3

Both Sides Now

I'm basically a very lazy person who likes to take credit for things other people actually do.

—*Linus Torvalds*[1]

INSIDE THIS CHAPTER

- Software platforms as information-goods and multisided platforms
- Economic and strategic characteristics of multisided platforms
- Economic aspects of open-source software

Two features of the technology we described in the last chapter shape the economics of software platforms.

Software platforms are a written product of the mind. They are in effect documents, usually written in a high-level computer language. The code involved is malleable. It can be moved, altered, added to, and subtracted from with great ease. It is created almost entirely by people—"almost" because, like composers and writers, most programmers use computers for help.

Software platforms are inherently multisided. They usually serve distinct groups of customers, who benefit from having each other on the same platform. Application Programming Interfaces (APIs) forge the crucial relationship between application developers and end users. The developer can benefit from using APIs when she can sell the resulting software to users who have those APIs on their computing devices.

1. Eric Raymond, *The Cathedral and the Bazaar* (Sebastapol, Calif.: O'Reily Press, 1999).

Although we examine both of these features in this chapter, the multisided nature of software platforms is a main focus of the remainder of the book and the economic aspect of these invisible engines from which we will glean many insights. We conclude this chapter with a discussion of another remarkable aspect of the software business: people working collaboratively over the Internet, often without pay, produce software, including software platforms, that compete with software produced by for-profit firms.

Information Goods

Software is one of many information goods in modern economies. Books, songs, screenplays, patents, and secret formulas are others. Of course, there are differences among these products. Most books are a final product read for enjoyment or knowledge. Musical scores instruct musicians on what to do with their instruments. And software is ultimately a series of instructions that directly or indirectly makes computer hardware work. But these differences pale next to the similarities.

Like all information goods, software has four major economic characteristics. It is a creation of the human brain; it is made of pliable symbols; the consumption of these symbols by one person does not exclude consumption by another; and it is almost costless to reproduce an exact replica of these symbols. A musical score has all these features. A composer can use musical notes to make scores of infinite variety and length. When an orchestra plays a particular score, it does not reduce the value of the score to anyone else. And it is cheap to reproduce the score, as well as any orchestra's rendition of it.

These software features have consequences shared by other information goods. Without intellectual property protection there is no obvious way to make money. (The free software movement discussed at the end of this chapter has found some unusual ways to motivate its participants.) There are extreme scale economies: fixed costs are high, marginal costs quite low. The addition of features is relatively easy and an important source of dynamic competition, incremental innovation, and product differentiation.

Software Characteristics

Produced by an Educated Workforce Software is designed and written by a diverse set of individuals, but typically they are college graduates who often have some training, and perhaps even a degree, in computer science. Software programmers and related professionals who worked in the U.S. software industry had an average of 15.3 years of education as of 2000. That compares with 13.8 years for the workforce on average and 14.7 years for professional service industries (including law, medicine, accounting, and architecture).[2]

There were 1,194 degree programs in computer science in American colleges and universities in 2006.[3] These programs usually offer courses in the design of operating systems. All of the top ten programs as ranked by *U.S. News and World Report* did.[4] There are a number of textbooks on the design of computer operating systems and related topics.

Microsoft is notable for screening people for intelligence and problem-solving skills. A 1995 study reported that Microsoft recruited from the top fifty colleges and universities and hired less than 3 percent of the people it initially interviewed. Microsoft is famous for asking job candidates to solve problems on the spot, such as estimating the number of gas stations in the United States. As of 2004, over 95 percent of the architects, designers, and programmers working on Windows had a college degree and 40 percent had computer science degrees. Ten years later Google has developed a reputation as a company where only brainiacs need apply.[5] It advertises mainly in technical magazines and puts people through numerous interviews that test intellectual skills before hiring one out of the 200 candidates who send in a résumé.

2. http://www.bls.gov/oco/ocos110.htm.

3. http://www.usnews.com/usnews/edu/college/tools/brief/cosearch_advanced_brief.php

4. "America's Best Graduate Schools 2005 Edition," *U.S. News and World Report*, December 31, 2004.

5. Michael A. Cusumano and Richard W. Selby, *Microsoft Secrets* (London: HarperCollins, 1996), pp. 92–93; Internal Microsoft information; John Battelle, *The Search: How Google and Its Rivals Rewrote the Rules of Business and Transformed Our Culture* (New York: Portfolio Press 2005); http://www.cbsnews.com/stories/2004/12/30/60minutes/main664063.shtml.

Made of Malleable Code We have already seen that software programs, including platforms, are a series of instructions usually written in a language such as C++. Managing the creation of millions of lines of code that work together as planned is no mean feat. But one reason these programs have grown so large is that they have been designed to make it easy to add to them. In some respects, doing so is like adding a paragraph to a chapter of a book or another riff when playing jazz. The key difference is that since the ultimate product is digital, users do not experience the addition in the way that adding a chapter makes a book thicker.

Like the contents of a newspaper, the contours of a software program can be changed readily. Just as newspapers have suburban or regional editions that add coverage specific to a particular geographic area, modern software programs may have versions targeted to particular groups, such as Java for small devices. And just as newspapers can add sections to bring in more readers, so software programs can add features or functionality.

Many operating systems added features in the mid-1990s that helped users communicate over networks such as the Internet. Apple's Macintosh included AppleTalk proprietary networking protocols in 1985 and added protocols for communicating over the Internet in 1995.[6]

Easy to Reproduce All information goods are easy to copy. But software programs, including platforms, are especially so because they are necessarily digital. The easiest way to see this is with the open-source operating system Linux. You can download this 5.7 million-line operating system over the Internet from numerous Web sites.[7] With a cable modem connection it takes a couple of minutes. Not surprisingly, piracy is a major problem for software firms that sell their products.

Software platforms are often installed on computer hardware before it is sold. The manufacturer does this itself when it makes both the soft-

6. Jim Carlton, *Apple: The Inside Story of Intrigue, Egomania, and Business Blunders* (New York: HarperCollins, 1997), p. 59; "Apple goes to the core; Apple introduces Power Macintosh 9500 that uses TCP/IP and PCI bus architecture; Product Announcement; Brief Article," *LAN Magazine*, October 1, 1995.

7. http://zdnet.com.com/2100-1104-864256.html.

ware and the hardware. Apple zaps its software right onto its iPods and Apple computers. Or the software manufacturer may license the software to other manufacturers that install it, often from a single master CD. Distribution costs are higher when the software platform is sold directly to users. Thus, Microsoft incurs some costs in reproducing its Windows software on CDs and distributing it through retail channels. The same is true for Linux distributors such as Red Hat. But even in these cases the per-unit costs are relatively low, as with other information goods such as music CDs.

Inexhaustible Once created, there is an inexhaustible supply of software such as platforms. Unlike most goods and services, but like all information goods, consumption by one user does not reduce the amount available for others. Indeed, software platforms are better than inexhaustible because consumption by one user is likely to increase the value of the software to others.

Complementarities and Network Effects System components are generally complements: adding another component to a system or improving an existing component generally increases the value of the other components. Moreover, in many cases, systems have what economists call *indirect network effects* linked to the presence of components.[8] That is, an increase in the number of users of one component often makes that component more valuable as a complement to the other components. As Sony's Internet–based game center for the PlayStation 2 draws more users, for instance, more PlayStation 2 owners will want to buy games supported by the Internet service. As its games become more popular, more consumers will prefer PlayStation 2 consoles to competitors' models. Likewise, an increase in the variety of components (printers for PCs, for example) often increases the value of other components to end users. In recent years

8. Michael Katz and Carl Shapiro, "Systems Competition and Network Effects," *Journal of Economic Perspectives* 8 (Spring 1994): 99. For a general discussion of network effects and their business implications, see Carl Shapiro and Hal Varian, *Information Rules* (Boston, Mass.: Harvard Business School Press, 1998).

economists have tended to apply the multisided platform framework to these situations, as we discuss in the next section.

There also may be direct network effects. These arise when an increase in the number of users of an application or platform directly makes that application or platform more valuable to each user. Its value increases because users can share information and work together more efficiently. When WordStar was the leading PC word-processing program, many people bought it at least in part because they could share documents with friends and co-workers.

Economic Consequences

These technological features shape the economics of software platforms just as they shape the economics of most software products.

Intellectual Property Protection If people could get the source code for any software product, they could reproduce it for next to nothing. The price would fall to almost zero, and the original writer would derive no financial benefit.

Software companies rely on all three major forms of intellectual property protection to guard their investments against this fate.[9] First, they keep the source code secret as much as possible. Before they distribute the software they turn it into the 1s and 0s of machine code. In principle, an able, dedicated, and patient programmer could translate machine code back into a high-level language. But this sort of decompiling is forbidden by almost all commercial software licenses and all but impossible in practice for multi-million-line software platforms. In addition, "trade secret" law protects software developers from the unscrupulous employee or agent who might try to release the source code without authorization.

Software companies also copyright their code. As with a book, you cannot reproduce copies of software programs without violating copyright law. Of course, as with other information goods, piracy is nonetheless rampant, especially in countries with weak intellectual property laws. In India, an estimated 80 to 85 percent of the copies of Macro-

9. Suzanne Scotchmer, *Innovation and Incentives* (Cambridge, Mass.: MIT Press, 2004), chap. 3.

media Flash and Dreamweaver in use are not legal. Even in the United States, almost 30 percent of Macromedia software is pirated.[10]

Finally, software companies get patents on algorithms and other features. The United States had granted about 127,000 software patents through 2004. (It takes about 3.5 years on average for the grant of a patent application.[11]) Apple's iTunes software, for example, allows users to import an unlimited number of audio tracks and encode them into the popular MP3 format, as well as listen to MP3s, audio CDs, or hundreds of Internet radio stations. A patented system for accessing digital media across networks was important for the success of iTunes.

Economies and Diseconomies of Scale Although no one has ever quantified it, it is generally understood that there are diminishing returns to scale in writing software platforms. That is, doubling the size of a platform by adding more features more than doubles the cost (holding the quality of the code constant—one can always write inefficient code). Increases in size create more interdependencies (with N objects, there are N^2 possible pairwise interactions, for instance), thus raising the likelihood of bugs, and thereby raising development, debugging, and test costs more than proportionately.[12] (Object-oriented programming and the use of modules are designed to temper these diseconomies.) But once created, cheap reproduction means that additional copies cost little. The production technology is therefore as shown in Figure 3.1.

These economic features suggest some caution in characterizing the marginal cost of producing software. It is true that once the costs of creating software have been sunk, the marginal costs of reproducing and distributing it are very low. That is an ex post perspective on cost. But it is also true that the likely adoption of a software program is not independent of the costs that are incurred in creating it or revising it.

10. Venkatesh Ganesh, "The continuing story of software piracy," *The Financial Express* (August 8, 2005), http://www.macromedia.com/devnet/logged_in/swozniak_piracy.html.

11. It is difficult to define "software patents," but in 2004, 127,098 patents were granted under the G06F classification, which covers electronic data processing. Data set available from www.uspto.gov.

12. "Linux: Fewer Bugs Than Rivals," *Wired News*, December 14, 2004.

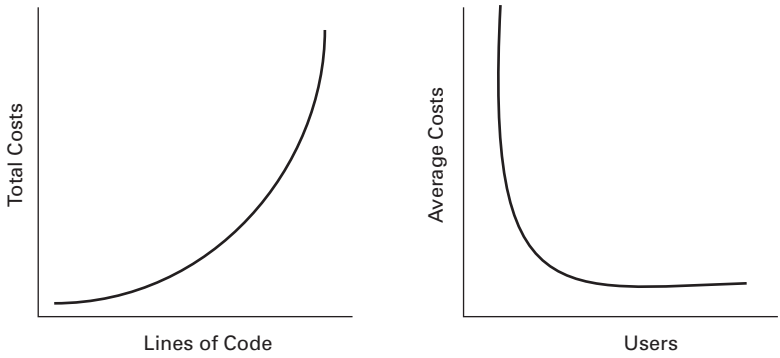


Figure 3.1
Diseconomies of scale.

Software designers add features in part to bring in more consumers; in the case of software platforms, those consumers include both users and developers of applications that run on top of the platforms. Ex ante, the marginal cost of acquiring additional customers by improving the platform, is likely to be much higher than the marginal cost of reproducing and distributing the software.

This distinction is also relevant for other information goods. Movies are a good example. The cost of making a completed movie available to an additional viewer is close to zero. However, the number of viewers a movie garners is partly dependent on the investment in the actors, special effects, and other features that make a movie popular. Movies made with low budgets are often aimed for a narrow audience, while movies with blockbuster potential typically have extravagant budgets. The marginal cost of garnering a viewer, viewed ex ante, is positive.

Pricing and the Recovery of Investments As with all information goods, software poses some challenges related to recovering investments and earning profits. Pricing at ex post marginal cost or anything close to it would lead to bankruptcy. Software pricing thus depends primarily on demand (particularly the responsiveness of demand to changes in prices) rather than on cost and has as its main goal at least recovering fixed and sunk development costs. The pricing of software platforms is considerably more complex because of their multisided nature, as we discuss later.

Why have software prices not declined at the same pace as hardware prices? Basically because software development costs have not declined as rapidly as hardware costs, for two related reasons. The first is that educated labor, which is not becoming cheaper, accounts for most of the cost of producing software. In 2001, U.S. software firms paid about 33 percent of revenues to their employees, while semiconductor companies paid less than half that percentage.[13] The second reason is that software products are becoming more complex: with advances in hardware, software programs typically grow over time through the accretion of features. A typical PC game program in 1994 was 20 megabytes; a typical PC game program in 2004 was 2,200 megabytes.[14]

Bundling Features Most goods are bundles of features, many of which could be provided separately but are not. Cars come with spark plugs and tires even though you could buy your own. Moreover, many goods are improved over time through the addition of features. Few cars come these days without air conditioners and rear window defrosters.[15] Many cereals add fruit and flavors over time, leading to many variations.

The same is true for computer systems. Microprocessors, memory, and other components are typically combined to create a hardware platform such as a Nokia mobile phone handset or an Xbox game console. With time, many peripherals come to be integrated into the hardware platform. Consider the case of the math coprocessor, which facilitates number crunching. Before the release of Intel's 486 chip, Intel's microprocessors did not include a math coprocessor. Customers who wanted one purchased it separately from one of several vendors at substantial cost. Today, one cannot buy an Intel x86 processor without a built-in math coprocessor.

13. 2002 Economic Census, Industry Series Reports, available at http://www.census.gov.

14. "Doom II Game Has 500,000 Pre-Orders," *Newsbytes News Network*, October 13, 1994; http://www.amazon.com/exec/obidos/tg/stores/detail/videogames/B00006C2HA/tech-data/103-8949918-3544622.

15. David Evans and Michael Salinger, "Why do Firms Bundle and Tie? Evidence from Competitive Markets and Implications for Tying Law," *Yale Journal on Regulation* (January 2005): 37–89.

Creating products through feature addition is particularly easy with information goods. That is the beauty of the pliability of music and language. Pop music was mainly distributed as singles on seven-inch records until the early 1960s. The success of the Beatles' *Sergeant Pepper's Lonely Hearts Club Band* album made clear the value of compiling songs and helped make the market for long-playing albums. Bundling multiple songs into albums became standard practice, and the distribution of songs as singles became less common. (We return to this issue in Chapter 11, where we will see that music downloading is helping to promote unbundling.) Newspapers have added various features such as style and living sections over time.

Similar forces apply to software in general and platforms in particular. Where exactly the tasks performed by software are accomplished is a matter of business and design decisions. Many tasks that used to be performed by stand-alone applications have become integrated into other applications (such as spell checkers, which originally were sold separately from word processing programs) or into the software platform itself. Early operating systems, for example, did not include communications functionality.

The malleability of code reinforces several economic forces that encourage the inclusion and accretion of features in products.

Bundling and integration. Combining features in a single product reduces transaction costs for consumers. Rather than having to buy two products, they can buy just one. Moreover, the manufacturer can create additional value by creating connections between the features. An example is making the features of a spreadsheet program available to a word-processing program.

Economies of scope. When there are fixed costs in offering separate products, firms may find it profitable to bundle those products if demand for the separate components is not particularly strong. Several major automobile makers, for example, have decreased the number of different cars people can purchase.[16] They have done this by developing bundles of options that "most people" want, even though some people

16. Ibid.

would not value some of those options separately. For software there are cost savings from combining several features into a single package, as well as savings in distribution and product support. (There may be diseconomies, of course, if making the program larger and thus more complex results in disproportionately large increases in the costs of writing, debugging, testing, and supporting the package.)

Demand aggregation. When there are fixed costs of producing and distributing products but low marginal costs of adding components, it may be possible to lower average costs and reduce variation in what people are willing to pay by combining components that appeal to different groups of customers.[17] Hardware and software typically include many features that most consumers never use. However, by including these features vendors expand the number of consumers who find the product valuable at the offered price. This is why many word processors include equation editors, newspapers have horoscopes, and cable companies include channels that most of us never watch.

Multisided Platforms

That shopping malls and software platforms have much in common is one of the important insights of the economics of multisided markets. The mall is available to stores and shoppers. Once there, the merchants and consumers interact directly on the platform. The merchant rents space. The shoppers often get amenities such as free parking, in addition to getting into the mall for free.

Likewise, the software platform is available to developers and users. The developer licenses its software to the user, who then runs the application on the platform. Both user and developer rely on the services provided by the platform. For many software platforms the user pays to license the platform, while the developers get to use the platform services for free and may even get some subsidized software tools to help them do so.

17. Yannis Bakos and Eric Brynjolfsson, "Bundling and Competition on the Internet," Marketing Science, 1 (Winter 2000): 63–82.

Both platforms help reduce duplication and thereby lower the cost of providing services. Shopping malls provide parking, restrooms, and many other common facilities. Stores benefit because they do not have to provide these facilities on their own. Shoppers benefit because retailers have lower costs. Software platforms make services available through APIs. Developers benefit from these because they avoid having to write some of their own code. Users benefit from a greater variety of and lower prices for applications.

The economics of multisided platforms provides a set of tools for understanding the past, present, and future of software platforms.

The Economics of Multisided Platforms

Multisided platforms cater to two or more distinct groups of customers. Members of at least one customer group need members of the other group for a variety of reasons. Platforms help these customers get together in a variety of ways and thereby create value that the customers could not readily obtain otherwise. The village market is one of the oldest examples of a two-sided platform. It is a place where buyers and sellers can get together and trade. So is eBay. Another old example is the village matchmaker, who helped men and women find marriage partners. Match.com provides a similar service using Internet technology; speed dating is another important innovation. The publisher of this book operates a platform, too. It is in the business of finding authors in search of an audience and audiences in search of content.

Governments run some two-sided platforms. Cash is an example. The government institutions behind the euro help ensure that sellers will take it for payment and buyers will use it for payment. Standards sometimes give rise to two-sided platforms. Fax machines facilitate communication between senders and receivers. Cooperatives of firms also operate two-sided platforms—Visa is the most significant example. For-profit businesses operate two-sided platforms in a wide variety of industries and in many economically significant ones. Highly visible examples include American Express (travelers checks and charge cards), Google (search engine–based portal), and News Corporation (advertising-supported media).

William Baxter presented one of the first formal analyses of a two-sided business in 1983.[18] He was a law professor who was self-taught in economics. He observed that payment cards provided a service only if both cardholders and merchants jointly agreed to use a card for a transaction. He derived some of the fundamental economic consequences of this joint demand. (Baxter went on to become a highly innovative antitrust chief in the United States.)

The notion, however, that diverse industries are based on two-sided platforms and are governed by the same basic economic principles is due to a pathbreaking paper by Jean Tirole and Jean-Charles Rochet that began circulating in 2001.[19] They showed that businesses such as computer operating systems, dating clubs, exchanges, shopping malls, and video game consoles were two-sided.

Economists now recognize that many industries, including the manufacture of software platforms, are guided by economic principles that differ in important ways from those that govern traditional industries. Many of these two-sided or multisided industries are subject to network effects, which were studied extensively by economists during the 1980s and 1990s.[20] Network effects are also central to the economics of multisided platforms, and more recent analysis provides additional insight into their business implications.

Internalizing Externalities Multisided platform businesses tend to arise in markets that have three characteristics:

1. There are two or more distinct groups of customers.
2. There is some benefit from connecting or coordinating members of the distinct groups.

18. William Baxter, "Bank Interchange of Transactional Paper: Legal and Economic Perspectives," *The Journal of Law and Economics* 26 (October 1983): 541–588.

19. Jean Tirole and Jean-Charles Rochet, "Platform Competition in Two-Sided Markets," *Journal of the European Economic Association* 1, no. 4 (2003): 990–1029.

20. Shapiro and Varian, *Information Rules*. See the articles in Symposium on Network Externalities, *Journal of Economic Perspectives*, 8 (Spring 1994): 93–150.

3. An intermediary can make each group better off through coordinating their demands. For example, dating clubs provide a service to men and women, who benefit from meeting each other, and provide an efficient way for men and women to connect.

As a practical matter, multisided platforms tend to arise when a stronger version of condition 2 applies: most platform businesses exhibit indirect network externalities. Consumers, for example, get more value from their credit cards when more merchants take them, and merchants get more value from accepting credit cards when more consumers use them. This has not been lost on the card systems. The current advertising slogans highlight merchant acceptance: "Visa. Everywhere you want to be." "MasterCard: No card is more accepted." American Express, MasterCard, and Visa persuade merchants to pay for taking their cards by emphasizing the millions of consumers that have these cards and want to use them to pay. The sales pitch for the merchants is similar: then the card systems tout the number of cardholders they have who could transact at the merchant if it accepted the card for payment.

Customer groups can sometimes get together without a platform. Men and women have found each other without matchmakers. Buyers and sellers figured out ways to transact before there was money. Merchants can advertise their wares without the media. Successful multisided platforms, however, generally reduce the transactions costs that members of different customer groups would incur in trying to reap the benefits of getting together.

The fact that a platform could exist does not mean that it necessarily will or that it will provide the only method for providing benefits to customers. As we discuss in Chapter 8, Apple has thus far operated its iPod/iTunes platform as a single-sided business. It buys music by paying publishers royalties and distributes this music to customers who want it. Similarly, many consumers have "store cards"—payment cards issued by stores such as Bloomingdale's. In fact, the payment card industry was based entirely on this single-sided model until Diners Club introduced a card in 1950 that put multiple merchants and consumers on the same platform. We explore the decision to become a platform—along with platforms' related decisions regarding which system components to produce and which to rely on the market to supply—in more detail in Chapter 9.

Similarly, many businesses deal with multiple diverse groups without being platforms. There is a sense in which auto companies bring tire manufacturers and consumers together, but they do not do so in a way that makes Toyota, for example, a multisided platform business. In this case there is no direct interaction between the two sides. Toyota substitutes itself for consumers when dealing with tire producers, just as Apple does before sending music to consumers through iTunes. By contrast, two-sided platform businesses provide support for direct interaction between the two sides. Thus, game developers sell directly to PlayStation users, for instance, not through Sony.

Multisided businesses can generate profits for themselves and benefits for their customers if they can figure out ways to increase and then capture indirect network externalities. There are three major ways in which they do this.

First, they serve as matchmakers. Financial exchanges such as NASDAQ and on-line auction sites such as eBay match buyers and sellers. The Yahoo! Personals and 8MinuteDating match men and women.

Second, they build audiences. Advertising-supported media do mainly that: they use content to attract eyeballs and then sell access to those eyeballs to advertisers. Many platforms engage in less overt audience building. Auction houses such as Sotheby's try to build an audience of buyers for the art they sell on consignment. Nightclubs sometimes try to build an audience of women for men, and vice versa. We saw that payment card systems try to build an audience of cardholders for merchants and an audience of merchants for cardholders.

Third, they reduce costs by providing shared facilities for the customers on each side. That's the shopping mall case with which we began. But other platforms also do this to some degree. Buyers and sellers have shared auction institutions and auction sites from the Roman forum to eBay. Readers and advertisers share the pages of *Vogue*. Payment card systems provide a shared network for conducting transactions between merchants and consumers.

Software platforms provide value through matchmaking and building audiences, as well as through reducing duplication. Apple, for example, helped bring commercial artists and developers of design software

together. It did this by including services in the Mac OS that developers could use to develop programs such as Adobe Photoshop for commercial artists. Sony PlayStation has developed an audience of console users that it can make available to game developers. The main economic value of software platforms, however, is in economizing on the amount of code that developers must write to serve the needs of consumers.

The Pricing Balancing Act In single-sided markets, price usually tracks costs and demand for the product pretty closely. Firms figure out what their marginal cost will be and then mark it up—a little if customers are price-sensitive because there is a lot of competition, more if there is little competition. Particularly in stable markets, this is not rocket science. That is why how-to books on starting your own small business can offer reliable advice, such as "charge *X* times cost in sector *Y*." For example, one guide advises that the markup is generally 40 percent of the retail price in hardware stores and that for jewelry it ranges between 400 and 800 percent.[21]

In multisided markets, pricing is more complicated because of indirect network effects between the distinct customer groups. If you charge women the same price as men to enter your singles club, you may not get enough women. If this happens, men will not come, and suddenly you will have an empty club. Many Internet publications discovered that viewers deserted in droves when they attempted to charge them, although some did make the successful transition to paid subscriptions plus advertising.[22]

Multisided platform economics shows that it may make sense for firms to charge very low prices to one or more groups or even to pay them to

21. Stephen C. Harper, *The McGraw-Hill Guide to Starting Your Own Business*, 2nd ed. (New York: McGraw-Hill, 2003), pp. 100–101; Jan Kingaard, *Start Your Own Successful Retail Business* (Irvine, Calif.: Entrepreneur, 2003), pp. 152–153.

22. Michael Liedtke, "Online Subscriptions Herald the End of Web Freedom," Associated Press newswires, March 18, 2002; Thomas E. Weber, "Web Users May Balk at New Fee Services That Deliver Little Value," *Wall Street Journal*, April 8, 2002; Timothy J. Mullaney, "Sites Worth Paying For? The Paid Web Is a Work in Progress, But Some Are Already Getting It Right," *Business Week*, May 14, 2001.

take the product. And that is what multisided businesses do. Magazines, newspapers, and television broadcasters typically earn the preponderance of their revenues from advertisers.[23] Charge card companies such as American Express earn the bulk of their revenue from merchants.

Businesses in multisided markets often subsidize one side of the market to get the other side on board—sometimes explicitly by charging low or negative prices. At other times subsidies are less apparent, such as when the platform makes significant investments in one side and does not charge for it. Table 3.1 shows some examples. We will see that all software platforms make services available to at least one side for free. Most make free services available to developers through the APIs.

Table 3.1
Revenue in Selected Multisided Platforms

Industry	Multisided Platform	Sides	Side That Is "Charged Less"
Real estate	Residential property brokerage	• Buyer • Seller	Buyer
Real estate	Apartment brokerage	• Renter • Owner/landlord	Typically renter
Media	Newspapers and magazines	• Reader • Advertiser	Reader
Media	Network television	• Viewer • Advertiser	Viewer
Media	Portals and Web publications	• Web "surfer" • Advertiser	Web "surfer"
Finance	Proprietary terminals	• Trader/analyst • Content provider	Content provider
Shopping malls	Mall	• Merchant • Shopper	Shopper
Payment system	Travelers' checks	• Check holders • Merchant	Merchant
Payment system	Charge/debit card	• Cardholder • Merchant	Cardholder

23. Lisa George and Joel Waldfogel, "Who Benefits Whom in Daily Newspaper Markets?" NBER Working Paper no. 7944 (October 2000), p. 9.

The economics of pricing for multisided platform businesses has another key implication. In single-sided businesses, the principle that the one who causes the cost should pay the cost is good advice, for businesses as well as for policymakers. For example, a car buyer "causes" the cost of manufacturing the car, and thus pays the full cost. That principle usually does not make any sense in multisided markets, however. Often a product cannot exist unless several different customers participate simultaneously. They all "cause" costs and "cause" benefits. That is true even if it is possible to identify costs that increase as a result of an additional user on one side—for example, the cost of printing another copy of the Yellow Pages. Economists have shown that the best prices—either from the standpoint of the business maximizing profits or from the standpoint of policymakers maximizing social welfare—involve complex relationships between the price sensitivity of each side, interdependencies between these demands, and marginal costs.

Is There Anything New Here? Multisided platforms have a number of features that economists have examined before. Yet traditional learning does not deal with the role of intermediaries in internalizing network externalities. Most businesses have distinct consumer types: workers or retirees, households or corporate entities, men or women. But multisided platforms differ in that they must serve two or more distinct types of consumers to generate demand from any of them. Hair salons may cater to men, women, or both. Heterosexual dating clubs *must* cater to men *and* women. For hair salons the ratio of men to women does not matter much; for heterosexual dating enterprises it is absolutely critical.

Most businesses in single-sided and multisided markets engage in price discrimination (charging different prices that aren't proportional to the corresponding marginal costs) because it is possible to increase revenue by doing so and because, in the case of businesses with extensive scale economies, it may be the only way to cover fixed costs. A dating club may charge men a higher price just because they have more inelastic demand and because it is easy to identify that group of consumers. But

businesses in multisided markets have an additional reason to price discriminate: by charging one group a lower price the business can charge another group a higher price; and unless prices are low enough to attract sufficient numbers of the former group, the business cannot obtain any sales at all.

Like firms in multisided markets, many firms in single-sided markets sell multiple products, and there is extensive economic literature explaining why they do so. The standard explanations for why firms produce multiple products probably apply to many of the platforms discussed here. But firms that make multiple products for several one-sided markets (for example, General Electric makes light bulbs and turbine engines) or several complementary products for the same set of consumers (for example, IBM sells computer hardware and computer services) do not secure profit opportunities from internalizing indirect network effects.

Finally, it is important to ask how the business implications of the recent work on multisided markets differ from those of the older economic literature on network effects. This is not as simple as it might seem, since popular discussions of network effects often missed important subtleties in the academic literature.

Take the case of single-sided markets with direct network effects. Because of those effects it follows that there is an advantage to size, all else equal. But it does not follow that this advantage, if present, is large, and it certainly does not follow that the firm with the biggest market share always wins in the end, let alone that the first entrant always wins. Nonetheless, in the frenetic days of the Internet bubble, lots of businesses were founded on the assumption that network effects were present and important in their markets and that the key to success was to get in fast, price low, and build share at any cost. Proponents of this simplistic view emphasized tipping—you build up critical mass, and then the whole market flocks to you. And they emphasized an extreme sort of lock-in—once you get most of the customers, nobody can enter against you, even with a better product.

As Brian Arthur, an author of several influential papers in network economics, put it, "You want to build up market share, you want to

build up user base. If you do you can lock in that market."[24] This is a nice, simple theory—much simpler than the economic literature from which it claimed to be derived. But it is hard to find many businesses that succeeded by following it. Unfortunately, many dot-com entrepreneurs and investors thought that "build share fast" was the path to great riches. Only a few made it very far down that path before reality closed it off and supposedly locked-in buyers left en masse. It turns out that only rarely are direct network effects strong enough to prevent buyers from switching to a better product, as the massive defections of buyers from once dominant word-processing programs illustrates.

Those who believed that riches could be made quickly and easily by harnessing network effects tended not to distinguish sharply between direct and indirect network effects. In both cases the managerial prescription was to build share rapidly; indirect network effects, like direct network effects, would kick in automatically and both fuel and protect further growth. Work on two-sided markets makes it clear that this is dangerously simplistic in two important respects.

First, even though at least two distinct groups must be involved or there to be indirect network effects, the network enthusiasts assumed both that it is obvious that one should pay and the other should not, and that it is obvious which group should pay.

Second, they generally assumed that the group that did not pay could be ignored in setting business strategy because it would automatically fall into line and generate valuable network effects. In contrast, economic analyses of multisided platforms, along with the industry case studies discussed in the following chapters, show that successful multisided platform businesses must pay careful attention to all relevant groups, and typically must worry more about balance among them than about building share with one of them. The multi-sided approach is consistent with asymmetric treatment of customer groups, but getting it right requires great luck or careful analysis.

The popular network economics literature also suggested that markets with direct or indirect network effects would tend to tip toward a single

24. Joel Kurtzman, "An Interview with W. Brian Arthur," *Strategy+Business* 11 (1998): 100.

provider.[25] That does not happen much in practice, though. Sometimes congestion costs outweigh network effects—that is the case with nightclubs, trading pits, and shopping malls. Platforms also differentiate themselves, and thereby counter the network effects of their rivals, by trying to appeal to different consumer preferences. That is one of the reasons for the proliferation of magazines.

Consider some markets that seem to display important indirect network externalities: PC operating systems, real estate agencies, payment cards, auction houses, local and national newspapers, broadcast networks, parcel delivery services, banks, dating services, standards for encoding DVDs, financial information services, music publishers, and recorded music manufacturers. Of these many markets, the only ones in which a single large player accounts for the preponderance of sales are PC operating systems (i.e., Windows) and some local newspaper markets (such as the *Los Angeles Times*).

Most software platform categories are competitive as a result of providers differentiating themselves to appeal to different types of customers on either side of the market.

Business Models in Multisided Platform Markets Making a platform a success is a delicate process. Businesses have to get the pricing structure right; they must balance the demands of the various customer groups and nurture the several sides of the market. Getting the balance right seems to be more important than building shares. Platform markets do not tip quickly because as a practical matter, it takes time to get things right. And the first entrant often does not win in the end: many other firms may come in and successfully tweak the pricing structure, product design, or business model. eBay is a successful business-to-business (B2B)

25. Some of the network effects models allowed for differentiated tastes and the coexistence of multiple networks. See Jeffrey Church and Neil Gandal, "Network Effects, Software Provision and Standardization," *Journal of Industrial Economics* 40, no. 1 (March 1992): 85–104. S.J. Liebowitz and S.E. Margolis, "Network Externality: An Uncommon Tragedy" *Journal of Economic Perspectives* 8 (1994): 133–150. However, this literature was often taken to suggest overall that network effects naturally lead to a single firm dominating a category. See Brian Arthur, "Increasing Returns and the New World of Business," *Harvard Business Review* 74 (July–August 1996): 100–109.

exchange now, for example, but many earlier B2Bs failed.[26] Most B2Bs tried a big-bang strategy: make substantial investments in a platform and hope both sides show up when the platform opens for trading. The first and third entrants into the payment card industry, Diners Club and Carte Blanche, barely exist today. The second entrant, American Express, had a 14 percent share of credit and debit card purchase volume in 2003.[27]

Getting All Sides on Board An important characteristic of multisided markets is that the demand on each side vanishes if there is no demand on the others, regardless of what the price is. Merchants will not accept a payment card if no customer carries it because no transactions will materialize. Computer users will not use an operating system for which the applications they need to run have not been written (except those rare users who plan to write their own applications). The businesses that participate in such industries have to figure out ways to get both sides on board.

One way to do this is to obtain a critical mass of users on one side of the market by giving them the service for free or even paying them to take it. Especially at the entry phase of firms in multisided markets, it is not uncommon to see precisely this strategy. Diners Club gave away its charge card to cardholders at first—there was no annual fee, and users got the benefit of the float.[28] Netscape gave away its browser to many users, particularly students, to get a critical mass on the end-user side of

26. eBay was not begun as a B2B Web site, but as more and more businesses began to do business on it, it became one. In 2003, eBay officially launched a separate B2B site. "Prior to the B2B site, eBay listed more than 500,000 business items for sale every week on its consumer site, with business buyers representing more than $1 billion in annualized gross merchandise sales, officials said." Renee Boucher Ferguson, "eBay Launches B2B Site," *eWeek*, January 28, 2003 (http://www.eweek.com/print_article/0,3048,a=36363,00.asp); eBay Press Release, "eBay Launches eBay Business to Serve Its Growing Community of Business Buyers," January 28, 2002 (http://investor.ebay.com/news/20030128-100772.cfm); Mark Berniker, "SAP, eBay Setup Industrial B2B Marketplace," *internet News.com*, June 16, 2003 (http://www.internetnews.com/xSP/article.php/2222371).

27. Nilson Report 805 (February 2004).

28. "Credit Cards for Diners," *New York Times*, March 30, 1950, p. 37; Diners Club display advertisement, *New York Times*, March 30, 1950, p. 42.

its business.[29] (Initially the other side was providers of Web sites, to whom Netscape sold its server software.)

Another way to solve the problem of getting the two sides on board simultaneously is to invest to lower the costs of consumers on one side of the market. As we saw earlier, for instance, Microsoft invests in the creation of software tools that make it easier for application developers to write application software for Microsoft operating systems and provides other assistance that makes developers' jobs easier. In some cases, firms may initially take over one side of the business in order to get the market going. Palm would never have succeeded in creating the vibrant Palm economy, with thousands of software applications and hardware add-on developers and millions of users, had it not provided the first applications itself (especially Graffiti, the handwriting recognition system).[30]

Providing low prices or transfers to one side of the market may help the platform solve the simultaneity problem by encouraging the benefited group's participation—which in turn, owing to network effects, encourages the nonbenefited group's participation. In addition, providing benefits to one side can discourage its use of competing multisided platforms. For example, when Palm provides free tools and support to PDA applications software developers, it encourages those developers to write programs that work on the Palm OS platform and automatically induces those developers to spend less time writing programs for other operating systems.[31]

Pricing Strategies and Balancing Interests Firms in mature multisided markets—those that have already gone through the entry phase, in which the focus is on getting the various sides on board—still have to devise

29. David Plotnikoff, "Internet Born with Netscape," *Mercury News*, February 28, 2003 (http://www.tc.umn.edu/~jbshank/7_NetscapeIPO.html).

30. Annabelle Gawer and Michael Cusumano, *Platform Leadership: How Intel, Microsoft and Cisco Drive Industry Innovation* (Boston: Harvard Business School Press, 2003).

31. Jean-Charles Rochet and Jean Tirole, "Platform Competition in Two-Sided Markets," working paper, December 13, 2002; http://www.palmsource.com/developers/why_develop.html.

and maintain an effective pricing structure. In most observed multisided markets, companies seem to settle on pricing structures that are heavily skewed toward one side of the market, as Table 3.1 shows. Google earns the preponderance of its revenue from advertisers, for instance, and real estate brokers usually earn most or all of their revenues from sellers.

Sometimes all competing platforms converge on the same pricing strategy. In principle, Microsoft, Apple, IBM, Palm, and other operating system companies could probably have charged higher fees to applications developers and lower fees to hardware makers or end users. Most discovered that it made sense to charge developers relatively modest fees for developer kits and, especially in the case of Microsoft, to give away a lot for free.

Getting the pricing balance right, however, requires considerable care. For example, in 2000, Yahoo!'s Internet auction site was second only to eBay in terms of the number of listings. Sellers found the site appealing because unlike eBay, Yahoo! did not charge sellers a fee for listing their products. In 2001, Yahoo! changed its pricing strategy and began charging a fee. Yahoo!'s listings dropped by 90 percent as sellers presumably moved to the larger venue, eBay.[32] The price change affected Yahoo!'s buyer-side market as well, since buyers were now left with little to bid on.

Two important factors influence multisided pricing structures. There may be certain customers on one side of the market—Rochet and Tirole refer to them as "marquee buyers"[33]—who are extremely valuable to customers on the other side of the market. The existence of marquee customers who create strong network effects tends to reduce the price to all customers on the same side of the market and increase it to customers on the other side. A similar phenomenon occurs when certain customers are extremely loyal (or captive) to the multisided platform firm, perhaps because of long-term contracts or sunk cost investments. The effect is then opposite: the presence of captive customers leads to an increase in the price charged to those on the same side and a decrease in the price charged to the other side.

32. Saul Hansell, "Red Face for the Internet's Blue Chip," *New York Times*, March 11, 2001, section 3, p. 1.

33. Rochet and Tirole, "Platform Competition in Two-Sided Markets," pp. 23–24.

For example, American Express has been able to charge a relatively high merchant discount as compared to other card brands, especially for its corporate card, because merchants have viewed the American Express business clientele as extremely attractive.[34] Corporate executives on expense accounts were "marquee" customers, who allowed American Express to raise its prices to the other side of the market, merchants. Similarly, marquee customers—in the guise of popular stores, often called anchor tenants—are important for shopping malls as well: by attracting customers they make a mall more attractive to other stores. The decline of a marquee store can sound the death knell for an entire mall.

In the software world, marquee customers are usually businesses on the user side and "killer applications"—an application so innovative and popular that people and businesses buy the computer system mainly because they want the app—on the developer side. VisiCalc was the killer app for the Apple II computer. It was one of the most important reasons behind the initial popularity of this platform. Likewise, Mario Bros. was largely responsible for Nintendo's millions of sales of its NES video game console in the United States, and Sonic the Hedgehog was the main reason for its displacement by Sega's Genesis as the dominant console several years later.

Multihoming As Table 3.2 illustrates, customers on at least one side of a multisided market often belong to several different networks. This is known as multihoming. Take payment cards. Most merchants accept charge, credit, and debit cards associated with several systems; consider how many card symbols there are at the next gasoline pump you use. On the other side of the market, the average consumer has 3.6 payment cards.[35] Advertisers typically place advertisements in several different magazines, and consumers read various magazines.

In general, multihoming by one side of the market relaxes platform competition for that side and intensifies it on the other. For instance, if game developers suddenly become more prone to porting their games to

34. Jon Friedman and John Meehan, *House of Cards: Inside the Troubled Empire of American Express* (New York: Kensington Publishing, 1992), pp. 13, 56.

35. http://www.cardweb.com/cardtrak/pastissues/april2004.html.

Table 3.2
The Presence of Multihoming in Selected Multisided Platforms

Multisided Platform	Sides	Presence of Multihoming
U.S. residential property brokerage	• Buyer • Seller	*Uncommon*: Multihoming may be unnecessary, since a multiple listing service allows the listed property to be seen by all member agencies' customers and agents.
Securities brokerage	• Buyer • Seller	*Common*: The average securities brokerage client has accounts at three firms. Note that clients can be either buyers or sellers, or both.
Newspapers and magazines	• Reader • Advertiser	*Common*: In 1996, the average number of magazine issues read per person per month was 12.3. Also common for advertisers: for example, on August 26, 2003, AT&T Wireless advertised in the *New York Times*, (the) *Wall Street Journal*, and the *Chicago Tribune*, among many other newspapers.
Network television	• Viewer • Advertiser	*Common*: For example, viewers in Boston, Chicago, Los Angeles, and Houston, among other major metropolitan areas, have access to at least four main network television channels: ABC, CBS, FOX, and NBC. Also common for advertisers: for example, Sprint places television advertisements on ABC, CBS, FOX, and NBC.
Operating system	• End user • Application developer	*Uncommon for users*: Individuals typically use only one operating system. *Common for developers*: As noted earlier, the number of developers that develop for various operating systems indicates that developers engage in significant multihoming.
Video game console	• Game player • Game developer	*Varies for players*: The average household (that owns at least one console) owns 1.4 consoles.

Table 3.2
(continued)

Multisided Platform	Sides	Presence of Multihoming
		Common for developers: For example, in 2003, Electronic Arts, a game developer, developed for the Nintendo, Microsoft, and Sony platforms.
Payment card	• Cardholder • Merchant	*Common*: Most American Express cardholders also carry at least one Visa or MasterCard. In addition, American Express cardholders can use Visa and MasterCard at almost all places that take American Express.

both Sony PlayStation 3 and Microsoft Xbox 360, there would be less reason for Sony and Microsoft to hold royalties down to attract developers. Moreover, in this case the two consoles would become closer substitutes from the users' perspective, since they would have more games in common, so one might expect the battle for the end users (many of whom buy only one console) to become fiercer, resulting in lower console prices.

Sometimes unrelated platforms evolve into intersecting ones, which target one or more groups of customers in common; we will see this for digital media platforms. Platform competition can be fierce when either group of customers is price-sensitive because they have other alternatives. The *Houston Chronicle* may have 89 percent of the newspaper readers in Houston, but that does not mean that it can exercise a great deal of pricing power.[36] Advertisers have many other ways of getting messages to readers, so they are sensitive to prices. And while readers may not have many newspaper alternatives, they do have other ways of getting the news, and having a lot of readers is what makes advertisers pay.

36. This number is the total daily circulation of the *Houston Chronicle* divided by the total daily circulation of all daily newspapers in the Houston area. *Circulation* 2003, SRDS (2002), p. 67.

Scaling Many successful multisided firms seem to have adopted a fairly gradual entry strategy in which they scale up their platform over time.[37] Many payment card systems, for example, started in one city or region before expanding nationally. It is often difficult to predict just what the right technology and operations infrastructure will be. Therefore, the multisided firm may find it advantageous to establish efficient buyer-seller transactions and balanced pricing first, and make large investments only after the platform has been tested. Platforms such as eBay, Palm, and Yahoo! have expanded gradually and methodically, building up customers on both sides of their markets.

Strategy Markets hardly ever cooperate with professors by following simple textbook rules exactly. But in traditional markets there are classic truisms that can at least serve as a benchmark, a starting point for more nuanced analysis. By contrast, multisided platforms, especially those in new markets, all too often require clean-sheet planning. With multiple yet interdependent business constituencies to serve, costs provide little guidance for pricing strategies. By the same token, early entry may yield first-mover advantages or provide an instructive failure that simplifies the search for successful strategies by businesses that follow. And, in light of the interdependence between different stakeholders, changes in the business environment may have multisided effects that are very difficult to anticipate.

Open-Source Software

The fortune made by the founders of Google is built in part on the efforts of thousands of volunteers around the world who helped develop the operating system that powers the massive array of server computers that helps us conduct searches and in return peppers us with customized

37. An example of a failed strategy is the case of Chemdex, a business-to-business marketplace, and its parent company, Ventro, which made initial technological and operational investments in the hundreds of millions of dollars (http://www.zdnet.com.au/newstech/enterprise/story/0,2000025001,20107754,00.htm).

advertising messages. Google uses Linux. Like anyone else, it can download this software platform for free from the Internet and customize the source code to meet its own needs. As of 2004, around 19 percent of server computers ran Linux worldwide.[38]

This is almost unthinkable in any other industry. It could not happen in manufacturing, because someone would have to pay for the raw materials to assemble an automobile, for example. Yet even in intellectual property-based businesses such as movies one seldom sees products made by volunteers beating those built by for-profit firms. Linux is the result of the remarkable open-source production model. We turn now to the underlying economics of that model.

Open source is based on a decentralized method for producing software that relies heavily on the Internet. Programmers working on their own or through their companies contribute code to open-source projects. The source code of the resulting programs is made available for free; hence the term open source. Users must sign a license that requires that if they redistribute the program, even in modified form, they must make the source code available. As a result, it is hard to make money directly from open-source programs or anything derived from open-source programs. Open source began as an ideology—"free software is a matter of liberty," according to Richard Stallman—but has evolved into a multi-billion-dollar business based on selling hardware, software, and services that are complementary to open-source programs.[39]

The Production of Open-Source Software

In its early days, individuals who donated their time to work on projects that interested them were the main contributors to open-source software. Typically, a person or a small group of people gets an idea for a project that is interesting, useful, or both. The original developers begin work on the project and eventually solicit support from other interested pro-

38. Al Gillen and Dan Kusnetzky, "Worldwide Client and Server Operating Environments 2004–2008 Forecast: Microsoft Consolidates Its Grip" (ID C report no. 32452), December 2004, table 2.

39. http://www.fsf.org/philosophy/free-sw.html.

grammers. Over the course of the project, programmers, including the original developers, may come and go as they complete work and as their interest waxes or wanes.

The programmers communicate with each other over the Internet. A core group, often consisting of one or more of the original developers, has responsibility for incorporating changes and suggesting things that need to be done. Modified versions of the source code are posted on the Internet and available for free to anyone who wants to use them or modify them further. Over time, users regularly identify bugs that had originally escaped detection, and worthwhile features to add. These users can provide feedback to the developers (or become developers themselves). Through this ongoing process the software becomes tested, debugged, and developed.

The Apache Web server is one of the most successful and famous open-source projects. An early version was written at the National Center for Supercomputing Applications (NCSA) and became the most popular Web server by 1995. Development stalled when Rob McCool, the core developer, left NCSA. Following his departure, some Webmasters began coordinating their fixes via email. Eventually, the Apache group, consisting of eight core contributors, was formed. In April 1995 the first version of the Apache server (version 0.6.2) was released, and became a huge success. The server was completely revamped during the second half of 1995, and Apache 1.0 was released in December 1995. Less than a year after its release, Apache 1.0 became the most popular Web server in the world.[40] The Apache group was incorporated in June 1999 as the Apache Software Foundation. Apache 2.0 was released in 2002, and minor fixes and updates have been periodically released since then. Apache remains the most popular Web server in use, with more than a 50 percent share of its segment.[41]

This production method differs from the commercial approach.

First, there is typically little analysis of consumer needs other than introspection: "What would I like my software to do?" This may be augmented by user feedback, but these users are self-selected; except in

40. http://httpd.apache.org/ABOUT_APACHE.html.

41. http://en.wikipedia.org/wiki/Apache_web_server.

unusual circumstances, they are not drawn randomly from the universe of potential users of the software.

Second, there is little formal testing of the type that commercial firms often must engage in: internal testing using hundreds, perhaps thousands of hardware and software configurations in a controlled manner. Testing is instead performed by the users who try versions of the software in uncontrolled environments, much like beta tests for commercial software developers (although perhaps with more sophisticated users providing feedback to the developers).

Third, the development of open-source software is less structured than the development of proprietary software. Although the core developers may provide direction, changes in the software result much more from individual action.

As open source has evolved, commercial businesses have become more intimately involved in steering open-source projects. They do this by having their employees spend time contributing open-source code and working on the various committees that oversee open-source projects. In a 2003 survey of open-source contributors, nearly 15 percent reported that their employer paid them to develop open-source code, 13 percent noted they were paid to "support" open source, and 13 percent stated they were paid to "administer" open-source projects.[42]

IBM is arguably the best example of a traditional for-profit company with strong ties to open-source software. The bond was officially created in 2000, when IBM announced a $1 billion investment (including marketing expenditures) in a variety of open-source initiatives, including adapting Linux and Apache to IBM's various computer hardware platforms.[43] IBM's hardware business was unusual in that it marketed several fundamentally different types of servers with mutually incompatible operating systems. Adopting Linux permitted IBM to unify its server product line, so that proprietary IBM software (and other software) could be used on all the different servers. By making Linux available on all of its servers, from the smallest to the largest, IBM added consistency to its product line

42. http://www.idei.fr/doc/conf/sic/papers_2005/pdavid_slides.pdf.

43. Joe Wilcox, "IBM to Spend $1 Billion on Linux in 2001," *CNETNews.com*, December 12, 2000 (http://news.com.com/2100-1001-249750.html).

that was missing before. IBM therefore had an incentive to do open-source development that would make Linux run (or run better) on its servers because the investment would provide benefits to IBM.[44]

The open-source investment strategy appears to have paid off handsomely for IBM. For example, China's postal service hired IBM to build Linux-based networks for over 1,200 of its branch offices.[45] A year after its initial $1 billion investment, the company announced that it had already recouped that amount and more.

Intellectual Property Rights

The proponents of open-source software faced a problem. On the one hand they wanted to make open-source software widely available. That meant that they did not want to use copyrights, patents, or trade secrets to limit the distribution of open-source programs. On the other hand, they wanted to make sure that commercial enterprises could not free-ride on the efforts of the open-source community by making minor changes or additions to open-source programs but then enforcing their own intellectual property rights on the entire modified programs.

The General Public License (GPL) was an ingenious solution to this dilemma. The GPL is based on "copyleft":

> You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.[46]

(Despite the copyleft name, the GPL is enforced by copyright law. Copyright is the source of the property protection that enables those who release software under a GPL to impose conditions on others who obtain that code.) The copyleft provision means that if people choose to distribute software that is based in part on other software covered by the GPL, they must distribute their new software under the GPL. GPL soft-

44. James Evans, "IBM to Invest Almost $1 Billion on Linux Development," *InfoWorld*, December 12, 2000 (http://www.infoworld.com/articles/hn/xml/00/12/12/001212hnibmlin.html).

45. http://www.infoworld.com/articles/hn/xml/03/01/23/030123hnibmlinux.html?0124fram.

46. http://www.fsf.org/licensing/licenses/gpl.html.

ware thereby propagates itself. Copyleft makes it difficult for anyone to earn significant profits from selling software code subject to the GPL. As Richard Stallman observed,

> We encourage two-way cooperation by rejecting parasites: whoever wishes to copy parts of our software into his program must let us use parts of that program in our programs. Nobody is forced to join our club, but those who wish to participate must offer us the same cooperation they receive from us.[47]

Proprietary programs can use or communicate with GPL programs in some limited ways without themselves becoming subject to the viral license condition, but the FSF recognizes that the dividing line can be murky. The terms of the GPL apply only to the distribution of software licensed under the GPL, although what "distribution" means in this context is not entirely clear either. It may be possible for an enterprise to modify a GPL program and use it internally without being legally bound to make the source code for its modified version available to others. On the other hand, if the same enterprise distributed its modified GPL program to a subsidiary, the terms of the GPL might well require it to make the source code available to all comers.

Most open-source projects are subject to the GPL. However, several commercial ventures have chosen to use modified licenses. The two most prominent examples are the Common Development and Distribution License (CDDL) that covered Sun's Solaris as it went open source and the Mozilla Public License (MPL) that governs the Firefox browser, among other products. Both contain provisions that GPL does not, and thus code cannot be freely moved between GPL and projects covered under these other licenses. Opponents of this balkanization of open-source licenses contend that it leads to islands of legally incompatible code. For example, owing to different licenses, no cross-pollination between Linux (GPL) and Solaris (CDDL) is possible. Proponents argue that companies have varying needs and catering to these differences is necessary for open-source software to flourish. In addition to relying on more restrictive licenses some open-source software companies are using intellectual property rights to help protect their investments and guard

47. http://www.rons.net.cn/RMS/ms_oss.html.

their profits. Red Hat, for example, has used trademark law to help protect its compilation of Linux from others.

Incentives

The incentives for writing open-source software are different from those for writing commercial software. Many people write open-source code without being paid directly for it. These are volunteers who write code in their spare time because it interests them. Others write code because their companies have asked them to. This may sound traditional but is not, since their employers cannot sell the resulting code or obtain intellectual property rights over it.

Why Individuals Work on Open-Source Software Why programmers donate time to open-source software projects is a subject that has generated considerable discussion.[48] Open-source advocates have suggested several motives, four of which involve nonfinancial rewards:

- It is a good way to learn how to program and develop skills.
- It is fun. Since a programmer is free to pick and choose among open-source projects, he need only work on matters of interest.
- It is prestigious. Success at open-source development rates highly among those whose opinions most programmers most value—other programmers.
- It "scratches an itch." Programmers attack problems that they personally face or because they are intrigued by the intellectual challenge.
- It meets an ideological urge—the desire for free software and the "liberty" it entails.

The "scratches an itch" motive has been considered by some analysts as leading to something like a cooperative of users. A number of developers all consider a particular type of software potentially useful, so they pool their talents to develop the software. With this type of motivation, the GPL has sometimes been considered beneficial as an enforcement mechanism: it ensures that no one can take the collective intellectual

48. For a more detailed discussion of open-source software development, see Joshua Lerner and Jean Tirole, "Some Simple Economics of Open Source," *Journal of Industrial Economics*, 2002; Joshua Lerner and Jean Tirole, "The Open Source Movement: Key Research Questions," *European Economic Review* 45, nos. 4–6 (2001): 819–826.

property, add some private intellectual property, and treat the whole as a private good.

Business Models Based on the Open-Source Concept Businesses have incentives to "donate" employees to the development of open-source projects that stimulate the demand for other products or services sold by the firm. This has become an increasingly large source of labor for open-source projects.[49] IBM and Red Hat illustrate the motivations. As discussed earlier, IBM's model is built on driving the sales of its key products: supporting Linux software increases IBM's sales of hardware, proprietary software, and services. Linux offered a way for IBM to integrate its entire line of servers without having to develop a software platform of its own, and without having to shoulder the continued support and development of that system on its own.

Red Hat is a somewhat different story. That company began as a pure open-source vendor offering a distribution of Linux. Over time, it has gradually moved toward more traditional software licensing, presumably because it is difficult to support a for-profit company with a pure open-source business model. Red Hat is focused on solving a problem inherent in the Linux development model. For major proprietary operating systems such as Windows, the components of the software are integrated by the distributor and sold as a single program. Since no one developer exists for Linux, bits and pieces of the operating system tend to float around—some in forms unusable by nonprofessionals. Specific Linux distributions consolidate these bits and pieces into a convenient package.

Red Hat is arguably the premiere Linux distribution, with more than 46 percent of Linux server distribution shipments in 2004.[50] The company was founded in 1995 and subsequently enjoyed significant

49. "OpenOffice Team Wants IBM Contribution," *VNUNet*, April 25, 2005; Timothy Prickett Morgan, "Novell Creates Hula Open Source Collaboration Server," *ComputerWire,* February 16, 2005 (http://www.computerwire.com/industries/research/?pid=23554158-E49D-4ED2-9482-72B4B5D4119F&type=CW%20News).

50. Al Gillen, Milla Kantcheva, and Dan Kusnetzky, "Worldwide Linux Operating Environments 2005–2009 Forecast and Analysis: Product Transitions Continue" (IDC report no. 34390), table 2.

growth, topped off with an IPO in 1999 that generated the eighth biggest first-day percentage gain in Wall Street history.[51] Like many other high-tech companies, Red Hat lost quite a bit of value in the dot-com crash, but it has since rebounded successfully.

Red Hat really does three things. First, it integrates components of Linux into a cohesive distribution, including commonly used open-source products along with the core operating system. Second, it adds its own software to provide a better user experience and to make installation and updating easier. Third, it sells support packages and certifies that external administrators are qualified to work on Red Hat products. The company includes only open-source software, and the code it writes is licensed mainly under the GPL.

Red Hat changed its business model drastically in 2003 by splitting its distribution into two products—the Fedora Project, a more traditional open-source project, and Red Hat Enterprise Linux (RHEL), the flagship product. Fedora is the place for experiments to run and outside developers to submit code, while RHEL is a stable version of Linux for paying customers. Along with the split came a new licensing agreement. RHEL source code is available for free from Red Hat, but the code computers need to run the operating system is available only with the purchase of a support subscription.[52] And support subscriptions must be purchased for *each* computer, just like traditional proprietary software licenses.

One fundamental problem with generating revenue from GPL software is that anyone can take the source code, "compile" it for computers to read, and resell it without incurring the original creator's development costs. Red Hat has tried to sidestep this problem. Another company could rebuild RHEL from freely available source code, but it would have to strip out all references to Red Hat to comply with trademark law. Purchasers could not be certain that the distribution really contained all of the pieces in Red Hat's, or that the installation would

51. W. G. Rohm, "Inside The Red Hat IPO," *Linux Magazine*, November 15, 1999 (http://www.linux-mag.com/1999-11/redhatipo_01.html).

52. Advanced users could compile the source code themselves. As discussed later in the chapter, this has many disadvantages for corporate customers.

work as seamlessly. Thus, Red Hat has used its reputation in combination with trademark law to limit the potential for another company to undercut its profits.

Open source seems so New Age. Yet when one looks over the history of the computer industry, it turns out that the business of selling software—including software platforms—really didn't take hold until the late 1970s. Microprocessors created a mass market for software that attracted entrepreneurs. Many of these pioneers wrote applications that would run on the new microprocessor-based personal computers. A few focused on refining software platforms whose shared code could be used by many developers and customers at the same time. The next two chapters examine the almost contemporaneous birth of software platforms on PCs and video game consoles.

INSIGHTS

- Like other information goods, software platforms are produced by educated workers, are malleable and easily changed, and are reproducible at virtually no cost.
- Bundling features into the software platform is often efficient for the platform producer and for end users, as it is for most information goods, because it lowers distribution costs and expands demand.
- Software platforms create value by reducing the costs for their multiple customer groups to come together and thereby enhance the value that each customer group delivers to the other. They do this mainly through providing shared services—made available through APIs—that reduce costs for developers and users.
- Multisided platforms must consider marginal costs and price sensitivity in pricing, like single-sided businesses, but they must also consider which side values the other side more. Software platforms generally charge low prices on one side in order to attract customers who can then be made available to the other side. Getting the balance right among all sides is more important than building market share.

• Commercial and open-source production methods have both proved viable models for producing software platforms. Commercial methods seem better suited for managing the multisided aspects of platforms, while open-source methods have produced reliable platforms and applications.

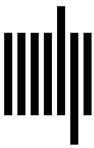
Invisible Engines

How Software Platforms Drive Innovation and Transform Industries

By: David S. Evans, Andrei Hagiu, Richard Schmalensee

Citation:

Invisible Engines: How Software Platforms Drive Innovation and Transform Industries
By: David S. Evans, Andrei Hagiu, Richard Schmalensee
DOI: 10.7551/mitpress/3959.001.0001
ISBN (electronic): 9780262272421
Publisher: The MIT Press
Published: 2008



The MIT Press

MIT Press books may be purchased at special quantity discounts for business or sales promotional use. For information, please email special_sales@mitpress.mit.edu or write to Special Sales Department, The MIT Press, 55 Hayward Street, Cambridge, MA 02142.

This book was set in Sabon by SNP Best-set Typesetter Ltd., Hong Kong.
Printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Evans, David S. (David Sparks)
Invisible engines : how software platforms drive innovation and transform industries / David S. Evans, Andrei Hagiu, and Richard Schmalensee.
p. cm.
Includes bibliographical references and index.
ISBN 0-262-05085-4 (alk. paper)
1. Application program interfaces (Computer software). 2. Industries—Data processing. I. Hagiu, Andrei. II. Schmalensee, Richard. III. Title.
QA76.76.A63 E93 2006
005.3—dc22

2006046629

10 9 8 7 6 5 4 3 2 1