of physical records (i.e. statements punched onto paper cards) to electronic storage (i.e. magnetic disk) to which access privileges could be programmed as open or private (Flores and Feuerman 1975).

Private property, in opposition with ownership put into the public domain, can be either associated with, or decoupled from, private source and open source. Incorporated businesses can separate control from ownership, creating "powers in trust"[12]. In the advancement of developmental domains (e.g. technology, creative works), development and adoption relies on mutual conduct by authors/creators, distributors, and users that is productive for all parties. Issues arising on claims associated with tangible properties become even murkier when applied to intangibles.

2.1.2.2 Open source and private source can interoperate through interface specifications, even when internals are hidden

While utopians might like the whole world to be open source, nature itself retains its mysteries.[13] In man-made (artificial or artifactual) systems, parts of the whole may be open source or private source.[14] This study takes the perspective that open source and private source are relevant not only for technologies (i.e. mechanical systems), but also for social systems, for both informal communities and incorporated organizations.

Some basic foundations in systems theory can clarify relations between wholes and parts. "A *system* is a whole that can not be divided into independent parts" (Ackoff 1994, 21).[15] A *subsystem* is "an element or functional component of a larger system which fulfills the conditions of a system in itself, but which also plays a specialized

11 "Private source" appears in a press release "IBM Unveils Development Roadmap and Business Strategy for Open Source Beyond Linux" at Linux World Conference and Expo, in San Francisco, August 15, 2006 at http://www-03.ibm.com/press/us/en/pressrelease/20131.wss .

12 The separation of powers of control and ownership are described in (Berle and Means 1991) in Book 4, Chapter 1, "The Traditional Logic of Property".

13 Gregory Bateson describes five levels of learning, of which the top level is beyond conscious human capability as genetic change, described as phylogenesis (of the the tribe or species) with ontogenesis (of the individual living being). See (Bateson 1972), The Logical Categories of Learning and Communication.

14 The idea of a science of the artificial was developed in (Simon 1996). The distinctions between artificial and artifactual are described at http://coevolving.com/blogs/index.php/archive/science-of-service-systems-as-an-artifactual-science/ .

15 This definition is "summarizing and oversimplifying". More rigourously, "A system is a whole that contains two or more parts that satisfy the following five conditions:
1. The whole has one or more defining functions.
2. Each part in the set can affect the behavior or properties of the whole.
3. There is a subset of parts that is sufficient in one or more environments for carrying out the definition function of the whole; each of these parts is separately necessary but insufficient for carrying out this defining function.
4. The way that the behavior or properties of each part of a system affects its behavior or properties depends on the behavior or properties of at least one other part of the system.
5. The effect of any subset of parts on the system as a whole depends on the behavior of at least one other subset. (Ackoff 1994, 18–21)

role in the operation of the larger system" (François 1997, 336).[16] In recent years, the label of a *system of systems* has been applied to complex arrangements where a larger whole incorporates independently viable parts. As an example, a personal computer is a system, and the Internet is a system of systems. Thus, a personal computer can be a subsystem of the Internet. Similarly, a neighbourhood may operate as a system, with a city as a larger containing system of systems.

The definition of the boundaries of an *open system* can be complicated. "Most concrete systems have boundaries which are at least partially partially permeable, permitting sizeable magnitudes of at least certain sorts of matter-energy or information transmission to cross them. Such a system is an open system (in which) entropy may increase, remaining in steady state or decrease" (François 1997, 252).[17] In reality, system boundaries are not defined by nature, but by the (human) observer. Systems concepts are a means that enable human beings to understand relations.

In a mechanical system (e.g. a technology), the mapping of *components* to *subsystems* is relatively linear. A component is "a unit of a system that in combination with other units functions to combine, separate or compare the inputs to produce outputs" (François 1997, 69).[18] A system can be decomposed into parts as independent components, and into parts as subsystems that may include part-part and other part-whole interactions. "If a system does have subsystems, not just components, then the cohesion of the subsystems competes with that of the overall system (François 1997, 336).[19] The mapping of components to subsystems is more complicated in a social system than in a mechanical system.

Editorial note: Section 2.1.2.2 has been excised to shorten the research paper for Arctic Workshop 2012.

Information is a feature of systems that has a property of abstraction. *Abstraction* is "the isomorphic transformation from an interpreted system into the corresponding general system" (François 1997, 17).[20] Digitalization enables the power of abstraction in information (sub)systems. Aircraft now "fly by wire" with electronic interfaces between subsystems rather than only mechanical linkages. The volume of money flowing in an economy dwarfs counts of paper currency that are no longer tied to physical standards (e.g. gold). Modularizing a system into components by abstraction reduces complexity for systems designers (and possibly system users).

> Abstraction is a technique for managing complexity that is deeply ingrained in human beings. As information processors we are always suppressing details we take to be unimportant; as problem solvers, we instinctively focus on a few manageable parts of a problem at one time. If our abstractions match the true underlying structure of the problem – if we are good carvers, not bad ones – then the analysis of abstractions will lead to a good solution to the problem. In any use, given our limited mental capacities.

16 This encyclopedia definition cites Csanyi, Vilmos: "Culture and society as a replicative system", in Banathy, B. & Rodrigues Dalgado, R (eds.): Intern. Syst. Sc. Handbook, Systemic Publ., Madrid 1993, p. 67.

17 This encyclopedia definition cites Collen, Arne et al:. "Logical openness in systems". Sys. Research, 11 (2), 1994, pp. 65-58.

18 This encyclopedia definition cites Berrien, Kenneth F.: "General and social systems". Rutgers Univ. Press, New Brunswick 1968, p. 32.

19 This encyclopedia definition cites Bunge, Mario: "Causality and modern science". Dover, New York, 1979, p. 37.

20 This encyclopedia definition cites Klir, George (ed.): "Facets of Systems Science". Plenum Press, New York, 1991, p. 17.

> we have no choice but to work with simplified representations of complicated problems (Baldwin and Clark 2000, 73).

Abstraction separates “knowing that” a component will reliably carry out a function from “knowing how” the internals of that component work. Another way of reducing complexity is through *information hiding*.

> The principle of *information hiding* was first put forward in the context of software engineering by David Parnas. However, the principle is perfectly general, and can be usefully applied to any complex system. With respect to software programs, Parnas reasoned that if the details of a particular block of code were consciously “hidden" from other blocks. changes to the block could be made without changing the rest of the system. The goal was then to enumerate and as far as possible restrict the points of interaction between any two modules of a program. The fewer the points of interaction. the easier it would be for subsequent designers to come in and change pans of the code, without having to rewrite the whole program from scratch.
>
> "Information hiding" is closely related to the notion of abstraction defined above:
>
>> When the complexity of one of the elements crosses a certain threshold, that complexity can be isolated by defining a separate “abstraction" with a simple interface. The abstraction '”hides" the complexity of the element ....[21] (Baldwin and Clark 2000, 73)

Private source is a design choice that includes information hiding. Generally, more people are interested in (i) “knowing that” a component will satisfy their needs, rather than (ii) “knowing how” that component was designed and constructed. Further, human beings often don't want to be overloaded with “knowing how”, as they are boundedly rational in their decisions .[22] In man-made systems, modularization of design enables open source components and private source components to interoperate, as long as interface specifications are available.

> For a modularization to work in practice, the architects must partition the design parameters into two categories: *visible information* and *hidden information*. This partition specifies which parameters will interact outside of their module, and how potential interactions across modules will be handled. [....]
>
> Information hiding begins as an abstraction. But to achieve true information hiding, the initial segregation of ideas must be maintained throughout the whole problem-solving process. This means that as points of interaction across problem boundaries arise, they cannot be dealt with in an ad hoc way. Instead, the interactions must be catalogued and a set of interfaces specified.
>
> An *interface* is a preestablished way to resolve potential conflicts between interacting parts of a design. it is like a treaty between two or more subelements. To minimize conflict, the terms of these treaties – the detailed interface specifications – need to be set in advance and known to the affected parties. Thus interfaces of a common information set that those working on the design need to assimilate. Interfaces are visible information (Baldwin and Clark 2000, 73).

---

21 This passage in Baldwin and Clark (2000) is cited to Parnas, David L. (1972), “Information Distribution Aspects of Design Methodology”, Proceedings of IFIP Congress 71, North-Holland, pp. 339-344.

22 Bounded rationality is one of the “models of man” developed by Herbert Simon from the 1950s into the 1970s.

Component interface specifications are a sufficient (and encouraging) condition for interoperability in a larger containing system.[23] The declaration of specifications is not a necessary condition, with the option of reverse engineering possible with some effort. If the source code has been lost or forgotten, interface behaviours can be rediscovered. If further development of the component is desirable, construction of a "clone" hardware or software may result in a variety of compatible substitute legacy systems, as well as a foundation for additional interoperability specifications going forward.

2.1.2.3 Private source can simplify development, in the absence of industry-accepted standards

Interoperability among system components requires cooperation between system designers. The responsibility for interoperability may be assumed unilaterally, bilaterally, or even multilaterally. In complicated multilateral arrangements, an open source approach can accelerate attainment of common goals. Open source is an ideal form of information transparency. "Information transparency is defined as the degree of visibility and accessibility of information" (Zhu 2002, 93).[24] While external collaborators may or may not have the authority to modify the internals of a component, they should at least be able to point out issues or incompatibilities that may arise during joint development. "Given enough eyeballs, all bugs are shallow" (Raymond 2000). Responsibility distributed across multiple components requires synchronization, so that sequential coordination of "moving targets" can be precluded.

Editorial note: Section 2.1.2.3 has been excised to shorten the research paper for Arctic Workshop 2012.

2.1.3 A definition of open source that acknowledges interoperability with private source encourages contributions from both individual and commercial participants

Open source and private source are social constructs. The vitality of the associated communities is sustained through constructive conduct in self-governed behaviours. Open source respects contributions by volunteers. Private source respects ownership claims by creators and entrepreneurs. The enforcement of licensing terms and challenges via legal proceedings on intellectual property are relatively infrequent, as litigation is expensive. The adoption of open source by commercial businesses has benefited from (a) understanding of open source as "free as in liberty"; (b) constructive contributions by private enterprises, and (c) continuing coexistence of private source and private source in commercial business models.

23 Component interface specifications, as standards have been described as the most important "open". "[Open] has become associated with software source code, industry standards, developer communities and a variety of licensing models – four distinct phenomena that are often intermingled in indistinct ways. [....] Of the four, open standards are the most critical, because making a choice today shouldn't preclude you from making a different choice tomorrow" (Schwartz 2003).

24 Zhu's definition of information transparency was written in the context of B2B exchanges. The application of the definition here speaks to its generality.

2.1.3.1 Like the free software movement, open source means free as in liberty rather free as in gratis

Since the late 1990s, the idea of *free software* has flourished in the open source movement.[25] The interpretation of free, however can be take in two contexts: free as in *liberty*, and free as in *gratis* (i.e. out of favour or kindness, without charge, cost or pay). Amongst software developers, this distinction is known as “free as in speech”, and opposed to “free as in beer”, based on an 1999 panel discussion including Eric Raymond, Richard Stallman and Linus Torvalds, describing different philosophies.

Editorial note: Section 2.1.3.1 has been excised to shorten the research paper for Arctic Workshop 2012.

Both the Free Software Definition and Open Source Definitions encourages technical interoperability, i.e. one software package should work with the other. By specifying that Free Software is only for gratis, however, social interoperability is limited: software developers can engage in commercial relationships in restricted ways.[26] The Open Source Definition was constructed with social interoperability between non-commercial and commercial parties in mind.[27]

2.1.3.2 The open source movement can and has benefited from contributions by private enterprises

The open source movement has benefited from corporate contributions that are “free as in liberty” as well as “free as in gratis”.

Editorial note: Section 2.1.3.2 has been excised to shorten the research paper for Arctic Workshop 2012.

2.1.3.3 The open source definition provides a liberty of potential profits for commercial business models

While the open source definition permits commercial use, the ways in which profit can be gained is left to creative entrepreneurs. In the domain of software, money may be made when open source is packaged with private source software, complementary services, or complementary hardware.

Editorial note: Section 2.1.3.3 has been excised to shorten the research paper for Arctic Workshop 2012.

2.1.3.4 Simile: Private source is akin to a trade secret; open source is akin to disclosing a standardized recipe

While private source and open source are new labels, they reflect a long-standing choice by businesses: should an idea be retained as a trade secret[28], or does wider

25 (Tuomi 2002) describes the contribution of Linux to the open source movement. (Weber 2004) describes the rise of open source in greater detail.

26 Section 2.1.3.3 points out that charging to operate or support Free Software is allowed, although charging for redistribution is not.

27 Bob Sutor creates a distinction between *interoperability* where standards do not favour any specific party, as opposed to *intraoperability* where one party becomes central and dominant. See http://www.sutor.com/newsite/blog-open/?p=1260 .

28 Definitions of trade secrecy can be devolved to other scholars. Josh Lerner provides a helpful summary at http://www.people.hbs.edu/jlerner/TSintro.html . “The definition of trade secrecy with the widest acceptance is that in the American Law Institute's Restatement of Torts [1939]:
'A trade secret may consist of any formula, pattern, device or compilation of information which is used in one's business, and which gives him an opportunity to obtain an advantage over competitors who do not know or use it. .... A substantial element of secrecy must exist, so that, except by the use of improper means, there would be difficulty in acquiring the information.'

disclosure lead to negligible economic harm and/or potential benefits (e.g. licensing fees)?

A trade secret is a protection against misappropriation, e.g. an employing taking a formula to a competitor for replication. Even without an employee breaching a trade secrecy agreement, a diligent analyst can conduct reverse-engineering. In the recipe of "11 secrets herbs and spices" for the original Kentucky Fried Chicken, analysis has led a conclusion that ingredients only included four spices and no herbs (Poundstone 1983, 20).[29] In food service operations, standard recipes are shared with employees as a process for maintaining quality and costs, but they are only part of the complete dining experience. An alternative form of intellectual property protection is a patent granted on a design, that requires disclosure in the filing process, potentially leading to enforcement over for a defined period.

Codified information is generally covered instead by copyright. Copyright may or may not be practically enforceable on the product, services and infrastructure and/or interpersonal relationship dimensions of an offering. Information has peculiar attributes that impacts its economics.

Editorial note: Section 2.1.3.4 has been excised to shorten the research paper for Arctic Workshop 2012.

Private enterprise has a long history associated with the idea of proprietary ownership, trade secrets, and exclusive relationships. An open source approach based on collaboration presents opportunities for advancements both for society and for businesses. The industrial age pattern of secrecy is strong in western civilization, even permeating education. In the rise of social media, educators have been challenged by questions as to whether homework should or should not be shared. Collaboration is generally regarded by teachers with suspicion.

Editorial note: Section 2.1.3.4 has been excised to shorten the research paper for Arctic Workshop 2012.

This research study sees the major challenges in open source with private source not as economic or financial, but in social practices. Businesses and institutions originating from the industrial era are likely to exhibit inertia in desires to maintain ways that have previously proven successful for them. As they become less relevant and/or viable, their motivation to embrace open source with private source should rise.

Bibliography

Ackoff, Russell L. 1994. *The Democratic Corporation*. New York: Oxford University Press.

Asay, Matt. 2009. "World's Biggest Open-source Company? Google." *The Open Road - CNET News*. http://news.cnet.com/8301-13505_3-10354530-16.html.

Aslett, Matthew. 2009. "On Open Source Business Strategies (again)." *451 CAOS Theory*. http://blogs.the451group.com/opensource/2009/02/23/on-open-source-business-strategies-again/.

Baldwin, Carliss Young, and Kim B. Clark. 2000. *Design Rules: The Power of Modularity*. MIT Press.

Bateson, Gregory. 1972. *Steps to an Ecology of Mind*. Northvale, NJ: Jason Aronson.

Trade secrecy is quite different from other forms of intellectual property protection."

29 Todd Wilbur, author of Top Secret Recipes, believes that consumer preference for juicy and moist chicken can largely attributed to the ten-minute pressure cooking process. See http://www.topsecretrecipes.com/recipedetail.asp?id=874 .

Berle, Adolf Augustus, and Gardiner Coit Means. 1991. *The Modern Corporation and Private Property*. Transaction Publishers.

Capek, Peter G., Steven P. Frank, Steve Gerdt, and David Shields. 2005. “A History of IBM’s Open-source Involvement and Strategy.” *IBM Systems Journal* 44 (2): 249–257. http://domino.research.ibm.com/tchjr/journalindex.nsf/495f80c9d0f539778525681e00724804/16428e226ac9f64a85256ff800672a0d?OpenDocument.

Capra, Eugenio, Chiara Francalanci, Francesco Merlo, and Cristina Rossi Lamastra. 2009. “A Survey on Firms’ Participation in Open Source Community Projects.” In *Open Source Ecosystems: Diverse Communities Interacting*, 299:225–236. IFIP Advances in Information and Communication Technology. http://dx.doi.org/10.1007/978-3-642-02032-2_20.

Eclipse Foundation. 2009. *The Open Source Developer Report*. Eclipse Community Survey. http://www.eclipse.org/org/press-release/20090527_survey09.php.

Flores, I., and M. Feuerman. 1975. “Source Statement Libraries and IBM System/370.” *Computer Languages* 1 (2) (June): 139–150. doi:10.1016/0096-0551(75)90013-2. http://www.sciencedirect.com.myaccess.library.utoronto.ca/science/article/B6TYK-48V1XCV-7F/2/f08d912f7b633a110b62a6789581567c.

François, Charles. 1997. *International Encyclopedia of Systems and Cybernetics*. Munich: K. G. Saur.

Gabriel Consulting Group. 2008. *IBM & Linux – 10 Years Later*. ftp://ftp.software.ibm.com/linux/pdfs/GCG_IBM_and_Linux-9_years_later.pdf.

García-García, Jesús, and Mª Isabel Alonso de Magdaleno. 2010. “Commons-based Innovation - The Linux Kernel Case.” In Seville, Spain. http://iri.jrc.ec.europa.eu/concord-2010/posters.html.

IBM. 2001. *Annual Report 2001*. Annual Report. http://www.ibm.com/annualreport/.

———. 2009. *Annual Report 2009*. Annual Report. http://www.ibm.com/annualreport/.

Poundstone, William. 1983. *Big Secrets: The Uncensored Truth About All Sorts of Stuff You Are Never Supposed to Know*. Morrow.

Raymond, Eric S. 2000. “The Cathedral and the Bazaar.” http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/.

Schwartz, Jonathan. 2003. “Open Source Versus Open Standards.” *CNET News*. http://news.cnet.com/2010-1071-995823.html.

Simon, Herbert Alexander. 1996. *The Sciences of the Artificial*. MIT Press.

Tuomi, Ilkka. 2002. *Networks of Innovation : Change and Meaning in the Age of the Internet*. Oxford, [England]; New York: Oxford University Press.

Weber, Steve. 2004. *The Success of Open Source*. Cambridge, MA: Harvard University Press.

Zhu, Kevin. 2002. “Information Transparency in Electronic Marketplaces: Why Data Transparency May Hinder the Adoption of B2B Exchanges.” *Electronic Markets* 12 (2): 92–99. doi:10.1080/10196780252844535. http://dx.doi.org/10.1080/10196780252844535.