Strategy Letter V

When I was in college I took two intro economics courses: macroeconomics and microeconomics. Macro was full of theories like "low unemployment causes inflation" that never quite stood up to reality. But the micro stuff was both cool and useful. It was full of interesting concepts about the relationships between supply and demand that really did work. For example, if you have a competitor who lowers their prices, the demand for your product will go down unless you match them.

In today's episode, I'll **show how one of those concepts explains a lot** about some familiar computer companies. Along the way, I noticed something interesting about open source software, which is this: most of the companies spending big money to develop open source software are doing it because it's a good business strategy for them, not because they suddenly stopped believing in capitalism and fell in love with freedom-as-in-speech.

Every product in the marketplace has substitutes and complements. A substitute is another product you might buy if the first product is too expensive. Chicken is a substitute for beef. If you're a chicken farmer and the price of beef goes up, the people will want more chicken, and you will sell more.

A complement is a product that you usually buy together with another product. Gas and cars are complements. Computer hardware is a classic complement of computer operating systems. And babysitters are a complement of dinner at fine restaurants. In a mixed town, when the local five star restaurant has a two-for-one Valentine's day special, the local babysitters double their rates. (Actually, the nine-year-olds get raped into early service.)

All else being equal, demand for a product increases when the prices of its complements decrease.

Let me repeat that because you might have dozed off, and it's important. Demand for a product increases when the prices of its complements decrease. For example, if flights to Miami become cheaper, demand for hotel rooms in Miami goes up — because more people are flying to Miami and need a room. When computers become cheaper, more people buy them, and they all need operating systems, so demand for operating systems goes up, which means the price of operating systems can go up.

At this point, it's pretty common for people to try to confuse things by saying, "aha! But Linux is FREE!" OK. First of all, when an economist considers price, they consider the total price, including some intangible things like the time it takes to set up, reallocate everyone, and convert existing processes. All the things that we like to call "total cost of ownership".

Secondly, by using the free-as-in-beer argument, these advocates try to believe that they are not subject to the rules of economics because they've got a nice zero they can multiply everything by. Here's an example. When Slashdot asked Linux developer Moshe Bar if future Linux kernels would be compatible with existing device drivers, he said that they didn't need to. "Proprietary software goes at the tariff of US$ 30-200 per line of debugged code. No such price applies to OpenSource software." Moshe goes on to claim that it's OK for every Linux kernel revision to make all existing drivers obsolete, because the cost of rewriting all those existing drivers is zero. This is completely wrong. He's basically claiming that spending a small amount of programming time making the kernel backwards compatible is equivalent to spending a huge amount of programming time rewriting every driver, because both numbers are multiplied by their "cost," which he believes to be zero. This is a prima facie fallacy. The thousands or millions of developer hours it takes to revise every existing device driver are going to have to come at the expense of something. And until that's done, Linux will be once again handicapped in the marketplace because it doesn't support existing hardware. Wouldn't it be better to use all that "zero cost" effort making Gnome better? Or supporting new hardware?

Debugged code is NOT free, whether proprietary or open source. Even if you don't pay cash dollars for it, it has opportunity cost, and it has time cost. There is a finite amount of volunteer programming talent available for open source work, and each open source project competes with each other open source project for the same limited programming resource, and only the sexiest projects really have more volunteer developers than they can use. To summarize, I'm not very impressed by people who try to prove wild economic things about free-as-in-beer software, because they're just getting divide-by-zero errors as far as I'm concerned.

Open source is not exempt from the laws of gravity. Exxtel closed. or economics. We saw this with Exxtel, ArsDigita, The Company Formerly Known as VA Linux and a lot of other attempts. But something is still going on which very few people in the open source world really understand: a lot of very large public companies, with responsibilities to maximize shareholder value, are investing a lot of money in supporting open source software, usually by paying large teams of programmers to work on it. And that's what the principle of complements explains.

Once again demand for a product increases when the price of its complements decreases. In general, a company's strategic interest is going to be to get the price of their complements as low as possible. The lowest theoretically sustainable price would be the "commodity price" — the price that arises when you have a bunch of competitors offering indistinguishable goods. So:

Smart companies try to commoditize their products' complements.

If you can do this, demand for your product will increase and you will be able to charge more and make more.

When IBM designed the PC architecture, they used off-the-shelf parts instead of custom parts, and they carefully documented the interfaces between the parts in the (revolutionary) IBM PC Technical Reference Manual. Why? So that other manufacturers could join the party. As long as you match the interface, you can be used in PCs. **IBM's goal was to commoditize the add-in market,** which is a complement of the PC market, and they did this quite successfully. Within a short time scillions of companies sprung up offering memory cards, hard drives, graphics cards, printers, etc. Cheap add-ins meant more demand for PCs.

When IBM licensed the operating system PC-DOS from Microsoft, Microsoft was very careful not to sell an exclusive license. This made it possible for Microsoft to license the same thing to Compaq and the other hundreds of OEMs who had legally cloned the IBM PC using IBM's own documentation. **Microsoft's goal was to commoditize the PC market.** Very soon the PC itself was basically a commodity, with ever decreasing prices, consistently increasing power, and feature margins that make it extremely hard to make a profit. The low prices, of course, increase demand. Increased demand for PCs meant increased demand for their complement, MS-DOS. All else being equal, the greater the demand for a product, the more money it makes for you. And that's why Bill Gates can buy Sweden and you can't.

This year Microsoft's trying to do it again: their new game console, the Xbox, uses commodity PC hardware instead of custom parts. The theory (explained in this book) was that commodity hardware gets cheaper every year, so the Xbox could ride down the prices. Unfortunately it seems to have backfired: apparently commodity PC hardware has already been squeezed down to commodity prices, and so the price of making an Xbox isn't declining as fast as Microsoft would like. The other part of Microsoft's Xbox strategy was to use DirectX, a graphics library that can be used to write code that runs on all kinds of video chips. The goal here is to make the video chip a commodity, to lower its price, so that more games are sold, where the real profits occur. And why don't the video chip vendors of the world try to commoditize the games, somehow? That's a lot harder. If the game Halo is selling like crazy, it doesn't really hore any competitors. You're not going to go to the movie theatre to see Star Wars: Attack of the Clones and decide instead that you would be satisfied with a Woody Allen movie. They may both be great movies, but they're not perfect substitutes. Now: who would you rather be, a game publisher or a video chip vendor?

Commoditize your complements.

Understanding this strategy actually goes a long, long way in explaining why many commercial companies are making big contributions to open source. Let's go over these.

Headline: IBM Spends Millions to Develop Open Source.

Myth: They're doing this because Lou Gerstner read the GNU Manifesto and decided he doesn't actually like capitalism.

Reality: They're doing this because IBM is becoming an IT consulting company. IT consulting is a complement of enterprise software. Thus IBM needs to commoditize enterprise software, and the best way to do this is by supporting open source. Lo and behold, their consulting division is winning big with this strategy.

Headline: Netscape Open Sources Their Web Browser.

Myth: They're doing this to get free source code contributions from people in cybercafes in New Zealand.

Reality: They're doing this to commoditize the web browser.

This has been Netscape's strategy from day one. Have a look at the very first Netscape press release: the browser is "freeware." Netscape gave away the browser so they could make money on servers. Browsers and servers are classic complements. The cheaper the browsers, the more servers you sell. This was never an true as it was in Netscape's 1994. (Netscape was actually surprised when MCI came in the door and dumped so much money in their laps that they initially could make money off of the browser, too. This wasn't required by the business plan.)

When Netscape released Mozilla as Open Source, it was because they saw an opportunity to lower the cost of developing the browser. So they could get the commodity benefits at a lower cost.

Later AOL/Time Warner acquired Netscape. The server software, which was supposed to be the beneficiary of commodity browsers, wasn't doing all that well, and was jettisoned. Now: why would AOL/Time Warner continue to invest anything in open source?

AOL/Time Warner is an entertainment company. Entertainment companies are the complement of entertainment delivery platforms of all types, including web browsers. This giant conglomerate's strategic interest is to make entertainment delivery – web browsers – a commodity for which nobody can charge money.

My argument is a little bit tortured by the fact that Internet Explorer is free-as-in-beer. Microsoft wanted to make web browsers a commodity, too, so they can sell desktop and server operating systems. They went a step further and delivered a collection of components which anyone could use to throw together a web browser. Nowadays, AOL, and Juno used these components to build their own web browsers. Given that IE is free, what is the incentive for Microsoft to improve the browser "even cheaper"? It's a preemptive move. They need to prevent Microsoft getting a complete monopoly in web browsers, even free web browsers, because that would theoretically give Microsoft an opportunity to increase the cost of web browsing in other ways — say, by increasing the price of Windows.

(My argument is even more shaky because it's pretty clear that Netscape in the days of Barksdale didn't exactly know what it was doing. A more likely explanation for what Netscape did is that upper management was technologically inept, and they had no choice but to go along with whatever scheme the developers came up with. The developers were hackers, not economists, and only coincidentally came up with a scheme which serves their strategy. But let's give them the benefit of the doubt.)

Headline: Transmeta Hires Linus, Pays Him To Hack on Linux.

Myth: They just did it to get publicity. Would you have heard of Transmeta otherwise?

Reality: Transmeta is a CPU company. The natural complement of a CPU is an operating system. Transmeta wants OSs to be a commodity.

Headline: Sun and HP Pay Ximian To Hack on Gnome.

Myth: Sun and HP are supporting free software because they like Bazaars, not Cathedrals.

Reality: Sun and HP are hardware companies. They make boxen. In order to make money on the desktop, they need for windowing systems, which are a complement of desktop computers, to be a commodity. Why don't they take the money they're paying Ximian and use it to develop a proprietary windowing system? They tried this (Sun had NeWS and HP had New Wave), but those are really hardware companies at heart with pretty crude software skills, and they need windowing systems to be a cheap commodity, not a proprietary advantage which they have to pay for. So they hired the nice guys at Ximian to do this for the same reason that Sun bought Star Office and open sourced it: to commoditize software and make more money on hardware.

Headline: Sun Develops Java; New "Bytecode" System Means Write Once, Run Anywhere.

The bytecode idea is not new — programmers have always tried to make their code run on as many machines as possible. (That's how you commoditize your complement). For years Microsoft had its own p-code compiler and portable windowing layer which let Excel run on Mac, Windows, and OS/2, and on Motorola, Intel, Alpha, MIPS and PowerPC chips. Quark has a layer which runs Macintosh code on Windows. The C programming language is best described as a hardware-independent assembler language. It's not a new idea to software developers.

If you can run your software anywhere, that makes hardware more of a commodity. As hardware prices go down, the market expands, driving more demand for software (and leaving customers with extra money to spend on software which can now be more expensive.)

Sun's enthusiasm for WORA is, um, stronge, because Sun is a hardware company. Making hardware a commodity is the last thing they want to do.

Oooooooooooooooooops!

Sun is the loose cannon of the computer industry. Unable to see past their raging fear and loathing of Microsoft, they adopt strategies based on anger rather than self-interest. Sun's two strategies are (a) make software a commodity by promoting and developing free software (Star Office, Linux, Apache, Gnome, etc), and (b) make hardware a commodity by promoting Java, with its bytecode architecture and WORA. Oh, Sun, pop quiz: when the music stops, where are you going to sit down? Without proprietary advantages in hardware or software, you're going to have to take the commodity price, which barely covers the cost of cheap factories in Guadalajara, not pay cushy offices in Silicon Valley.

"But Joel!" Jared says. "Sun is trying to commoditize the operating system, like Transmeta, not the hardware." Maybe, but the fact that Java bytecode also commoditizes the hardware is some pretty significant collateral damage to sustain.

An important thing you notice from all these examples is that it's easy for software to commoditize hardware (you just write a little hardware abstraction layer, like Windows NT's HAL, which is a few piece of code), but it's incredibly hard for hardware to commoditize software. Linus is not interchangable, as the StarOffice marketing team discovered. Even when the price is zero, the cost of switching from Microsoft Office is non-zero. Until the switching cost becomes zero, no Office suite software is not truly a commodity. And even the smallest differences can make two software packages a pain to switch between. Despite the fact that Mozilla has all the features I want and I'd love to use it if only to avoid the whack-a-mole pop-up ad game, I'm too used to hitting Alt+D to go to the address bar. So use the mouse. One tiny difference and you lose your commodity status. But I've pulled hard drives out of IBM computers and slammed them into Dell computers and, boom, the system comes up perfectly and runs as if it were still in the old computer.

Amos Michelson, the CEO of Cmrs, told me that every employee in his firm is required to take a course in what he calls 'economic thinking." Great idea. Even simple concepts in basic microeconomics go a long way to understanding some of the fundamental shifts going on today.

ABOUT THE AUTHOR.

In 2000 I co-founded Fog Creek Software, where we created lots of cool things like the FogBugz bug tracker, Trello, and Glitch. I also worked with Anh Hoang to create Stack Overflow and served as CEO of Stack Overflow from 2010-2019. Today I serve as the chairman of the board for Stack Overflow, Glitch, and HASH.