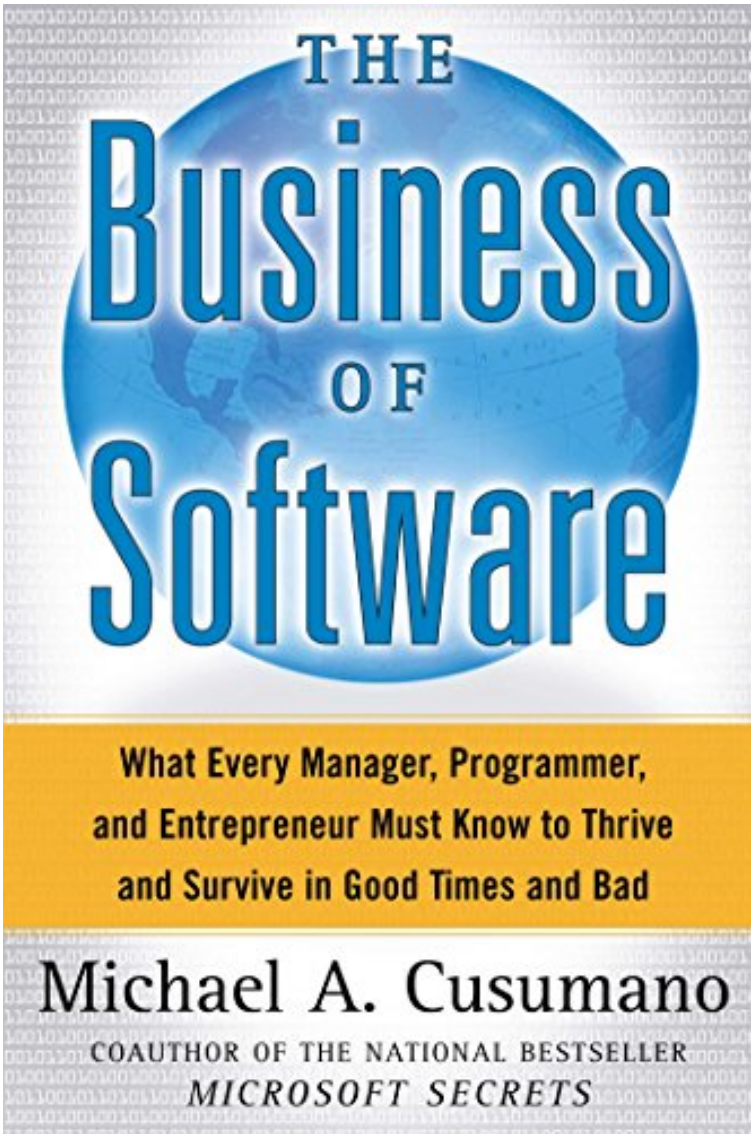
The Business of Software: What Every Manager, Programmer, and Entrepreneur Must Know to Thrive and Survive in Good Times and Bad

Michael A. Cusumano
*DOC | *audiobook | ebooks | Download PDF | ePub*



#918554 in eBooks 2004-03-15 2004-03-02File Name: B000FC1B56 | File size: 40.Mb

Michael A. Cusumano : The Business of Software: What Every Manager, Programmer, and Entrepreneur Must Know to Thrive and Survive in Good Times and Bad before purchasing it in order to gage whether or not it would be worth my time, and all praised The Business of Software: What Every Manager, Programmer, and Entrepreneur Must Know to Thrive and Survive in Good Times and Bad:

1 of 1 people found the following review helpful. Good overview for the uninitiated, tedious if you know the businessBy IraqiInAmericaMr. Cusumano attempted to provide an overview of the software business from its early history to current times. Mr. Cusumano builds on two major sources, his close familiarity with two companies, i2 and Business Objects and from the history of IBM and Microsoft, which in my opinion is not a broad enough base. Secondly, he does not delve deeply enough into any of the topics, and glosses over some of the more interesting ones (at least from my perspective), such as services and the business behind them. On the other hand, he spends a full chapter (60 pages, or 20% of the book's length) on best-practice development methodologies. Frankly, no one will obtain sufficient knowledge to manage any development effort from this review, yet the space taken by it could have been dedicated to topics more in-line with the book's title. Also, the book seems to have been written in several stages and the gaps in timing are obvious. For example, despite the book being dated to 2004, it occasionally refers to Accenture as Andersen Consulting while in other places it is called Accenture (name established in 2000). There are a few additional such examples.Conclusion - it is not a bad review of the history of the software business and some of the changes it went through over the years. Sadly, it overextends itself and therefore suffer from lack of focus. Too bad.0 of 0 people found the following review helpful. Great Software Business ScopeBy Rodil RiveraIf you are in the business of software, or planning to develop any software and selling it, this is a great way to know on what you are getting into. I run a Entrepreneur style Software Business company in Honduras, Central America, and this the book has clear my mind about alot of things that everyone inside the SW business should know. Basically the book covers this topics:-Strategic thinking for the SW Business: great chapter with a big scope on what is a SW strategy and how does it work.-History of SW business: it gives a good reference about the different trends the SW industry has gone through.-SW Business for entrepreneurs: he gives a 8 points frame to be aware of.-Start-up cases: he shows the different strategies that companies have used and about the output they have gotten, and everything under sight of the 8 points frame.0 of 0 people found the following review helpful. Solid book on softwareBy M. LinThis book was part of a course that several friends took at MIT Sloan. I bought it when I started a role and needed to beef up my software knowledge. I think that Michael, the author, really understands the software business. It was a well-written book and very educational. That said, it doesn't really come to mind when I'm recommending my favorite business books to anyone - so it wasn't incredibly memorable for me.

The world's leading expert on the global software industry and coauthor of the bestseller Microsoft Secrets reveals the inner workings of software giants like IBM, Microsoft, and Netscape and shows what it takes to create, develop, and manage a successful company -- in good times and bad -- in the most fiercely competitive business in the world. In the $600 billion software industry it is the business, not the technology, that determines success or failure. This fact -- one that thousands of once glamorous start-ups have unhappily discovered for themselves -- is the well-documented conclusion of this enormously readable and revealing new book by Michael Cusumano, based on nearly twenty years of research and consulting with software producers around the world. Cusumano builds on dozens of personal experiences and case studies to show how issues of strategy and organization are irrevocably linked with those of managing the technology and demonstrates that a thorough understanding of these issues is vital to success. At the heart of the book Cusumano poses seven questions that underpin a three-pronged management framework. He argues that companies must adopt one of three basic business models: become a products company at one end of the strategic spectrum, a services company at the other end, or a hybrid solutions company in between. The author describes the characteristics of the different models, evaluates their strengths and weaknesses, and shows how each is more or less appropriate for different stages in the evolution of a business as well as in good versus bad economic times. Readers will also find invaluable Cusumano's treatment of software development issues ranging from architecture and teams to project management and testing, as well as two chapters devoted to what it takes to create a successful software start-up. Highlights include eight fundamental guidelines for evaluating potential software winners and Cusumano's probing analysis, based on firsthand knowledge, of ten start-ups that have met with varying degrees of success. The Business of Software is timely essential reading for managers, programmers, entrepreneurs, and others who follow the global software industry.

From Publishers WeeklyCusumano, a professor at MIT's Sloan School of Management and coauthor of Microsoft Secrets, offers a comprehensive overview of the software business and how the right approach is key to the success of technology companies. Cusumano first identifies the key distinction between software and other businesses. In fact, he believes it is unlike every other business because software doesn't have one purpose but becomes whatever function it is handling for a particular customer or company. As a result, software companies must sell both products and services, according to the author. The two typical ways software companies operate is by getting the lion's share of revenues from new product sales or via IT consulting. The third way is what the author calls "hybrid solutions companiesmdash;software firms that have some new product sales, but derive as much as 80% of their revenues from services and "maintenance." However, what's essential for company success in today's rapidly changing technological marketplace is having sufficient flexibility to change to meet customer needs. Citing both real companies including

IBM, Netscape, etc., along with academic studies, Cusumano describes the changing face of the software industry over the past two decades. The writing is coherent and, given the somewhat technical subject matter, surprisingly graspable even for technophobes. Still, this is a niche book, apt to appeal to people involved in the world of software, rather than a general business audience. About the AuthorMichael A. Cusumano is the Sloan Management Distinguished Professor at the MIT Sloan School of Management and one of the world's leading authorities on software development and the management of software companies. He is the author or coauthor of seven other books, including the bestsellers Microsoft Secrets and Competing on Internet Time, as well as Japan's Software Factories and most recently Platform Leadership.Excerpt. Preface My goal in this book is to provide an overview of the software business for managers who are already working in the business, programmers who would like to be managers, and anyone who would like to be a software entrepreneur. I focus mainly on firms selling what we can call "enterprise software" to other companies and large organizations, although much of what I say about products, services, and software development also applies to companies selling software to individual consumers. My primary concerns are with strategy and business models, a historical look at software entrepreneurship, best practices in managing software development, and the do's and don'ts of founding a software start-up. The examples and topics reflect my personal experiences as a researcher, teacher, consultant, director, and company founder. I have used a version of this book in my MIT class "The Software Business." The material should also be a useful reference for investors and analysts who follow software companies and for anyone else interested in how high-tech firms deal with problems of strategy and product development in rapidly changing markets. Chapter 1 starts out with a personal sketch of my involvement in the software business and how the business seems to differ around the world. I also sketch out the experiences of two firms that I have worked with recently and that demonstrate many of the strategic and business-model issues that I take up in subsequent chapters: Business Objects in France and i2 Technologies in the United States. Chapter 2 focuses on strategy for software companies -- the most important things that managers and entrepreneurs, as well as programmers, should think about. I begin with the most fundamental question: Do you want to be mainly a products company or a services company? I also talk in more detail about a third alternative: hybrid solutions. Other key issues are the kinds of customers and markets (enterprise, consumer, mass-market, niche, vertical, or horizontal) that the company will target, as well as platform versus complementor decisions. Chapter 3 probes these issues of strategy, management, and entrepreneurship in more depth. I use history to suggest where most opportunities in the business have come from and how the industry has evolved from services to products and now back, to some extent, to a renewed emphasis on services, especially in bad economic times, when customers are reluctant to buy more software products. I also examine the history of IBM and end with a discussion of open-source and "for free" software. Chapter 4 presents what I have learned about how to manage the most fundamental technical activity in a software company: software development. The discussion is not specifically for programmers, but is more for managers of programmers and general managers who need to know what goes on in software projects and how best to deal with issues ranging from architecture and teams to project management and testing. I begin with basic problems in software development and how organizations introduced "software factories" and process improvement initiatives from the Software Engineering Institute (SEI). I then discuss practices that provide an effective balance of structure and flexibility. In particular, I review "synch-and-stabilize" techniques that have characterized software products companies such as Microsoft and Netscape, and how managers can apply these concepts in different settings. I also include some preliminary observations from survey data on the practices and performance of software companies in the United States, Japan, India, Europe, and elsewhere. Chapters 5 and 6 apply ideas about the software business to start-ups. I give my view of how to interpret entrepreneurship in times of boom and bust and reflect on what the most important elements in a successful software venture are. I then evaluate ten case studies of start-ups I have been involved with since the mid-1990s. I also use these examples to illustrate strategic issues as well as differences among the three main business models for software companies: products, services, and hybrid solutions. Chapter 7, the conclusion, offers some final thoughts on the three basic models and the different capabilities they require, and contrasts the "ideal" software business with a more realistic business model. I also make some suggestions on how to run a successful software business, whether you are a products company, a services company, or something in between. Chapter 1 The Business of Software: A Personal View If the software business were like other businesses, there would be no need for this book. But software is not like other businesses. First of all, the technology consists of a digital "soft" good -- usually English-like programming commands eventually translated into zeros and ones -- that provide instructions to a computer. These instructions form products that companies can standardize for many users, customize for individual users, or do something in between. Companies that rely on this highly malleable technology for their livelihoods must be unique in many ways, particularly in how they deal with business models, product strategy, people (especially software engineers), and management of a core activity: software development. There are many examples of how software technology and software companies differ from what we see in traditional manufacturing and service industries. In how many businesses does making one copy or one million copies of your product cost about the same? How many businesses

have up to 99 percent gross profit margins for their product sales? In how many businesses do many products companies eventually become services or hybrid companies (that is, providing some customization of product features and technical services such as system integration and maintenance), whether they like it or not? In how many businesses is there frequently a ten- or twentyfold difference in productivity between your best employee and your worst one? How many businesses tolerate some 75 to 80 percent of their product-development projects routinely being late and over budget, with "best practice" considered to be 20 percent on time? How about a company where the people who build products often consider themselves artists rather than scientists or engineers and have the mercurial temperament to go with it? In how many businesses are customers "locked in" to a particular vendor because of product decisions someone made a decade or two ago that can't easily be reversed? The software business also differs from conventional industries because it is not really one kind of business. Software becomes whatever function or application it addresses. This means that the range of possible products and services is almost infinite. Software can help you write a report, calculate your taxes, build a bridge, navigate an automobile, control the space shuttle, or dial your telephone. Not surprisingly, there are many categories and even layers of software products and customized programs that work with one another to form complete systems (such as networking software with operating systems, and operating systems with applications). The definitions of these categories and layers are relatively well accepted, though Microsoft and other companies have been blurring the traditional distinctions for many years. These and other observations describe aspects of software technology, software companies, and the software business. They also describe some other high-tech markets, such as telecommunications and various types of businesses heavily dependent on information systems and digital content. But surely these observations describe an unusual type of business. As I discuss in Chapters 2 and 3, get the strategy and the management side right, and the software business can be like having a license to print money. Just ask Bill Gates of Microsoft or Larry Ellison of Oracle, among many other software billionaires and multimillionaires around the world. But get the business model wrong, and -- to borrow a metaphor from Frederick Brooks's The Mythical Man-Month -- software companies can resemble dinosaurs futilely trying to escape the death grip of an ancient tar pit. The more you struggle -- that is, the more time, money, and people you pour into product development, sales, and marketing in the hope of a turnaround -- the deeper you sink and the quicker you die. In the software business, this is not only because the more people you add to a late software project, the later the project can become -- a rule of thumb now described as "Brooks's law" (and not always true). But the broader downward spiral can accelerate for a whole company and become self-fulfilling as present and potential customers flee from software producers unlikely to survive long enough to deliver, support, and upgrade their products. In bad economic times, or when there is bad corporate news, we can see the sales of once high-flying software companies suddenly drop as if they had fallen off a cliff. Billion-dollar companies can shrink to half their peak size (e.g., i2 Technologies), declare bankruptcy almost overnight (e.g., Baan), or turn suddenly from modern-day gold mines into investment nightmares. SAP, Oracle, Siebel Systems, Business Objects, and many other blue-chip software companies lost 80 to 90 percent of their value at one time or another during 2000-2002, depending on when an investor bought the stock -- despite having solid products and businesses. Even Microsoft dropped about two thirds of its value during this period of boom and bust. We have seen this phenomenon of rapid growth and dramatic decline as well in the telecommunications equipment and services industries (e.g., Lucent and WorldCom), businesses that are also heavily based on software technology but even more distressed in terms of profits and sales. Since software technology, software compa...