

Title:

The Economic Dynamics of Software: Three Competing Business Models Exemplified through Microsoft, Netscape and Linux

Short Running Title: The Economic Dynamics of Software

Author: Dr. Maureen McKelvey

Address: Department of Technology and Social Change; 581 83 Linköping University, Sweden;
maumc@tema.liu.se

Abstract:

This article proposes three ideal business models to analyze innovation in knowledge-intensive goods and services. The three models are 1) Firm-based control, 2) Hybrid, and 3) Network-based. Each is defined in relation to the two sides of innovation, e.g. creation of novelty and of economic value. Defining the models this way leads to a discussion of the advantages and disadvantages of each model for organizing the development of different types of software and for appropriating economic benefits. Each business model is also exemplified through the economic history of one example. The examples are, respectively, Microsoft, Netscape and Linux. The concluding section relates software development to the broader forms of economic dynamics in knowledge-intensive sectors.

Key Words:

Innovation; Evolutionary Economics; Microsoft; Netscape; Linux; Software

JEL classification: L86; O31, O32; O33

1. INTRODUCTION

This article analyzes three ideal business models which are visible in the new, information-based economy. Most agree that some knowledge intensive activities create high economic value, leading to a great interest in high tech, or research and development (R&D) intensive, sectors (Abramovitz 1989). Although knowledge is often assumed to create economic value in general or through spill-overs (Romer 1990, Krugman 1987), there is, in fact, a variety of ways both to develop and diffuse new knowledge and to capture – or not – the economic returns (Nelson 1996). This theoretical argument implies that several competing business models can exist at the same time. The existence and the implications of such diversity has not been sufficiently explored in the existing economic and innovation literature. What commonly happens is that authors proclaim a 'new' business or economic model, like (Malone and Laubacher 1998), where the new is assumed to replace the old. In contrast, this article argues that a diversity of business models compete in the modern information economy and that they have different technical and economic advantages, and disadvantages, for innovation.

Early in the 20th century, innovations and entrepreneurs played a central role in Joseph A. Schumpeter's theories of economic development (Schumpeter 1979/1943), which have experienced a revival in recent decades (Freeman 1994). Some of Schumpeter's influence comes from his recognition that there was a difference between an invention and an innovation. An invention is a technical novelty whereas an innovation is something of economic value and/or involved in a commercial transaction. In short, there is a distinction between how knowledge develops and how money may be made. Schumpeter's economic analysis also implies that the process of changing an invention into an innovation is a process. Innovation process is an uncertain process involving different sorts of persons, where entrepreneurs lead the way in creating novelty while imitators follow after and diffuse the novelty. As has been further developed within evolutionary and neo-Schumpeterian economics, innovation processes may be carried out in different ways, with different models of knowledge and relationships being important for the creation of economic value (Dosi et al eds 1988; McKelvey 1996).

This article therefore analyzes different ways of inventing and innovating in software development, based on three ideal business models. The three are defined in relation to the two sides of innovation, e.g. creation of novelty and of economic value. The article analyzes and compares the three models in ways of innovating: 1) Incentives to engage in knowledge-seeking activities, 2) Ways of organizing the creation of novelty, and 3) Strategies and possibilities to make profits. The analysis here is of software development over time, including how firms and individuals manage, react to, and/or create changes in a dynamic technical and economic environment. In short, the focus is on the economic dynamics of software. Here, the ideal model models are exemplified through software and services created around Microsoft, Netscape and Linux.

The next section introduces the three ideal business models, then the subsequent three sections present each model and example, respectively, along a series of variables. In addition to being examples, however, there are specific interactions among these three software bundles – Microsoft, Netscape, and Linux - that are historically specific and which indicate interesting outcomes about the relative advantages, and disadvantages, of each when they compete over time. The final section relates these results to the broader innovation and economics of technical change literature.

2: THREE COMPETING BUSINESS MODELS FOR KNOWLEDGE-INTENSIVE GOODS AND SERVICES

Three competing business models are presented here, in terms of major characteristics and outcomes for technical and economic variables. They are argued to be useful to understand knowledge-intensive sectors, although the examples given are of software development. Software is a useful case because competition among software takes a variety of forms. It sometimes seems more like sales of standardized industrial goods, coordinated through market transactions, and sometimes more like development of public and freely available knowledge.

What differentiates knowledge intensive products from many other industrial goods is, obviously the relative economic importance of knowledge as compared to physical and distributional aspects. Packaged software is basically a set of codified knowledge (code, structure of the program) plus instructions (manuals; help services and products) and packaging. Other software may appear even more knowledge-intensive.

A highly relevant phenomena is that in software, the different models can compete, complement, and merge with each other over time. This helps us further understand how public, and private, knowledge interact and how bits of each may be useful for economic competition in other knowledge-intensive sectors.

The three ideal business models presented here are:

1) *Firm-based control of knowledge and of the economic returns.*

This would be a firm selling software as a standardized, closed, mass market product at a given time, albeit a product whose boundaries may expand over time into new uses and services. Software is a product, requiring strategies to sell it, and the firm retains control over software development as well as over the economic returns.

2) *Hybrid. Firm-based control of economic returns to a service and/or good but a hybrid of firm and network-based development of knowledge.*

This would be a case where the firm tries to take existing knowledge (or infrastructures) and to create control by selling software, or internet-based services, as a means to try to capture some of the economic

benefits. Software development should be a hybrid between in-house firm and network based development.

3) *Network-based development of knowledge, with node control, which is not linked to firm-based control of economic returns.*

This would be software development which develops through open networks, much like public knowledge, although there are nodes of control.¹ Development runs parallel, or else instead of, commercial proprietary development. Here, both individuals, communities of programmers and firms can be important for different aspects of organization and control.

The examples given of the three business models are, respectively, Microsoft, Netscape and Linux. These three are, or have been, direct competitors over software, where they are competitors either in terms of users and/or of market share. The three software bundles compete in related areas of software for PCs, ranging from operating systems to applications to browsers. Operating systems allow communication between the machine hardware while programs, or applications, which run tasks which interface to the users. Microsoft's Word (NT, 97, 2000) is a well known operating system, whereas AUTOCAD has drawing applications. The main reason why operating systems are important is well summarized by Torvalds, initial inventor of Linux.

One thing that makes operating systems special is that it is extremely hard to change them. Changing an operating system means changing everything from under you. So changing an operating system is like trying to go in and transplant a person's brain....Once you have the operating system niche, you're really home free. That's what people are worried about. It implies that you can leverage your other products on top of that. It also means that you are free to do whatever you want to. And few people are going to switch because most people are going to just tag along and take what ever you give them (*InfoWorld* 1998a).

The crucial importance of the operating system is an important reason for the American Department of Justice anti-trust suit against Microsoft. In 1998, the only operating system gaining ground on Microsoft was Linux, and many felt that was for corporate systems users, not PC users (*Economist* 1999d). Browsers, search engines, and portals all facilitate use of the internet, such as Internet Explorer, Netscape, Yahoo! American on Line (AOL) and other internet service providers allow users to get onto, or port into, the WWW (world wide web). The three examples used represent three very different ways of organizing the development of software as well as its distribution and its commercialization.

The competition among these three software bundles illustrate the three ideal business models because they exhibit a range of organisational forms for either integrating, or else separating, creation of novelty and creation of economic value. In these example, creation of novelty means developing knowledge relevant to the software development and creation of economic value means appropriating economic returns. The differences between the models are visible in how much and what models of control are maintained over software development inside the firm as well as the opportunities for creating profits. Innovation opportunities are in turn

related to the number and types of users willing to use and/or develop the software. What is particularly interesting is that each ideal model seems to result in a particular type of market/user base and particular type of product. As analyzed in the following sections, each ideal model has advantages and disadvantages in relation to the speed and rate of software development and in terms of commercialization.

Despite arguing that diversity is evident in that three business models competing in modern economic history, there is also an element of dynamic change over time. It appears that any one firm may choose to move towards a different business model, over time, or that a competing software which continues to develop will stimulate firms to follow a different model, too. For example, there are incentives for firms following the firm-based control to move towards a network-based model when they can no longer control (or afford) software development or cannot appropriate returns. Similarly, there are incentives for individuals to move out of a network-based software development into a hybrid model firm, in the pursuit of some economic returns. In other words, software development which is driven through public domain software may converge with commercial software and products and/or it may stimulate attempts at firm-based control of economic returns. As with all ideal business models, the three proposed here try to represent generalizable characteristics and relationships from specific historical cases. The economic dynamics of software imply, however, that any given example may transform itself over time.

3: FIRM-BASED CONTROL IDEAL MODEL

The first ideal model is that of firm-based control of knowledge and of economic returns. It has the characteristics that the firm keeps tight control over knowledge development in-house as well as control over the product sold. This ideal model works best with a specific model of product, that is, with selling a standardized, closed, mass market good at a given time. It also means firms try to develop knowledge in such a way as to retain control. For software, such control means selling compiled, closed products which cannot easily be modified by the user. The definition of that mass product may change over time, as the packaged product boundaries may expand into new uses and services. What is important to the firm is to keep control over development and over use, in order to make sure that existing users and new buyers become repeat, or future, buyers of an improved mass product.

This business model works for knowledge intensive - but mass market - products because volume and profitability allow the firm to invest much money into development and related services but to spread those costs over – and to appropriate the economic returns from – many buyers. For software, this implies a certain type of standardized product which can be used by the non-programmer, or else one which is useful for many

programmers, such as a library or programming tool. Exploiting economies of scale may imply a standardization and some degree of simplification of the product, at least as perceived in the user interface.

In this model, the firm should work to retain control over crucial software, like an operating system software, and/or over infrastructures. The reason is that operating systems and infrastructures are the foundation upon which other, complementary programs and services can be provided. Operating systems will compete with each other, where successful ones can be identified in terms of many users. The broader theoretical question, which is particularly prominent among American economists, has been to explore the existence, and relative importance of, network externalities and spill-overs when diffusion and/or conglomeration of knowledge occurs (Katz and Shapiro 1985 and 1986; Liebowitz and Margolis 1994). In other words, why do systems (or regions) which attract more users tend to continue to attract additional users? For operating systems, the successful ones have tended to attract more product and service development, thereby making it more attractive to more (and repeat) users in the future. This is known as system externalities.

In this business model, one firm tries to dominate the others. Such control over knowledge development can be important in an economic sense. The economic returns are highest under the following conditions. If the firm has control over a crucial component and if a widely distributed product is used as the basis of other software products and services and if the firm can also retain control. Under these conditions, then the firm can charge a price for packaged software which allows them to have large profits. Therefore, the firm has incentives to try to control and influence the distribution channels, as well as the technical development of the core operating system and also influence complementary products or software.

However, this situation leads to at least one important implication for strategy, at least for firms following this first business model. All in all, it appears that it is not possible to keep selling the same standardized, knowledge-intensive product over time, if the firm wants to continue to make high returns. A leading firm which wishes to remain a leading firm has incentives to keep moving into new areas of related goods and services, in order to bring them into the standardized product bundle. The reason is that an imitating firm has incentives to follow the leading firm, as quickly as it is clear that large profits might be made, due to, for example, a rapidly expanding user base. To continue to make high economic returns, the firm has to continually envision new, and successful, ways to create and/or capture economic value. For software, this may imply not just defining the market for software as programs per se. Instead, the firm has to pursue other ideas and strategies about how economic value can come from more diffuse, or intangible, aspects related to the software. If the firm wants to retain control, then this is the technical path of integrating as much valuable and complementary aspects as possible into one bundle of goods and services. Moreover, the firm will have strong incentives to control any channels which influence the number, or percentages, of users for their foundation

software. There appears to be a logic of continuing to add things on – to a more or less appropriate but above all, firm proprietary base – in order to make sure that control remains within the firm.ⁱⁱ

3.1: Firm-based control model: Microsoft

For software, Microsoft is the penultimate example of a firm which has succeeded, by following this first ideal business model. Note, however, that other firms have not been as successful using this strategy. As shown in Table 1, Microsoft has been phenomenally profitable and grown quickly.

Table I: Microsoft’s growth over time, comparing 1985, 1989, and 1998

Year	1985	1989	Intervening Years	1998
Number employees		4 000 ^A		27 000 ^A
Net revenue (USD)	\$163 million ^B	\$804 million ^A		\$14.48 billion ^A
Average annual growth in net revenue, high/low			56% / 24% ^A	
Average annual growth in net income, high/low			66% / 20% ^A	
Size relative to other companies	Seventy-eight largest data processing company ^B			*Largest ^C * Greater net profits than next 49 software companies together ^C
Share of the operating system market				* More than 90% of operating system market (eg systems sold that year) ^D

(A = www.microsoft.com/presspass/fastfacts. Viewed on March 25, 1999;

B = Steinmueller (1996: 34);

C = *Economist* (1998f) and *Business Week* (1999a);

D = Court testimony in the American anti-trust suit (Opening statements, paragraph 21, lines 21-23.

Reproduced at www.microsoft.com/presspass/trial/transcripts/Oct98/10-19-am.htm). Viewed on April 7, 1999.)

Table 1 thus demonstrates that over time, Microsoft has succeeded in terms of installed software and in terms of economic profitability and growth. This was the situation before the American anti-trust suit decision in late 1999 and 2000, which will fundamentally change how Microsoft operates in the future. Microsoft’s position at the end of the 1990s should be contrasted with the previous decades. In the late 1970s and early 1980s, IBM was the dominate computer firm (Fischer et al 1983), and Microsoft was seen as an alternative to the ubiquitous IBM. Obviously, something happened in the intervening years.

The most important event for Microsoft's software probably came in 1981, the year the firm was incorporated, because IBM introduced its personal computer (PC). After that, both the American market and the installed base of IBM PCs and clones grew rapidly, which meant that this software market would also grow rapidly. "The enormous size of the personal computer market created unprecedented scale and profit opportunities" (Steinmueller 1996, 34). Microsoft got a crucial market leader opportunity in PC software, when IBM endorsed its PC-DOS operating system as the standard, then later Windows. Microsoft was quick to act upon this, becoming the dominant supplier of PC operating systems. Microsoft's operating systems has in turn often been sold with a package of other applications and also pre-installed on hardware. Over time, Microsoft successfully created a mass market bundle of programs around their core operating system, which has been attractive for large numbers of users.

Strategies for, and competition over, that bundle has changed over time. For example, Microsoft recognized the value of images on the PC start-up screen and desktop. They have been accused of going to extreme measures to keep their programs pre-installed and to keep their trademark image on the start-up screen, through threats to hardware vendors. The firm has, however, also made mistakes in trying to decide which areas will be of economic value in the future, and sometimes has been an imitator rather than a leader. In late 1995 in response to Netscape's browser, Microsoft woke up the race for software to make the PC entirely ready for, or part of, the world wide web (*Economist* 1996c; see section 4.1 below).ⁱⁱⁱ Microsoft's strategy was to include an internet browser for free as part of their operating system, thereby using their existing distribution channels to diffuse it to increase the number of users.

Note, however, that the challenge was not so much for the market, or sales, of the software per se. Indeed, Microsoft gave away its Internet Explorer for 'free', or as part of the package. The challenge at the time lay in a combination of Netscape's browser with the workstation company Sun's Java language. The idea was that NCs could actively use programs found on the web and based on Java rather than relying on installed software in PCs (*Economist* 1996e). Together, they might have had the potential to replace PCs and PC software with network computers (NCs), by dominating users' access to the web.^{iv} If realized, that vision would decimate Microsoft's core market for standardized, mass market software bundles for the PC. People would no longer need to buy these products nor would the operating system be the foundation software necessary for the whole bundle.

In addition to reaching individual users of PC software (whether those PCs were used privately or in a firm), Microsoft also competed with Netscape and Sun for corporate users of systems. Corporate users needed hardware and software to run larger networks, connect their system to the internet, etc. Sun's workstations were being used in distributed networks as well as for internet servers, but they were increasingly being challenged in

the mid-1990s by high performance PCs running Microsoft Windows NT software (*Economist* 1997e, 1997f). In other words, these corporate product markets had previously been segmented, niche software markets, but now they could increasingly be supplied by more standardized software and hardware. Another side of the Sun-Netscape competition was thus that Microsoft had an opportunity to move into its competitors' corporate market.

It is clear that over time, Microsoft has had strong incentives to keep control over, and encourage more use of, the whole bundle of applications and services around its core operating system. There are many reasons for why a system around one operating system has become so dominant, leading to the American anti-trust suit.^v The Department of Justice experts argued that in order to determine market share, it was appropriate to look at distribution through installed hardware as well as sales of programs. Microsoft has had a combination of alliances and relationships with hardware assemblers, which has led to large numbers of pre-installed software on PC hardware.

If we turn to how Microsoft organizes the search for novelty, Microsoft does retain control in-house over software development. Thus, the firm follows this century's trajectory of large firms investing into in-house R&D, and/or hiring educated employees in order to innovate. Company employees develop new ideas, new products, and who write new code, debug, test, etc., and their incentives to do so come partly through a combination of wages and stock options. Of the 29 000 Microsoft employees in 1999, more than 30% work with research and development, in different groups.^{vi} Throughout the period studied, Microsoft has shown little interest in collaboration and joint R&D with other companies, except for hardware producers. The majority of its external relationships seems to be investments and acquisitions of small and medium sized firms, so that, for example, twenty cases of acquisition or investment were reported for 1998 (*ibid*). Increasingly during 1999, however, there were more joint development efforts with other firms, particularly to try to move a Microsoft operating system into other electronic devices and/or to develop their internet services. Microsoft has an incentive to collaborate there because they do not have a dominant foundation system.

When there is firm-based control over software development, then the firm needs an internal management structure to coordinate. The problem for Microsoft's technical strategy of including more and more into the core mass market product (with some differentiations for different markets) is that the structure of software has also become increasingly complex. The firm has in practice been continually adding new features, more complexity to the core and therefore, it has relied on increasingly powerful hardware to run increasingly complex software. In this case, the source code has become enormously complex. In such a large structure, changing one thing may affect many other things, even though attempts to modularize helps minimize some systemic effects. This implies that Microsoft needs an organizational structure capable of handling a large amount of complexity to

develop its product. Whether it works well or not within this firm is another question. The main point is that increasing complexity when control over development is kept in-house also demands increasing coordination in-house.

That Microsoft faces these types of technical problems has been evident in professional IT literature, which often complains that the current Microsoft operating systems are instable (eg crash), especially when one additional thing is added to the existing complex and large structure. Since the Windows environment connects all the parts, then if one thing breaks, the whole system crashes, rather than just that subroutine. Another indication is that Microsoft's forthcoming operating system was delayed (and renamed) from NT 5 to Windows 2000. The new system is supposed to replace all the various professional, corporate and individual user versions, especially Windows NT, 95 and 98. In that sense, it is to replace product differentiations for different markets with one product.

The problems with this strategy, as identified by the IT community, can be expressed by the following quote. "Which average computer user wants to upgrade to an operating system that is seven times larger and consists of 35 million lines of program code?" (*Ny Teknik* 1999, author translation). Microsoft seems to be placing its bet that most people will not care about technical details in a situation where hardware continues to improve so rapidly. Although increasing complexity in software has in the past been supported by better hardware, it is not clear if this strategy will continue to be viable in the future. Currently, the fastest growing segment in the American market is at the low price and less than frontline hardware. Not only may consumers not purchase the latest hardware just in order to be able to run the latest software, many users will expect to pay much less total for software.

As development, design and testing are the expensive parts of software development, the firm which keeps control over software development – and thereby has to pay for these activities - also has incentives to protect the results in order to make profits. Protection of software intellectual property rights may take many forms, such as secrecy or ownership over certain code and algorithms. Like many companies, Microsoft has traditionally keep tight control over its economic value through secrecy of the source code. Up to April 1999, Microsoft only released already compiled versions. Moreover, Microsoft had also closed its source code to those companies which develop applications software based on the Microsoft operating system, which were instead given access to interfaces. Like most software companies, however, Microsoft does release early, so called Beta releases of programs. *Economist* (1996d) describes them thus: "Add 'beta' to the name of any software package and your product need not work as it should - or at all. It can be full of bugs crashing constantly, sometimes taking the rest of your computer with it. If it does, too bad. Expect no refund. Rather than compensate users, the software maker asks them to spend hours documenting the faults - for free". Such releases can, of course, be

economically important for the firm as they shift some of the costs of testing from the firm to users, who contribute their time and skills.

On the whole, Microsoft has had strong incentives to keep firm-based control over software development and over economic returns. One point about the Microsoft example is that the firm has not only had incentives to keep control, but it has also had the strategies and products to continue to have control, even in situations of dynamic competition. The firm has been very successful in terms of users and economic returns, although increasingly, their technical solutions have come under fire by professional users as have their business strategies in the American anti-trust suit.

4: HYBRID IDEAL MODEL

The second business model relies on firm-based control but also partly relies on network-based development of knowledge. It is thus a hybrid in the sense that it exhibits some characteristics of the firm-based control model as well as some characteristics of the network-based knowledge model. The challenge to the firm is often to decide how much firm-based control they should exercise in terms of knowledge development and in terms of economic returns, compared to allowing a freer and less directed development of knowledge across a distributed network.

This second ideal business model works best with knowledge-intensive products which are developing within a broader knowledge community, but where parts of those products (whether goods or services) can be separated into components which can be sold. For software, this may include a variety of things, probably mostly goods and services which are emerging and attractive to new users as well as to software developers outside the company. Examples would be where the firm tries to take existing knowledge (or infrastructures) and create control by selling software, internet-based services, etc., as a means to try to capture some of the economic benefits. As these are emerging areas with much competition, it is likely that what any one firm will face competition from imitators and parallel innovators and that the firm's strategies will change over time.

Here, it is likely that the firm faces some problems in continuing to adapt over time and that a population of firms shows diversity at any given time, as well as over time, in strategies to try to make profits. Adaptation is a challenge here because the firm has to continue reconsidering which 'bits' of public knowledge for which it can make a marginal improvement, and thereby appropriate the economic returns. Different firms may have very different strategies, depending on their interpretation of the innovation opportunities (Fransman 1999). It is particularly important for economic returns to have a good and/or service which is allied with an attractive operating system – applications – services bundle. Otherwise, the product may work well technically

but not be able to attract the needed numbers of potential buyers as well as developers in order to create the momentum of additional development.

One reason that this business model works for knowledge-intensive products is that the hybrid firm does not have to bear all, and perhaps not even most, the costs of developing knowledge which is embodied in the product. It is useful to compare the first and second business models presented here. Although the firm-based model also relies on publicly known, publicly developed and/or publicly funded sources of knowledge, the hybrid firm relies on this type of knowledge to a greater extent. Conversely, the firm in the second model has less control over knowledge (software) development, although the firm may try to gain control, thereby moving towards a firm-based control model. The firm following this second business model would have less investment in knowledge (software) development, but over time, they should have fewer possibilities to charge a high price for a unique, or at least dominate, product. Thus, in the hybrid model, the firm can appropriate a smaller proportion of the total economic returns to the knowledge. The sums may be quite sufficient for the firm to grow and be profitable.

Because this second model relies on development of public knowledge, other developers of software are also involved in, and important for, the current and future development of the software. This implies that for the firm, protection of rights through secrecy is more difficult, and at times impossible, although the firm may have intellectual property rights on parts of the overall relevant knowledge. Building up a brand name quickly in a new area should become increasingly crucial for success. The most important strategy may be speed, or time to market, rather than dominate product. Definitions of success and failure will have to be examined both in terms of users as well as in terms of economic returns.

4.1: Hybrid ideal model: Netscape

This second ideal business model, the hybrid, is exemplified through the changing fortunes of Netscape. The firm Netscape helped pioneer mass user access to the world wide web, through its browser. Although initially 'competing' for users with a university-based alternative, the firm Netscape then started competing head on with Microsoft when the latter realized the rapidly expanding mass market possibilities of internet use. The complex relationships among different parts of software and different types of users indicates competition can be difficult to define here, ranging from a market to sell browsers to the intangible asset of controlling a browser which funnels PC users into the WWW. Internet and the World Wide Web are, however, based on open standards and protocols, the TCP/IP protocol, and it is therefore not amenable to be under firm-based control.

Before browsers, the internet seemed a chaotic and undisciplined place, where it was difficult to retrieve or send information. It was mainly a place for computer hackers, with the time, skills, and interest to use it.

What browsers did was to create an interface that appealed to a mass audience. More and more users were drawn to the development of the internet in terms of a user-friendly, multimedia world wide web, and the doubling of the number of internet users every year has in turn spurred further development of the web as well as pages, services, etc. Thus, browsers were a first crucial step in changing the internet from a specialized virtual arena to something available, and attractive, to a large number of less expert users.

Netscape was started in April 1994 by Marc Andressen and Jim Clark in order to commercialize the Navigator browser, which is a graphical interface which enables, and also simplified, navigation on the internet (*Economist* 1996b, 1996a, 1996c). This Californian company had its initial public offering in August 1995, generating much stock market interest. By May 1996, their stockmarket value was \$6 billion but by August 1996, that value had already halved to \$ 3 billion (*Economist* 1996c).^{vii} Then in January 1998, Netscape issued warnings about losses and stock value dropped 21% (*Economist* 1998a). Finally, Netscape was acquired by America on Line (AOL), which was at the time the largest internet access provider. AOL put its bid in late November 1998 for \$4.2 billion (*Economist* 1998i), with stockholder approval coming in March 1999 at a somewhat higher price.

The changing fortune of Netscape has many explanations, which have been under dispute in academia as well as in the major US anti-trust suit. The purpose here is not to explain why Netscape had changing fortunes but instead to portray Netscape as an example of a hybrid business model. Moreover, it will exemplify that the history of Netscape exhibits different reliance on public software development over time as well as changes in firm strategies to create, and appropriate, some economic returns.

Netscape's first strategy was to achieve recognition of its name and product. It did so successfully by giving away its browser, Navigator, for free. Initially, their direct competitor in terms of access to users was Mosaic, a free software developed at a university. In fact, while an undergraduate at University of Illinois in Champaign, "Andressen created the NCSA Mosaic browser protomodel for the Internet with a team of students and staff at the university's National Center for Supercomputing Applications (NCSA)...it gained 2 million users in just over one year".^{viii} Andressen was joined by other members of the university team, thereby bringing their special competences direct from a university environment to the firm. The software which was competing for users was Mosaic. It was freeware and therefore could be distributed to anyone without cost or obligation, and Netscape started out giving away freeware, too.

Netscape initial strategy was, however, more than name recognition. The firm gave away its browser Navigator

with the hope that its popularity would help it to sell lots of server software, which can make the most of its browser's features. The give-away strategy worked better than anyone dared hope (within

months it had around 85% of the market), but sales of server software lagged. So Netscape changed tack: it now charges \$ 50 for its browser, and browser sales make up 60% of its revenues. *Economist* (1996b)

Although profitable in June 1996, over a longer period, charging individuals for the browser was difficult to enforce. Anyone could download, "try" it out, and keep using it without penalties, but it was up to individual conscious to pay a minimal fee, at a time when the internet provided no (or little) opportunities for financial transactions. So, although downloading on the net was beneficial in terms of diffusion of the software, few bothered to pay (*PC+* 1999). This change in strategy was, in short, rather difficult to enforce. This was a change in strategy compared to the initial one of trying to leverage its name recognition for other, corporate users, who would presumably pay more. This next strategy was closer to treating its program like a standard product, but rather than selling the product Navigator, they attempted to sell licenses. At this point, it was important to continue trying to build coalitions as an alternative to Microsoft, yet like all companies, they wanted to do so in a way which allowed them to still keep some control over development and returns (Cusumano and Yoffie 1998).

Competing changed, however, over time, and they no longer had a unique product. By August 1996, Microsoft had its own browser, Internet Explorer, which was considered more or less technically on par with Netscape. Explorer was also free with, or part of, a Windows package. At that time, Microsoft had less than 5% of the browser market, but their launch was the main reason that Netscape stock halved in value in the fall of 1996. Soon thereafter, Microsoft had a large installed base of this software, due to pre-installation in sold hardware and to compatibility with their existing PC software bundle. The firm Netscape continued to hope to make money through the adjacent markets of software to corporate users, but found that competition was fierce there too. By early 1998, their share of browsers sales had fallen to 13% of its revenues, and they were still having trouble selling in the corporate network market (*Economist* 1998a). The company was profitable some years, but it was hard to sustain. In 1998, Netscape had 2 000 employees.

As Microsoft started to win the battle for installed browser programs, other firms also recognize that browsers were important for providing mass market input into the internet. From then, Netscape's strategy for appropriating economic benefits became related to value in providing a service. Services became more important when the internet exploded into a mass market. This helps explain why Netscape was acquired by the largest American internet service provider, America on Line (AOL). AOL was not a direct competitor to Netscape in terms of software, because AOL instead provides a mass market service. Its membership (or subscription) rose from somewhat less than 3 million in 1995 to over 12 to 15 million in 1998 to 17 million in early 1999.^{ix} At the same time, however, AOL's strategy has also been based on expansion towards corporate users, especially system administrators for networks, etc. It is telling that in connection with the Netscape acquisition, AOL also

announced collaboration with Sun.^x Netscape, AOL and Sun are the three key government witnesses in the antitrust case against Microsoft.

One of the reasons that AOL has moved into adjacent areas appears to be that provision of internet access by itself has to be complemented with other ways of appropriating economic benefits from a mass service. One reason is that some new service firms provide free access to internet, for which AOL had been charged. These new firms' strategy is to try to make money through paid advertising on portals or websites, although this strategy has rapidly been questioned. For advertisement, name recognition is close to everything. In June 1998, Netscape was claiming that its "Internet site is now the most heavily trafficked site in the world, serving close to 5 million users and receiving more than 120 million hits each day".^{xi} From AOL's perspective, acquiring Netscape's brand-recognition, and its Internet site, was valuable. In terms of making profits, the acquisition pulls together two of the three most visited sites on the internet (*Economist* 1998h). This thus leads to a high combined value in terms of exposure to users, which could be valuable to advertisers. In short, the idea with the acquisition was to appropriate economic benefits more through advertising and service innovation rather than in technical improvements through software development.

In terms of software development in the hybrid model, Netscape thus initially relied on knowledge subsidized through the public research system, whether directly in terms of code or indirectly in terms of individuals' knowledge and skills. The firm then took over control of software development, and users considered they had a technically good product. However, they faced the same challenges, and potential benefits, in asserting in-house control over software development as discussed under the firm-based control model. Netscape was sometimes more successful at appropriating economic returns and sometimes less so. Note that the firm could never dominate the market for browsers in the same way exemplified above, with Microsoft using their foundation operating system to dominate PC software.

In terms of ways of organizing software development, Netscape did release firm-based control and move towards a network-based model of software development, towards the end of their period as an independent firm. About 11 months before AOL's bid, Netscape did something which commercial companies had not previously done. In January 1998, Netscape announced it would release the latest version of its browser program, Communicator, as open source, and called it Mozilla. Open source software has specific codes of conduct (see Linux section). Netscape in fact separated software development for Mozilla from the commercial part of the company.

As open-source is free and readily distributed without restrictions, this meant that technical development of that branch of software could go on independently of the fortune of the company. This also implies that development is not directly under the firm's control. Some thought the real question about Netscape's

unprecedented action in releasing source code was "whether Netscape can make money on it" (Moody, 1998: 44). A different interpretation was that releasing source code implied that the program might continue being developed, independent of the fortunes of the company. Netscape's actions to release Mozilla may have been a sign of foresight of a coming trend towards open source software and/or an action of a company in trouble, but where in-house programmers hoped for continued developments

This example indicates a change from the previously dominate model of firm-based control and secrecy over software development. This runs parallel to what appears to be a broader change in about 1998 to allow some coexistence of commercial software with free, or open source, software development. Many commercial companies had previously not distributed their source code in a non-compiled form nor had they been willing to include other open-source software as an option as installed programs in hardware or as compatible with other models of software. Secrecy was seen as an important way to protect ownership over the knowledge involved. In this sense, Microsoft was by no means alone in choosing the first business model, but they have been uniquely successful at it. However, firms' scepticism about mixing public domain and commercial software began to change in 1998, with various large companies like IBM, Oracle, Sybase, etc., announcing that they would include various open-source alternatives (Moody 1998; *Software magazine* 1998; *Electronic Engineering News* 1998; see further under Linux section). One reason appears to be that all the internet-based software and protocols are open, which implies that no one firm can dominate the source code. The strategies of the first business model are not possible to carry out under those conditions.

To summarize this example, the firm has created a strong brand name, were valued high on the stock market, and it is still currently one of the most popular websites. Yet, the firm has had trouble creating enough economic value to capture profits in-house, hence leading to trouble to continue financing software developments in the long run. It seems to have faced a continual dilemma, or tug-of-war. The firm was started with the idea of leveraging the value of intangibles in software in order to sell other software as commodities. When that did not happen as much as hoped, the firm changed strategies and tried a compromise between software as a free good and software as a valuable product. In other words, they tried to make money on the mass market, by selling (licensing) use of the shareware. Moreover, their apparent value as defined on the stock market dropped drastically when imitators moved in, especially one with a strong distribution channel and name-recognition. Finally, the firm was finally bought by a mass market internet service provider, which had the idea that profit will come through advertising and portals, eg related to internet service provision.

5: NETWORK-BASED IDEAL MODEL

The third ideal business model is the network-based development of knowledge, with node control, which is not linked to firm-based control of economic returns. In some senses, it is more an ideal model of how knowledge development occurs than a business model per se. Nevertheless, it is also a business model in that it helps identify how, where, and why economically valuable knowledge is developed. The development of economically valuable knowledge may in give incentives to some individuals, firms and/or organizations to develop a new service or good related to this publicly available knowledge in order to appropriate economic benefits. In that case, the network-based development of knowledge may change over time, where some remain committed to the network-based model while others pursue a hybrid business model.

On the one hand, networks of relationships, and related concepts like systems of innovation, have been very influential in recent years in describing the increasing importance of relationships for innovation (Edquist and McKelvey (eds) (forthcoming 2000)). Most of the innovation and business economic research focuses, however, on relationships between firms, or else between firms and special organizations involved in the production and distribution of knowledge. Here, the term network does not refer to relationships among firms. The term network is instead used to describe how knowledge is created, specifically through relationships between individuals involved in the production and distribution of knowledge. In the example, the network involves computer programs writing, testing, patching software, and they can easily be geographically and organisationally distributed. McKelvey (forthcoming 2000) defines and analyzes whether or not the Linux example of the network-based business model can be seen as a new form of 'internet entrepreneurship'.

In the case of software development, control does not so much center in one firm as it does in the other two business models proposed here. Still, despite popular discussions of democratic, evolving networks developing software, the network-based model does require some mechanisms for control over software development. At least for the core source code upon which all else may be based, there needs to be some sort of decision-making structure to decide which improvements will, or will not, be incorporated, the main characteristics of the architecture, etc. The main mechanism proposed for this model is control through a node, combined with flexibility. In addition to control over crucial issues, decisions about other aspects of the overall software bundle may be made less centrally, thereby allowing flexibility of all the complementary add-on programs. Both decision-making processes occur in parallel, depending on the relative importance, and centrality, of the decisions about knowledge.

This way of organizing the search for novelty as well as the lack of express attempts to appropriate economic returns seems to allow for the resulting knowledge-intensive products to differ from those found in the first and second business models. In this third model, there are no strong incentives to standardize a production in order to spread development costs over a large number of users as was seen in the first model of firm-based

control. In the network –based model, the knowledge (software) developed may be much more flexible and adaptable to different uses. It may even be unique or only relevant to a few users (or user-developers). As with each type of resulting products, this leads to advantages as well as disadvantages. Some advantages here are that the knowledge (software;code) may be used by many different type of users, for different types of applications. The software becomes malleable and easy to change, as it is open to user/developers with the competence to program. Some disadvantages are that the quality may vary from excellent to terrible, with more costs moved to the user/developer to test and trial the programs. Only those with the right skills may be able to understand and use it, thereby meaning it is difficult to sell or use as a mass market product. In addition, the software may not be tightly integrated into one bundle, so that while flexibility enables adaptability, it may also lead to confusion and difficulties of getting everything to work together.

Thus, in the network model, if there is a high rate of adoption and of interest in a public domain software, the number of users doing tests and improving the code can expand far beyond the capacities of a single firm. Moreover, the software structure of this operating system is truly modularized into different bits of code rather than all linked in one big complex structure. Users need competence to set up a system, and fix things as they break, but they may also have much flexibility to improve, change, whatever, the open software. As more hybrid models are emerging around this network model, then increasingly, some firms will try to gain control over software development as well as over economic returns.

The example used for this third business model is Linux. Linux has been often mentioned in the popular press in 1998-1999, and it is example of software development driven by public domain software. It indicates that the network model can lead to serious challenges to proprietary software, but it is also one of how public domain software begins to convergence with commercial software and products over time. Here, both individuals, communities of programmers and firms are important for different parts of the process.

5.1: Network-based: Linux

Linux exemplifies the third third business model. This model more clearly shows how software development and appropriation of economic returns can be decoupled from the control of one firm, and instead rely on the actions of many, distributed computer programmers. At the same time, Linux is an example where one set of knowledge (software) gains in popularity and in number of users, this may attract people interested in appropriate benefits according to the hybrid business model. In this example, the hybrid model has become visible through the establishment of new firms, which cannot sell the core operating system but can, and do, sell bundles of appropriate programs ('distributions'), help services, manuals, etc. Moreover, established software

and hardware firms begin allowing open source, or public domain, programs to be installed as an alternative to the dominate commercial ones. In short, they have begun to offer a choice to users.

Unlike the examples of Microsoft and Netscape, Linux is not one firm. It is instead an operating system, now based on open-source software, which has developed through a network.^{xii} The Linux operating system is called the Linux kernel. As with Microsoft's operating system, the Linux kernel is a sort of translator down to the ones and zeros of the computer language as well as a platform for running different applications. For Linux, there are official and test versions of the kernel as well as patches, or improvements, which can be added onto the kernels to make them work better. The test versions are like beta versions release by commercial companies, as discussed above under the Microsoft example.

One individual wrote the initial source code which forms the core of the Linux operating system, but further software development has been structured through a network, or community, of software programmers. The operating system Linux had an initial beginning in 1991, when Linus Torvalds, then a University of Helsinki student, decided to write to develop a Unix-based operating system which could run on a PC. He felt the existing operating systems for PCs were too expensive and/or did not work well enough, particular as his technical standards were based on his use of Unix from university machines. Unix was an operating system originally developed at Bell Labs, and it was widely available (Steinmueller 1996) as well as "open" for tinkering by programmers. In terms of future adaptation, it is thus important to note that Linux is based on a larger family of Unix software, already familiar to many computer programmers. As to why one person would take on this massive programming task, Torvalds said, "I was the traditional geekish person who easily sat in front of a computer for eight hours a day" (*Computer Reseller News* 1998a). In another interview, Torvalds was written up as saying he'd "been programming since he was 10 (what else are you going to do in Finland if you hate ice hockey?)" (*Time* 1998).^{xiii}

Later on, however, the Linux kernel and its adaptability to other applications and hardware has developed into much more. The reasons it has been able to develop so far, so quickly, exemplify a successful example of the network model, or 'internet entrepreneurship'. Torvalds was willing to release his source code via the internet, at a time when use of the internet was expanding rapidly. It is worth asking why Torvalds put his source code available to others. "Why did you decide to seek feedback via the Internet? Torvalds: It was fairly unplanned. I'd been on the Internet about a year and a half before I made Linux available. It was just the way things are done. In a university, you get recognition by publishing what you're doing" (*Computer Reseller News* 1998a).^{xiv} Torvalds initially decided to release Linux as GNU-software. This was a concept and licensing term for free software, based on Richard Stallman's ideas (*PC+* 1999). GNU-software was an early attempt to regulate some of the behavior about sharing bits of code so it might continue at a time when more and more commercial

and packaged software drew upon the efforts of this larger community. The concepts and/or licensing terms which apply to non-proprietary software have been widely diffused and discussed on the internet, mainly because development in technology and in concepts, or ideology, have gone hand in hand.

By January 1997, Linux was attracting relatively many expert user/developers who thought it was a good operating system and were willing to develop it further.^{xv} In 1998, the concept of "open source" software, with specific rules of conduct, was proposed to unite the various licenses for (more or less) free software (*PC+* 1999). These rules and certification were also self-organized and developed over the web by computer programmers. The following nine conditions for certification set some boundaries which will influence how and why invention and innovation in open source software will differ from commercial software^{xvi}:

- 1) Free redistribution, without restrictions or fees
- 2) Program must include, and allow distribution, in source code as well as compiled form.
- 3) Modified and derived works have the same rights of distribution,
- 4) Integrity of the author's source code, enabling users to distinguish between base source (eg non-modified) and patch files,
- 5) No discrimination against persons or groups,
- 6) No discrimination against fields of endeavor, eg including commercial users,
- 7) Distribution of license follows with all redistributions,
- 8) License must not be specific to a product, eg dependent on being part of a software distribution (package),
- 9) License must not contaminate other software, eg must not demand that all other software is also open source.

There are thus specific, if somewhat vague, rules about the characteristics of software development and diffusion. Note that they specifically restrict characteristics which were evident in the firm-based control model.

As the example of Linux shows, the implications of the source code being open and available on the internet has meant that the network model is not only possible, but it can also develop rapidly - if and only if the network expands. Technically, it has meant that many others can conceivably, and rapidly, participate in software development and testing of the kernel, applications and device drivers than one company can employ.^{xvii} Anyone can participate, as long as they have the necessary knowledge to understand and develop software. All are allowed to "read" the source code in a flexible structure as well as try to improve the kernel and/or make it run better in new situations. Thus, users can be distributed anywhere globally.

Note that speed and focus of the resulting software can differ here as compared to the firm-based model. At least as long as enough existing or new users are willing to act as unpaid developers, then software development continues in this third model. However, if no one is interested in developing an alternative software based on a network model, then it can just as easily die out, or run into dead-ends. The network-based model, relies on some sort of attraction so that individuals and organizations invest their time and energy into development. Under, in contrast, the firm-based control model, then the firm has the resources, incentives, decision-making to continue developing software, or else stopping it.

There are, however, nodes of control in this third model. In this example, Torvalds personally has continued to oversee a large, collaborative development process involving the operating system. While this organisational structure with one person making the final decisions about software development works up to a point, it also has obvious limitations in dealing with complexity and integrating novelty. What then happens to the decision-making structure developed so far when "the Linux model exploits the ingenuity of typically hundreds of programmers and hundreds of thousands of testers"? (Moody 1998, 42) What seems to happen is that although Torvalds has the final say, inclusion or not in the official release depends on a consensus view, where leading programmers in different fields can have their say (Moody 1998). Torvalds relies on a trusted group of advisors. "Linus began choosing and relying on what early Linux hacker Michael K. Johnson calls 'a few trusted lieutenants, from whom he will take larger patches and trust those patches. The lieuts more or less own relatively large pieces of the kernel'."^{xviii} In other words, the decision-maker with the final say about technical developments in turn relies on a small group with technical skills for some of the crucial decision. In addition, other decisions are decentralized. There are many small programs available to make different parts of the Linux structure compatible with each other. Proposals and trials are more freely available without an elaborate decision-making structure; it is to a larger extent up to the user to decide.

In parallel, however, the network based knowledge development in this example has stimulated firms with a hybrid model. There are firms bent on commercialising Linux related goods and services, and they contribute to software development by attracting additional users as well as participating in software development. This has contributed to a distribution of Linux well beyond the initial group of computer programmers who are willing and capable of improving the kernel. In fact, it is leading to a situation where under certain situations, Linux may also have the possibility to diffuse into a new user group, namely the non-expert mass market.

The current popularity of Linux depends on two groups of factors which can explain its diffusion and attraction for users. One set relates to the fact that the media picture of Linux as only the network of user-developers fits with much of the ideology of hacker culture. This could attract additional user-developers. Related to this is the exposure of the name, with an explosion of articles written in both professional IT press and business oriented magazines.^{xxix} Exposure is important in terms of making people aware that Linux may be an alternative to Microsoft, thus potentially leading them to a choice situation next time around.^{xxx} Another reason relates to the fact that other open-source software already runs much of interactive media on the world wide web as well as transportation of information on the internet. Particularly important open-source software which runs servers, engines and send email are currently Apache, Perl, BIND, and sendmail.^{xxxi} Linux thus fits into a context of commercial use of open-source software, as authorized (and unauthorized) corporate use has

been gaining ground in 1998 and 1999 mostly to run servers and networks. In this, Microsoft itself has played an important role in casting the limelight on Linux because of two leaked memos on respectively, open-source software in general and on Linux in particular. They were leaked in October 1998 and known as "The Halloween Documents".^{xxii} The reason that these memos, and hence Linux, garnered so much attention was that they contained two propositions guaranteed to get most people's attention – 1) that open-source software poses a serious threat to Microsoft and 2) that Microsoft should combat it.^{xxiii} However, there has been much speculation about why the two memos were leaked. "Was it intentional spin control, in the context of the ongoing federal case against Microsoft, or was it guerilla support for the open source software movement within Microsoft?" (*Information Week* 1998a).

The other set of reasons are related to compatibility and cost. Beyond expert users, most users seem to more functionality, compatibility with applications and lower costs. Linux is an example of an open source software which has met some of these requirements. Firstly, Linux has remained compatible with other versions of Unix, because "it was easier than porting 35 000 applications" (*Computer Reseller News* 1998b). In other words, using all those existing applications to run on this operating system decreases the costs of adoption. Moreover, many university students or employees are familiar with Unix (and Sun's version, Solaris), giving them the skills to understand this software structure as well as the potential to write short code to make things work. Secondly, for new groups of users, its attractiveness seems to increase when backed up by firms which guarantee service and help (*Information Week* 1998c; *Software Magazine* 1998; *InfoWorld* 1998c). Basically, what companies like Red Hat, Caldera, and S.u.S.E. do is put together so-called Linux distributions, which consist of the official version of the kernel, a number of applications, a manual, and more recently, technical support.^{xxiv} The reason that they can create economic value is that even though a user can pick up all the same information from the web, there are costs involved in obtaining all the bits and getting them to work together. They are willing to pay that money to someone else.

The distribution firms, which represent the hybrid model, each seem to be specializing towards different market segments, ranging from a more standardized mass market use, professional IT users, and now even, retaining flexibility over standardization. In fact, these commercial companies clustered around Linux are also heavily involved in software development. They will play an important role in influencing future directions of software development because at the moment, packages which work on one company's distribution will not necessarily work on another. There is a sort of competition of standards within Linux, which can be interpreted as attempts to gain firm-based control within the Linux community. This is interesting because the best for further diffusion would be for homogeneity in competition with other (commercial or other) standards.

The network model of development as well as these hybrid distribution companies are increasingly interacting with large companies, to make the software and hardware compatible, as well as new companies making applications. During the summer 1998, all the top sellers of database software except Microsoft said they would release Linux versions; these include Computer Associates, IBM, Informix, Oracle, and Sybase (Moody 1998; *Economist* 1998e). Moreover, Intel announced that its new 64-bit Merced chip would support Linux in addition to Microsoft and HP's Unix system. Oracle formed technology and marketing partnerships with four Linux distributors, namely, Pacific High Tech, Red Hat Software, SuSE and VA Research (*PC Magazine* 1998). That the computer special effects for the blockbuster movie Titanic were generated on a Linux operating system has also been good publicity (*Economist* 1998d). The view that has been emerging is that in addition to low price, Linux is a stable and flexible program for running servers, either alone or in configuration with a Mac or Windows environment (*Economist* 1998e; *Ny Teknik* 1999). In fact, Linux was thought to have tripled its share of the market for server operating systems to 17,2% in 1998, thereby expanding faster than Microsoft NT (*Economist* 1999c). Linux is seen as flexible and reliable.

A related advantage from the perspective of corporate users is cost, both of the software as discussed above but also of the equipment. Linux can run on most hardware (Moody 1998; 42). Thus, in addition to working well technically, Linux enables hardware equipment which was previously seen as inefficient and/or outdated for large and complex modern software. That hardware would now be worth something again, in the sense of being able to run applications, and/or part of a larger network. Price comparisons over new hardware alternatives can also be favorable. In a comparison of cost from December 1998, the head of human genetics lab at Rockefeller University, New York claimed that "a 400-MHz Pentium II computer costs \$ 1 500 and is as fast for our computations as an \$ 20 000 Silicon Graphics workstation with a 175 MHz R10000 processor" (*Network World* 1998). In other words, a Linux based system enables a PC to compete with higher end workstations, again making it a direct competitor to Microsoft's NT as well as to Silicon Graphics, Sun and other providers of high profit margin workstations.

Thus, this particular example of network-based business model has been gaining momentum for a number of related reasons. There are, however, no guarantees of success for this particular operating system bundle nor for other alternatives developed this way.

6. CONCLUSIONS

This article has argued that three concurrent, and competing, business models are visible in the knowledge-intensive sector of software development. The business model of firm-based control is exemplified through Microsoft, the hybrid through Netscape, and the network-based through Linux. These three do not

overlap in a traditional view of product markets, yet over time, they do compete for users. The previous sections detail the advantages, and disadvantages, of each model in relation to appropriating economic returns and in relation to organizing the development of knowledge.

Presenting these three models and examples has led to a comparison of the economic dynamics of software development. It shows that like many other products, the creation of a standardized product for a mass market is important in knowledge-intensive products like software. In software development, the firm-based model has created a mass market through protection. This has been possible through secrecy of source code, which could be kept by only distributing compiled source code and interfaces, and/or through ownership of patents. However, these cases also differ from the picture of standardized industrial products because the definition of what is "standard" and available for the mass market product rapidly changes over time in the example given. In this case, the product increasingly incorporates more and related features and services which were previously seen as separate.

There seems to have initially been a trend of increasing firm-based control over software, but this keeps, however, being challenged by other software architectures which emphasize flexibility, modularization, and the ability to adapt to the different needs of different users. These types of software development generally rely on network-based development, which may then stimulate a hybrid model where the firm tries to control and appropriate some of the economic returns.

All three models have to handle problems related to complexity and coordination. In short, each requires some means of managing software development, at least in terms of some mechanisms and principles to decide what is wrong with the current version, and also what should be incorporated into the next version. The examples give a range from control in-house in a firm to a combined system with control over nodes parallel with decentralized control for other issues. These mechanisms are in turn related to opposing views about whether or not to keep control over the development of knowledge within an organization, in relation to appropriating economic value. Having standardized and closed products demands control in order to protect economic value whereas having flexible individual products allows many to participate and emphasizes the lack of control. The three models exemplified through the cases illustrate the dynamic competition over time, rather than set categories which never change.

Software development is representative of a larger category of knowledge-intensive products and sectors. There are at least three important factors which make software more like information traded on the market than like traditional industrial products, even though there are also some important differences between information and software. For the first, when sold, software is more like the special case of information sold on a market (Nelson 1959; Arrow 1962). Like information, two parties can have the same piece of software, without

depriving the other of use. Thus, if the buyer gets access to the source code or the program, the buyer can duplicate it without paying the seller. Unlike information, however, software can and is increasingly often sold based on the tasks it can perform, not the underlying program code.

Secondly, the major costs of software development lie in designing, writing, and testing software and not in production or copying. This is similar to many R&D intensive sectors, where the cost of discovery, development and testing may be relatively more expensive than production. Pharmaceutical companies, for example, often justify the high price charged for new drugs as a way to recuperate past and future costs of R&D – and not on the high cost of production.^{xxv} This is one reason why competing business models can exist, side by side.

Depending on the model of software sold, however, the costs of production and of distribution may be high, say similar to industrial goods or else they may be very low to zero. One reason that these costs can vary so much is that distribution can occur through a variety of forms. These range from pre-installation on hardware (computer equipment) to buying and licensing software packages to distribution on the internet. If the software is customized and/or only sold to one customer, then production and distribution costs are based on the one buyer – one seller relationship, and are tied into the costs of development. If sold like standardized products (shrink wrap programs), distribution costs may be similar to other standardized goods, even if production costs are still low due to low cost of the medium and of copying. For other models of software like pirate copies or shareware, these costs can be negligible for the initial software author/producer. Given previous investment in the necessary computer equipment, a user needs only a few seconds to copy software. When software is placed on an internet website, then costs of distribution are even transferred from the author/producer to the user.

Third, software is a particular kind of market, where software sold as market products sometimes "compete" with software which is freely available. Different models of software may compete in terms of access and adoption by users. Users are not synonymous with buyers because software users have different possibilities to obtain computer programs ranging from market transaction (buying a legal copy) to downloading freeware or shareware (free or minimal charge, with the latter similar to licensing) to obtaining pirate copies. These are all ways of distributing software and to the extent that the distributed computer programs (or code) are substitutes, they should be viewed as competing forms of software in terms of users. One reason competition for users matters is that attracting additional users may be important to create momentum for further development (Katz and Shapiro 1985 and 1986). Overall adoption is important because having many users for one program tends to attract additional software development in complementary software as well as for future products. This in turn potentially attracts more future users. In addition to competing for future users, an important factor of software success lies in existing users, including already installed programs.

Thus, because of potentially low costs of production and distribution as well as its high knowledge intensity, software which is openly available on the internet tends to develop in a manner which is even more like public knowledge than like a traditional or standardized product. This leads to interesting dynamics when public knowledge competes with, complements, and merges with, private products. In this situation, software may be more similar than previously recognized to other sectors with on-going, two-way, relationships with university research. Mowery and Rosenberg (1998) argue that in different periods, different sectors with these characteristics have been important for American economic growth. Although everyone seems to increasingly recognize the importance of information technology for productivity growth, increasing questions remain to be explored about the more direct, and indirect, relationships between software development, economic exploitation of software, and university research.

These theoretical-based arguments about the economic dynamics of software lead to an interesting question about whether new forms of organising software development based on internet relationships represent an emerging and unique phenomena which has the potential to beat, or at the least seriously threaten, existing high tech firms which invest much resources in R&D. The R&D intensive, or high tech, firms and sectors have been the site of major innovation opportunities in the post World War II period. McKelvey (forthcoming 2000) analyzes the potential for 'internet entrepreneurship' to replace firm-based control, but finds that over time, both firm-based and network-based models seem to change towards the hybrid model of network development of knowledge, with firm-based appropriation of economic returns. Although clearly an alternative to firm-based software development, that analysis shows, however, that this network-based form for organizing creation of novelty is more a complement than a substitute for firm-based innovation.

At least for attractive software, in the sense of attracting larger number of users and thereby also attracting imitators (or substitutes), then knowledge intensity in the product leads us to identify a reason for economic dynamics as knowledge goes from private becoming public. As with other types of knowledge (Nelson 1996), it seems that software can also go from being private (in the sense of proprietary, secret, and/or tacit) knowledge to public (in the sense of accessible to a wider audience) knowledge. Examples in software could be from the source code becoming public, or the release of functionality and lay-out of a program. Imitators may be close after. The shift of economically valuable knowledge from private to public is thus important because it decreases the value of the current product bundle to the leading firm. When imitators come in, they will most likely lead to price competition, and hence a reduction of monopoly (or innovation) profits. Firms following the firm-based control therefore have incentives to keep control over innovation as well as over distribution and use of their software alternative. The arguments presented here show, however, that imitation is not the only form of competition for the software, or other knowledge-intensive, firms trying to follow firm-based control model. Serious

competition may come from the network-based model, and/or hybrids spinning off from either the firm-based model or the network-based model. While the network-based model may lead to software, or other knowledge structures, which are difficult to obtain ownership over, they do have advantages in terms of flexibility, trying out options, and so forth – as long as enough user / developers participate and/or hybrid firms begin affecting development.

REFERENCES

- Abramovitz, M. (1989). *Thinking about growth*. New York: Cambridge University Press.
- Arrow, K. (1962). "Economic Welfare and the Allocation of Resources for Invention" in *The Rate and Direction of Inventive Activities: Economic and Social Factors*. A Conference of the Universities. National Bureau. Princeton: Princeton University Press.
- Arthur, B. (1994). *Increasing Returns and Path Dependence in the Economy*. Ann Arbor: The University of Michigan Press.
- Buderer, R. (1999). "Software's Ultimate Sandbox". *Technology Review: MIT's Magazine of Innovation*. January-February. pp. 44-51.
- Business Week* (1999a) "The Best Performers: The Business Week Fifty, Top Companies of the S&P 500". March 29. European Edition. pp. 42-100.
- Computer Reseller News* (1998a). "Linux Movement's Founder", Issue 820, December 7.
- Computer Reseller News* (1998b). "Linus Torvalds", Issue 815, November 9.
- Computerworld* (1998). "Linux: Good (and bad news: No one owns it)", Vol. 32, Issue 38, p. 34.
- Cusumano, Michael A. and David B. Yoffie (1998). *Competing on Internet Time: Lessons from Netscape and Its Battle with Microsoft*. New York: The Free Press.
- David, P. (1985). "Clio and the Economics of QWERTY", *American Economic Review*, 75, pp. 332-337.
- Dosi, G.; C. Freeman, R. Nelson, G. Silverberg and L. Soete (eds.) (1988). *Technical change and economic theory*, London: Pinter Publishers.
- Economist* (1996a). "Why first may not last", 16 March 1996.
- Economist* (1996b). "Software. The rabbit and the pick-up truck", 29 June 1996.
- Economist* (1996c). "Microsoft vs. Netscape: Freer than free", 17 August 1996.
- Economist* (1996d). "Beware Geeks bearing gifts", 31 August 1996.
- Economist* (1996e). "Sun Microsystems. Collateral damage", 7 September 1996.
- Economist* (1997a). "Why Netscape isn't dead", 5 July 1997.
- Economist* (1997b). "America (and everyone) Online", 13 September 1997.
- Economist* (1997c). "Microsoft's browser: A bundle of trouble", 25 October 1997.
- Economist* (1998a). "Netscape. Growing up, slowing down", 10 January 1998.
- Economist* (1998b). "Microsoft's contradiction", 31 January 1998.
- Economist* (1998c). "Play nicely, or not at all", 23 May 1998.
- Economist* (1998d). "Software. Revenge of the hacker", 11 July 1998.
- Economist* (1998e). "Free software. Red Hat trick", 3 October 1998.
- Economist* (1998f). "Microsoft: Trial run", 10 October 1998.
- Economist* (1998g). "@ risk", 14 November 1998.
- Economist* (1998h). "War of the worlds: rivals' merger", 28 November 1998.
- Economist* (1998i). "Internet riders", 28 November 1998.
- Economist* (1999a). "When the bubble bursts", 30 January 1999.
- Economist* (1999b). "Economics Focus: Big Friendly Giant", 30 January 1999.
- Economist* (1999c). "Computer programming. Hackers rule", 20 February 1999.
- Economist* (1999c). "Microsoft and the Future", 13 November 1999.
- Edquist, C. and M. McKelvey (eds) (forthcoming 2000). *Systems of Innovation: Economic Growth and Competitiveness*. A Two Volume Reference Collection. Edward Elgar.
- Electronic Engineering Times* (1998). "The coming of Linux: It's just a matter of time", 11/02/98, Issue 1033, p. 57.
- Fisher, F.; J. McKie, and R. Mancke (1983). *IBM and the U.S. data processing industry: an economic history*. New York: Praeger.
- Fransman, M. (1999). *Visions of Innovation: Firm and Japan*. Oxford: Oxford University Press.
- Freeman, C. (1994). 'Critical Survey: The economics of technical change'. *Cambridge Journal of Economics* 18, pp. 463-514.
- InfoWorld* (1998a), "Open Source Guru", June 8, Vol. 20, Issue 23, p. 99.

- InfoWorld* (1998b), "Linux continues to pick up steam", Vol 20, Issue 48, November 30, p 34ff.
- InfoWorld* (1998c), "Linux moves up: Database leaders lend credibility", July 27, Vol. 20, Issue 30, p. 1.
- Information Week* (1998a). "Microsoft's Leaks: Trick or Treat?", Gallagher, Sean, 11/16/98, Issue 709.
- Information Week* (1998b). "Linux Takes Off", Garvey, Martin, 11/16/98, Issue 709.
- Information Week* (1998c). "Three Big Names Back Linux", 07/27/98, Issue 693.
- Katz, M. and C. Shapiro (1985), "Network Externalities, Competition, and Compatibility", *American Economic Review*. Vol 75, No. 3, pp. 424-440.
- Katz, M. and C. Shapiro (1986), "Technology Adaptation in the Presence of Network Externalities" *Journal of Political Economy*, Vol 94, No. 4, pp. 822-841.
- Krugman, P. (ed.) (1987). *Strategic Trade Policy and the New International Economics*. Cambridge, MA.: MIT Press.
- Liebowitz, S.J. and S. Margolis (1994). "Network Externality: An Uncommon Tragedy", *Journal of Economic Perspectives*, Vol. 8, No. 2, Spring 1994, pp. 113-150.
- McKelvey, M. (1996). *Evolutionary Innovations: The Business of Biotechnology*. Oxford: Oxford University Press. Reprint in paperback year 2000.
- McKelvey, M. (forthcoming 2000). "Internet Entrepreneurs: Why Linux might beat Microsoft" in Coombs, R. and Walsh V. (eds). Title of forthcoming book. Paper presented at the ASEAT (Advances in the Social and Economic Analysis of Technology) Fifth International Conference. Manchester, UK, 14-16 September.
- Malone, T. and R. Laubacher (1998). "The dawn of the e-lance economy", *Harvard Business Review*, Sept/Oct 1998, vol 76, issue 5, pp. 144-152.
- Metcalf, S. (1997). *Evolutionary Economics and Creative Destruction*. London. Routledge.
- Moody, G.(1998). "The Wild Bunch", *New Scientist*, 12/12/98, Vol. 160, Issue 2164, pp 42-46.
- Mowery, D. and N. Rosenberg (1998). *Paths of Innovation: Technological Change in the 20th-Century America*. Cambridge: Cambridge University Press.
- Nelson, R. (1959). "The Simple Economics of Basic Scientific Research", *The Journal of Political Economy*, Vol. 67, pp. 297-306.
- Nelson, R. (1996). *The Sources of Economic Growth*. Cambridge, MA.: Harvard University Press.
- Network World* (1998). "Linux makes headway", Burns, Christine, Vol 15, Issue 50, December 14.
- Time* (1998). "The mighty Finn", Janice Maloney, Vol. 152, Issue 17, October 26, 1998, p. 70ff.
- Ny teknik* (1999). "Microsoft på väg att tappa greppet?" (Microsoft losing its hold?). Vecka 4, 1999, Del IT/3.
- Ny Teknik* (1996). "Internet styr PC-utvecklingen" (Internet steers PC development) Jan Melin, Vecka 45, 1996.
- PC+* (1999). "Så fungerar OpenSource" in Linux +, Bilaga till PC+, (Borg, Magnus) Nr 3, Mars.
- PC Week* (1998). "I come not to praise Linux...", Surkan, Michael Vol. 15, No. 47, November 23, p. 96.
- PC Magazine* (1998), "Linux's shot of adrenaline", Rupley, Sebastian; Vol. 17, No. 2, p. 30.
- Pisano, G. (1997). *The Development Factory: Unlocking the Potential of Process Innovation*. Boston, MA.: Harvard Business School Press.
- Romer, P. M. 1990. Endogenous technological change. *Journal of Political Economy* 98:71 - 102.
- Schumpeter, J. (1979/1943). *Capitalism, Socialism and Democracy*. London: George Allen & Unwin Ltd.
- Software Magazine* (1998). "In Linux we...trust?", Harrison, Ann; Sept, Vol. 18, Issue 12.
- Steinmueller, E. (1996). "The U.S. Software Industry: An Analysis and Interpretive History" in Mowery, David (ed) *The International Software Industry: A Comparative Study of Industry Evolution and Structure*. Oxford: Oxford University Press.
- Svenska Dagbladet (SvD)* 1999). "Yahoos vinst överträffade förväntningar" (Yahoo's profits above expectations), Näringslivsdel, p. 29.
- Wolfe, A. (1997). "Trouble in Linuxland", *Electronic Engineering Times*, 01/06/97, Issue 935.

WEBSITES

- America Online, Inc. www.aol.com. Including general information from /corp as well as the "Form 10-K" for the fiscal year ending June 30, 1998. Filed with the Securities and Exchange Commission, Washington DC. Downloaded from www.aol.com/corp/inv/reports.
- Boeing. www.boeing.com. Under financial information, annual report and Form 10-K.
- A Linux commercial electronic clearinghouse. www.linuxmall.com
- Linux international organisation. www.li.org
- Microsoft. www.microsoft.com. Including various pages such as /msft/invest and presspass/fastfacts as well as witness testimony such as "Schmalensee's written testimony," "Microsoft faces competition from many directions", "Windows faces intense competition" and "VII. Competitors Reactions to the Success of Windows".
- Netscape. www.netscape.com. In particular, /company/about/background.html. This site seems to have been last updated about June 1998, where some months later AOL put in a bid.
- Open-source organisation. www.opensource.org. Including information about the conditions for open-source certification of licensing, information about existing open-source software, and antedocted versions of the Microsoft Halloween documents at /halloween2.html.

ⁱ McKelvey (2000 forthcoming) proposes the concept 'Internet Entrepreneurship' to see when, and why, the network-based business model does or does not have economic potential. The proposition examined there is whether Internet Entrepreneurship has the potential to replace and/or be more effective than the traditional research and development (R&D) model of firm-based control. The concept is defined as follows. "Characteristics of internet entrepreneurship are that multiple persons are distributed organisationally and/or geographically but can still interact in real time to create novelty; that user and developer can be the same person; that copying and distributing information may be costless; and finally, that instantaneous worldwide distribution of software and communication over the internet, or world wide web enables the process of knowledge creation to go about in a new way, as compared to traditional patterns of developing knowledge over a community over time."

ⁱⁱ Similar analogies may be found in pharmaceuticals, where, for example, a large company may retain its character as being based on fields of chemistry, but may add on bits of relevant knowledge, such as modern genetic engineering, to an existing knowledge structure about what pharmaceuticals are, and how they work in the body.

ⁱⁱⁱ There are various explanations for why Microsoft was so late in seeing the potential of internet, where one important one was that they were focused on other areas like interactive TV and consumer services online (which have not panned out) (Buderi 1999:49).

^{iv} See *Economist* (1997f) on the Java/Navigator threat. Network computing (NC) was launched 1995/96 and based on the idea that people could use many cheap "boxes" without the power of a PC and without the pre-installed programs (*Ny Teknik* 1996). What is interesting here is how the price/performance ration of PCs outperformed NCs so quickly. In a quote from only 3 years ago, Oracle's boss Larry Ellison gave a view of NCs which was wrong in its prediction of NCs but instead, basically describes PCs in 1999 - with the exception that today, the end user has even more power. Ellison's prediction, "Since they [NCs] are easier to use (turn them on and they are connected) and will cost a few hundred dollars at most the NC unlike the PC will spread beyond the rich and the corporate user" (*Economist* 1996f). Moreover, an interesting point in relation to corporate mind-set is that NCs can be interpreted as a PC-ified version of Sun's own system of distributed workstations with a large central server.

^v These reasons are in addition to the anti-trust suit question of whether or not, and if so, how, Microsoft may have abused a (temporary) monopoly position.

^{vi} www.microsoft.com/fastfacts on March 25, 1999 and www.microsoft.com/msft/invest. Viewed on March 25, 1999. In terms of R&D expenditures, in the 1980s, the larger companies developing software for PCs spent about 10-11% of revenues on R&D (Steinmueller 1996).

^{vii} On the one hand, the history of Netscape is specific as it is closely tied to the expansion of the internet and to competition with Microsoft, all the way into the Department of Justice's antitrust suit. On the other hand, the high valuation and quick dips of internet companies is not unusual (*Economist* 1999a), and hence it is in some sense representative of a larger category of firms. Amazon, the on-line book company which is located in Seattle saw its shares grow 966% in 1998, with its market value surging beyond \$30 billion at one point. In 1999, it is valued more than all of America's bookstores. It has never made a profit. Yahoo! is another company, which like Netscape is a starting point into the internet, a search engine and portal and advertising site. It has similarly had phenomenal growth in stockshares. By 1999 Yahoo! had been public only three years, its stock value was about \$43 billion, with an expected turnover of around \$400 million for 1999 (*SvD* 1999). Thus, it was worth more than Boeing, which in 1998 had net earnings of \$1.1 billion and approximately 240 000 employees (after the merger with McDonnell Douglas Corporation), where the operating revenues in 1997 were \$45.8 billion. Annual report for 1998 and the Form 10-K for 1997, downloaded at www.boeing.com/companyoffices/financial on April 8, 1999.

^{viii} Quote and information from www.netscape.com/company/about/backgrounder on March 28, 1999. However, the timeline of Netscape's history on the webpage at this time only goes to June 1998, after which America on Line (AOL) put a bid and later bought Netscape.

^{ix} Company information found at www.aol.com/corp on March 28, 1999.

^x In connection with the Netscape acquisition, AOL simultaneously announced a partnership with Sun, the maker of high-end workstations and server computers. The idea was that together, they could offer corporate customers the services, hardware and software needed for business solutions, but this also thereby means they planned to compete with IBM, Oracle, Microsoft and the like. (*Economist* (1997b, 1998i) and Form 10-K at www.aol.com/corp, viewed March 26, 1999.) Neither Netscape nor AOL have previously been very successful in this market segment whereas both have been successful in access to mass users of the internet.

^{xi} www.netscape.com/company/about/backgrounder.html. Viewed on March 26, 1999.

^{xii} See the international Linux website at www.li.org for discussions as well as links to relevant media articles. On top of the Linux kernel, there are a number of applications which have been made compatible or else which were written especially for Linux. In addition, there are bits of software interface that allow Linux to run

on a variety of hardware, so called device drivers. Thus, we can talk about the Linux kernel, compatible applications and device drivers.

^{xiii} His own history of events - to a large extent in terms of commented emails - is available at www.li.org/history/index.shtml. See also www.opensource.org/history.html. Viewed April 15, 1999.

^{xiv} See McKelvey (1996) for a discussion of how different combinations of insitutions and incentives create different "selection environments".

^{xv} By then, Stallman began complaining that GNU was not getting enough credit (Wolfe 1997). Torvalds reply was that GNU software had been important, but so had much else.

^{xvi} Available at www.opensource.org. Viewed March 1999.

^{xvii} Discussion of Linux based on a number of webpages and articles, including but not limited to: (*Economist* 1998d, 1998e), (*Electronic Engineering News* 1998), (*Software Magazine* 1998), (Moody 1998), (*Information Week* 1998b), (Buderi 1999).

^{xviii} The so-called Halloween documents which include an extensive history and analysis of Linux, can be found at www.opensource.org/halloween2.html in March 27, 1999. This quote comes from Halloween II - 1.4.

^{xix} Based on article search in Social Science Citation Index. See further McKelvey (forthcoming 2000).

^{xx} Linus Torvalds himself has been very active during this explosion of interest, such as participating in seminars, workshops, interviews with the press, etc. He does so whilealso working for a Silicon Valley company called Transmeta Corporation, a top-secret programming firm started by the Microsoft co-founder Paul Allen (*Time* 1998).

^{xxi} See www.opensource.org. Viewed March 27, 1999.

^{xxii} See at www.opensource.org/halloween.html. Viewed on March 27, 1999.

^{xxiii} The basic strategy, also used against Java, was to "embrace the standards they are based on, extend those standards with proprietary enhancements, and render the base standards themselves irrelevant (and useless with Microsoft products)" (*Information Week* 1998a).

^{xxiv} These and other companies which sell products including services related to Linux can also be found at www.linuxmall.com, as well as their respective websites. Whereas Torvalds is Finnish and the user-developers international, all the better known start-up firms are currently American, and Torvalds himself has moved to Silicon Valley.

^{xxv} Pisano (1997) makes a convincing argument that manufacturing costs are increasingly important to understand the relative success of pharmaceutical companies. This does not, however, negate that R&D costs are still very high, and are increasing as a proportion of sales over time in this sector.