HARVARD | BUSINESS | SCHOOL

Design Rules, Volume 2: How Technology Shapes Organizations

Chapter 16 Capturing Value by Controlling Bottlenecks in Open Platform Systems

Carliss Y. Baldwin

Working Paper 20-054

Design Rules, Volume 2: How Technology Shapes Organizations

Chapter 16 Capturing Value by Controlling Bottlenecks in Open Platform Systems

Carliss Y. Baldwin
Harvard Business School

Working Paper 20-054

Funding for this research was provided in part by Harvard Business School.

Design Rules, Volume 2: How Technology Shapes Organizations

Chapter 16 Capturing Value by Controlling Bottlenecks in Open Platform Systems

By Carliss Y. Baldwin

Note to Readers: This is a draft of Chapter 16 of *Design Rules, Volume 2: How Technology Shapes Organizations*. It builds on prior chapters, but I believe it is possible to read this chapter on a stand-alone basis. The chapter may be cited as:

Baldwin, C. Y. (2019) "Capturing Value by Controlling Bottlenecks in Open Platform Systems," HBS Working Paper (November 2019).

I would be most grateful for your comments on any aspect of this chapter! Thank you in advance, Carliss.

Abstract

The purpose of this chapter is to investigate the means by which firms capture value in open platform systems. I begin by arguing that the surplus value created by complementarities within a technical system will be split among the owners of the unique and essential components—the strategic bottlenecks in the system. However, most platforms must also execute a series of steps which are subject to "flow production" bottlenecks. Finding and fixing these flow bottlenecks is another way to capture value.

In addition, two types of platform improvements provide further opportunities for value capture. "Accelerators" speed up the processing of options, while "subsidiary" platforms increase the range of options available to users. Finally, members of a platform system or new entrants may seek to supplant the owner of a strategic bottleneck by "disintermediating" platform components. I describe four generic methods of disintermediation: substitution; reverse engineering; platform independent complements; and platform envelopment.

Introduction

The purpose of this chapter is to investigate the means by which firms capture value in an open platform system. I combine the theory of a platform as a provider of options with the theory of bottlenecks developed in Chapters 7, 8 and 9. I argue that, in an open platform system, firms capture value by: (1) controlling one or more strategic bottlenecks; (2) finding and fixing flow bottlenecks at the core of the system; (3) introducing accelerators and/or subsidiary platorms; and (4) avoiding disintermediation of the parts of the platform they control.

In focusing on bottlenecks as the key determinants of value capture, I am building on several streams of prior research. In the 1960s, Nathan Rosenberg proposed that what I am calling "technical bottlenecks"—unsolved technical problems constraining different

parts of the system—explained the rate and direction of technical change in many historical settings.[1]

Sendil Ethiraj provided empirical support for Rosenberg's and Hughes' conjectures in a study of the personal computer industry from 1981-1987. Ron Adner and Rahul Kapoor showed that resist chemistry was a technical bottleneck that slowed the transition to deep ultraviolet (DUV) lithography in semiconductors.[2]

In his seminal paper on profiting from innovation, David Teece observed that, in a system of complementary assets, profits (rents) are likely to flow to owners of bottleneck assets. Michael Jacobides, Thorbjorn Knudsen and Mie Augier identified becoming "the bottleneck of an industry" as a key goal of corporate strategy. Their use of the term corresponds to what I am calling "strategic bottlenecks"—components that are essential, unique and owned by a profit-seeking enterprise.[3]

Douglas Hannah and Kathleen Eisenhardt observed firms employing a "bottleneck strategy" in the nascent solar panel industry. Firms employing this strategy addressed a succession of technical bottlenecks turning each into a transient strategic bottleneck. Recently, Rahul Kapoor has argued that tracking bottlenecks, both technical and strategic, is essential to understanding the dynamics of open platforms and their ecosystems.[4]

In this chapter, I begin by arguing that the surplus value created by complementarities within a technical system will be split among the owners of essential and unique components—the strategic bottlenecks of the system. However, the magnitude of the surplus and precise sharing rules will be determined by the timing of moves and product market structure.

A platform exists to support the exercise of options. In performing this function, the platform must execute a series of steps, which are subject to flow production bottlenecks. The "core" of the platform contains those components and activities most likely to impede the platform's performance. Flow bottlenecks in the core are a type of technical bottleneck, and as such may become the basis of strategic bottlenecks in the future.

From an evolutionary perspective, there are two generic types of platform improvements: (1) "accelerators" that speed up processing of options; and (2)

[1] Rosenberg (1967; 1976); Hughes (1993) made a similar argument, but preferred the military term "reverse salient" to "bottleneck."

[2] Ethiraj (2007); Adner and Kapoor (2016)

[3] Teece (1986). Teece (2018) applied the same arguments to digital platforms and ecosystems. Jacobides, Knudsen and Augier (2006).

[4] Hannah and Eisenhardt (2018); Kapoor (2018).

"subsidiary" platforms that increase the range of options available to users. Both types of improvements may give rise to new strategic bottlenecks.

Finally, members of a platform's ecosystem and/or new entrants may seek to remove a strategic bottleneck by "disintermediating" one or more platform components. At the end of the chapter, I describe four generic methods of disintermediation: substitution; reverse engineering; platform independent complements; and platform envelopment.

16.1 The Value Structure of an Open Platform System

To be successful, a platform sponsor must make sure that all components and activities are brought together in a timely, efficient way. *However, the sponsor need not perform all necessary steps itself*. Steps that are not likely to become technical or strategic bottlenecks can be out-sourced without threatening the sponsor's profit. Thus decisions on which components and activities to bring under the sponsor's unified governance (within its zone of authority) are essentially judgments about which elements of the system are likely to be technical or strategic bottlenecks, now or in the future.

Thus we need a way to represent the value of a platform system from which we can predict how the system may evolve over time. Equation 2 from Chapter 13 is a compact and generic representation of the value of a *platform system*.

$$V_p \text{ is proportional to } P \cdot [O_1 + \ldots + O_N] \qquad (1)$$

Here V_p denotes the value of a platform system. P is a binary variable that denotes the presence or absence of the platform. Each O term denotes the value of an option, which can be provided by the platform in conjunction with a complement. By assumption, neither the platform nor any complement has stand-alone value.

In the simplest cases, there is only one level of performance for the platform (it works or it doesn't). In this case, the proportionality relation can be replaced by equality:

$$V_p = P \cdot [O_1 + \ldots + O_N] \qquad (2)$$

(Platforms with different levels of performance are modeled in Section 16-7 below.)

From this expression it is clear that an open platform will be more valuable than a closed platform if (1) the open platform generates *more* options (higher N) than the corresponding closed platform; and/or (2) the value of some or all options is greater in the open system than in a closed system (higher O_i). These conditions typically hold when the knowledge and skills needed to supply the options are widely dispersed and difficult to locate; when the cost to external agents of entering the ecosystem is low; and when high-powered incentives to complementors lead to more highly valued options.

If these conditions hold and property rights are established so that the platform and option providers can capture enough value to cover their costs, distributed

supermodular complementarity (DSMC) will hold between the platform and the option providers. In that case, an open platform system will be a long-term sustainable equilibrium, able to survive competition with closed platform systems.

The platform itself is generally made up of numerous functional components and activities split into several modules. Platform modules in turn are separated by thin crossing points with low transaction costs, hence may be supplied by different firms.

For purposes of developing a theory of value capture, let us assume that the platform is made up of three modules, each performing a different function. For example, the platform might be a personal computer consisting of a microprocessor, an operating system, and an integrated keyboard, display, and disk drive. The optional complements are software applications. For now, I continue to assume that there is only one level of performance for each platform module, thus the presence/absence of each can be denoted by a binary (0/1) variable. Equation (2) can then be expanded to:

$$V_p = P_1 \cdot P_2 \cdot P_3 \cdot \ [O_1 + O_2 + O_3 + \ ... \ + O_N] \ . \qquad (3)$$

In equation (3), each *P* component is essential: the absence of any one makes the value of the system zero. In contrast, the *O* components are optional: if one disappears, the system as a whole still has value because of the other options.

16.2 Technical and Strategic Bottlenecks

To indicate the location of technical and strategic bottlenecks, we can adopt the same notation used in previous chapters: an o-superscript " ^{o} " indicates that the technical recipe for the corresponding component does not exist; an asterisk " * " indicates that the component is unique or contains unique instructions; an asterisk followed by a letter " $*^{X}$ " indicates that the unique component is controlled by a profit-seeking firm, which has the ability to exclude others from using it.

At time 0, I assume that technical recipes for each platform component exist, i.e. there are no technical bottlenecks in the platform. (Otherwise the system itself could not exist.) The following equation illustrates one possible configuration of bottlenecks in the system we are studying:

$$V_p = P_1^{*X} \cdot P_2^{*} \cdot P_3 \cdot \ [O_1^{*Y} + O_2^{*} + O_3 + \ ... \ + O_N^{o}] \ . \qquad (4)$$

Platform component *1* is unique and owned by (profit-seeking) *X*. Platform component *2* is unique, but not owned by a profit-seeker: the technical recipe might be in the public domain or subject to second source agreements so that many firms are capable of supplying the component. Platform component *3* is not unique: its functions can be fulfilled in different ways by different firms. A similar pattern applies to optional components *1*, *2* and *3*. The technical recipe to supply optional component *N* does not yet exist—it is a technical bottleneck, but not a system-wide technical bottleneck.

Property rights theory, transaction cost economics, Teece's theory of profiting from innovation, the Nash bargaining solution, the Shapley value in cooperative game theory, and standard microeconomics all make consistent predictions as to how value will be apportioned in this platform system:

(1) *X* and *Y* will split the value surplus/profit stream created by Option *1*;
(2) *X* will be able to claim the entire surplus/profit stream created by Options *2* and *3*;
(3) The other platform modules and complements will be provided at (close to) marginal cost;
(4) Depending on the terms by which the platform is transferred to users, *X* may be able to claim some of the surplus/profit streams associated with options that do not exist yet, but may exist in the future.

We can interpret these "rules" for value capture in the following way. First, P_1, a unique platform module owned by a profit-seeking entity, constitutes a *system-wide strategic bottleneck*. By virtue of its rights of exclusion, the owner of the strategic bottleneck can claim a share of the surplus or profit stream associated with all the optional components.

How large a share the bottleneck owner can claim depends on the precise structure of the product market. For example, does the owner sell the bottleneck module as a separate product to all consumers at a single price? Or can the owner discriminate among consumers based on their willingness to pay? Alternatively, do the option owners purchase the bottleneck component and resell it to end-users (the value-added-reseller strategy)? Or does the bottleneck owner control access to customers (the apps-store strategy)? These and other configurations can be modeled using standard microeconomic techniques and lead to different surplus amounts and sharing percentages. The theory permits specific outcomes to vary while staying within the rules of value capture given above.

Similarly, O_1, a unique optional module owned by a profit-seeking entity, constitutes a *strategic bottleneck with respect to the option*. The owners of P_1 and O_1 are bilateral monopolists with respect to this option, hence they must split the surplus associated with it. Again the magnitude of the surplus and the sharing ratio depend on the structure of the product market.

16.3 "Free" Options, "Free" Platforms and Platforms with Split Governance

In this section I consider the impact of having a platform and/or options with no strategic bottlenecks, as well as a platform with two (or more) strategic bottlenecks.

First, suppose there is a single platform sponsor controlling a system-wide strategic bottleneck as shown in Equation 4. Consider the supply of optional complements. Those that are unique and owned have a claim to an ongoing stream of profits that justifies investment in the complement. Those that are not owned or not unique have no such claims. There is little or no *financial* incentive to improve these

options, unless the improved designs are both unique and owned.[5] Other things equal, a platform sponsor can expect more investment to be targeted towards areas in which profit-seeking innovators can establish property rights over new or improved options.

It follows that, if third parties can supply more or better options than the platform sponsor, *endowing complementors with property rights in their creations is a rational strategy*.[6] Assigning property rights to suppliers of complements reallocates system value so that the complementors' benefits are better aligned with their costs. Distributed supermodular complementarity (DSMC) is then more likely to hold as a dynamic equilibrium. If DSMC holds, an open platform system will survive.

Now suppose that none of the platform modules is owned by a for-profit entity. In that case the platform is "free" for anyone to build on without risk of holdup. Free platforms can be created under the auspices of a commons organization or an open source community. According to property rights theory, free platforms should attract more investment in optional complements than for-profit platforms, *but,* symmetrically, less will be invested in improving the performance of the platform itself.

Thus, if improvements to the platform provide a lot of "bang for the buck" in the system's value function, a privately controlled platform system may outperform a free platform system operating in the same competitive arena. However, it is also possible for complementors to fund standards setting organizations and/or open source communities that take responsibility for maintaining and improving unique platform components. As a result, we cannot conclude that either type of platform—free or privately controlled — will be dominant in any specific domain.

Finally, let us suppose that two unique platform modules are owned by different agents as in Equation 5:

$$V_p = P_1^{*X} \cdot P_2^{*Z} \cdot P_3 \cdot \left[O_1^{*Y} + O_2^{*} + O_3 + \ \ldots \ + O_N^{o}\right] . \qquad (5)$$

According to property rights theory, this is an inferior configuration: both platform claimants will use the threat of holdup (exclusion) to demand a share of the surplus or profit stream from the entire platform system.[7] This in turn diminishes the incentives of all complementors to invest in the platform.

[5] However, in some platform systems, free complements are very common. First, the owner of a free complement may benefit from it in other ways—through advertising and publicity for example. Second, amateurs may create complements as a form of recreation or to gain experience. See Boudreau and Jeppesen (2015) and Boudreau (2018) on unpaid complementors and von Hippel (2016) on free innovation.

[6] Arora and Merges (2004) make this point in the context of buyer-supplier relations. Parker and Van Alstyne (2017) derive the optimal degree of openness and duration of complementors' property rights for a platform with two levels of options.

[7] Hart and Moore (1990), p. 1135, Proposition 8: If two or more assets are strong complements, they should be owned or controlled together.

However, property rights theory assumes a static technology and a fixed surplus.[8] It does not contemplate the possibility of a dynamic technical system fueled by supermodular complementarity.

To clarify this point, suppose a new platform component is invented that enhances the value of every existing option. For the sake of concreteness, let us assume that the original surplus was 100, split 50-50 between the original platform sponsor and option owners. If the surplus with the addition of the new component is greater than 150, then the original platform sponsor and all option owners (not to mention users) will be better off with the second strategic bottleneck than without it. To a first approximation, the owner of the second strategic bottleneck will claim one-third of the *new* total surplus, the owner of the first strategic bottleneck will claim one-third, and option owners will claim one-third. But if enough complementary value is created by the new component, all claimants will be better off.

Notwithstanding this possibility, an incumbent platform sponsor still has strong incentives to invest to prevent the emergence of other strategic bottlenecks. In general, new technical bottlenecks will emerge as the platform evolves and functions once deemed optional may come to be essential. To maintain its privileged position, the original platform sponsor must keep unique solutions to essential technical bottlenecks within its own zone of authority *or* place such assets under the governance of non-profit-seeking actors such as commons organizations and open source communities.

16.4 Hierarchies of Platforms

Multiple strategic bottlenecks can also emerge in a single platform system when one platform is built upon another in a hierarchical fashion. For example, as shown in Equation 6, suppose the owner of an optional complement designs it as a platform supporting derivative options.

$$V_p = P_1^{*X} \cdot P_2^{*} \cdot P_3 \cdot \; [O_1^{*Y} \cdot \; \{o_1^{*Z} \; + \; o_2 + \; \ldots \; + \; o_N\} + O_2 \ldots \;] \quad . \qquad (6)$$

In the equation, a unique platform component, P_1, is controlled by agent X; a unique optional component, O_1, by agent Y; and a unique derivative option, o_1 by agent Z. O_1, is a binary variable indicating the presence or absence of the subsidiary platform. The basic platform might be a computer or smartphone, the subsidiary platform a software program with APIs such as Facebook, and the second-level options applications that use those APIs. In this case, by rights of exclusion X and Y can each claim a share of the incremental value created by Z.

In practice, the problem of complements arrayed in a hierarchy is sometimes mitigated by the fact that *rights to use* the platform are unilaterally transferred to users in a sale. For example, if a user purchases a personal computer or phone with an operating system, the rights to use the purchased platform components are transferred to the buyer and (as a general rule) cannot be revoked. For new complementors, the current platform's

[8] Ibid.

installed base amounts to a free platform. Users can purchase new complements (for example, applications) without paying a royalty or tax to the original platform provider. The platform is "doubly open"—open to external complementors and open to entry without payment to the platform sponsor.

16.5 Platform Dynamics

Up to this point we have looked at platform systems as a set of static components, which when combined in a specific way give rise to a functioning system. It is time to take a more dynamic perspective. Movement in computer platform systems occurs at two levels.

First, at the operational level, the technology of digital computation is based on converting instructions into into a flow of electronic waves through a series of circuits and gates. Electronic waves propagate at close to the speed of light, but the speed of instruction processing is affected by the length, complexity, and capacity of the circuits executing the instructions. Bottlenecks in the flow slow down processing speed, reducing the number of computations in a given time period, hence value of the system to users.

Second, at the evolutionary level, the designs of various parts of the system can be improved to increase processing speed and the range of options available to users. In digital systems, we have seen, the fundamental driver of evolutionary change is Moore's Law. However, the circuit line widths and density, which lie at the heart of Moore's Law, are not the only source of evolutionary improvement: the arrangement of circuits, the architecture of the system, and advances in programming methods all play a role as well.

The next section describes how the processing speed of a platform can be improved by finding and fixing bottlenecks in the flow of instructions. The section after shows how to model platform evolutionary dynamics.

16.6 Flow Bottlenecks in Computer Platforms

Instructions flow through computers in the same way that materials flow through the steps of a manufacturing process or assembly line. Beginning in the 1970s and 1980s, under the leadership of John Cocke at IBM, John Hennessy at Stanford, David Patterson at UC Berkeley and their students, computer scientists began to systematically study flows of instructions through different computer systems and to experiment with different ways or arranging devices and circuits to speed up processing time.[9] Carver Mead, Lynn Conway and their students did the same with circuit paths within semiconductor chips.[10]

The movement to systematize and quantify "flow production" processes within computers offers an eery parallel to the systematic and scientific management movements of the late 19th and early 20th centuries, discussed in Part 2. As the individual semiconductor components became individually capable of higher levels of throughput,

[9] Hennessy and Patterson (1990); Patterson and Hennessy (1994)).

[10] Mead and Conway (1980).

the system as a whole became hostage to the slowest elements on the main path of instructions. Finding these "flow production" bottlenecks and fixing them was the primary goal of a new field called "quantitative computer architecture."[11]

Just as automated machinery and systematic management are supermodular complements, quantitative computer architecture and Moore's Law are a supermodular complements. The higher the *potential* throughput rate of the underlying circuits, the greater is the value of finding and fixing the bottlenecks in the flow of instructions. However, the bottleneck elements in a computer system will change depending on the nature of instructions in the programs it is running. This means that instead of a "one size fits all" architecture, a computer needs to be optimized for its most common load. For example, gaming computers require fast video processing, while virtual machines must be capable of fast interrupts.

Optimizing a computer for the programs it most often runs is the fundamental principle underlying the quantitative computer architecture. The principle is often referred to as Amdahl's Law—"Make the common case fast."[12] Today the speed of computers is measured, not by the clock speed of the central processor, but by running benchmark programs reflecting different use cases. Moore's Law and Amdahl's Law both contribute to improvements in realized throughput from one generation to the next.

Core of the Platform

A platform component that is (1) not unique; and (2) not on the main path in the flow of instructions (i.e., not a "common case) will not be the cause of a technical bottleneck or the basis of a strategic bottleneck for the system as a whole. Such components do not need to be the focus of attention by platform sponsors, but may be delegated to third parties.

Table 16-1 provides a list of components and activities needed to create an IBM-compatible PC circa 1990.[13] It also indicates whether a given element supplies visible information to the rest of the system; is owned by a for profit entitity; or is on the "main path" of instructions.

As discussed in Chapter 15, the processor, operating system, buses, and BIOS all provided unique, visible information to the rest of the system. However, the bus designs and the BIOS were not owned by a for-profit entity, thus were not the basis of strategic bottlenecks in the mid-1980s.

[11]Hennessy and Patterson (1990); Patterson and Hennessy (1994); Asanovic et al. (2006); Gross et al. (2016).

[12] Hennessy and Patterson (1990).

[13] The list is based on the components shown in Figure 15-6.

Table 16-1 Status of Components and Activities Required to Create an IBM-Compatible PC circa 1990

	Visible Information	Owned	On the "main path"	
Components				
Processor	yes	yes	yes	**Core of the Platform**
Operating System	yes	yes	yes	
Buses and Chipset	yes	no	yes	
BIOS	yes	no	no	
Memory	no	no	yes	
Storage	no	no	no	
Input	no	no	no	
Output	no	no	no	
Activities				
System Integration	no	no	no	
Manufacturing	no	no	no	
Advertising	no	no	no	
Distribution	no	no	no	
Sales	no	no	no	

The original BIOS, though a source of visible information, quickly became outdated. It had to be included in every IBM-compatible system, but by the early 1990s, it had been relegated to the sidelines and was used only to start up the machine.

In terms of throughput, transfers to and from storage, input and output devices were relatively infrequent thus did not generally slow the system down. (This changed with the advent of graphical user interfaces in the 1990s.) The activities required to manufacture and sell personal computers might slow down production, but did not slow down processing inside the machines.

By the mid-1980s, memory technology was widely diffused and many companies were vying to supply fast memory chips at low cost. Memory device makers had also adopted industry-wide standards, thus no single company controlled access to the chips' visible information.

There remained the processor, operating system, buses and chipset. These components were both a source of visible information and on the main path of instructions. The processor set the upper bound for the speed of instruction processing. The operating system's commands were used by many applications, hence their speed of execution could have a noticeable affect on processing time and throughput. Finally, the

buses and chipset "fed" the processor and carried away its output. If processors became more powerful, these auxiliary devices had to keep up.

Thus in the late 1980s and early 1990s, the processor, operating system, buses and chipset were the parts of the platform most likely to be the source of technical and strategic bottlenecks in the future. They constituted the "core of the platform." As we have seen, Intel owned the 80x86 microprocessors and their instruction sets. Microsoft owned the operating system and its APIs. In contrast, the original system bus was unpatented and chipsets consisted of commodity chips widely available at low cost. Nevertheless, because they were on the main path of instructions, the buses and chipset had to increase in throughput and capacity as processors grew more powerful. Not surprisingly, during the 1990s, the buses and chipset were at the center of much strategic jockeying among members of the PC ecosystem. That story is told in Chapter 17.

However, before we can understand these events, we must expand our theory of platforms to deal with platform evolution. Thus far, I have assumed that platform components do not change. However, technologies evolve and improve over time, and platforms are no exception. In the next section, I show how to represent platform evolution in a robust way that permits the analysis of bottlenecks.

16.7 Platform Evolution—Throughput and the Range of Options

In general, the value of a platform can increase in two ways: (1) by increasing throughput and/or (2) by increasing the range of options the platform can support. The two routes to improvement are not mutually exclusive. Quite often, a given option—say a game—will be technically infeasible because the system has insufficient capacity. Thus increasing capacity generally increases the range of options as a matter of course. However, the range of options may also increase as complementors figure out new ways to use existing capacity. For this reason, I will treat the two types of evolution separately but symmetrically.

Increase in Throughput

To fix ideas, let us imagine that improvements in a particular platform component increase the speed at which the platform can process instructions. The impact of the speedup on value would normally differ across the different platform options: some would go much faster, others might not be affected at all. Accounting for this variation, the value of the improved platform system, $V_{P(T)}$, can be written as:

$$V_{P(T)} = P_1^{*X} \cdot P_2^{*} \cdot P_3 \cdot [(1 + a_1) \cdot O_1^{*Y} + \ldots + (1 + a_N) \cdot O_N] \quad . \qquad (7)$$

As before, the P variables indicate the presence/absence of the essential platform components; and the O variables represent the dollar value of each option before speedup. The a variables indicate the percentage increase in the dollar value of each option after speedup. We assume each $a_i \geq 0$, that is, no option is harmed by speedup and some $a_i > 0$, that is some options increase in value after speedup.

We can rewrite equation (7) in the following way:

$$V_{P(T)} = P_1^{*X} \cdot P_2^* \cdot P_3 \cdot (1 + \bar{a}) \cdot \sum_{i=1}^{N} O_i \quad ; \qquad (8)$$

where:

$$\bar{a} \equiv \frac{\sum_1^N a_i \cdot o_i}{\sum_1^N o_i} > 0 \quad \text{(because each } a_i \geq 0 \text{ and some } a_i > 0\text{)} \quad .$$

Increase in the Range of Options

We can model an increase in the range of options in a similar way. Let us imagine that an innovation brings new options into the system. The innovation might arise in one of the platform components: for example, the introduction of Windows™ by Microsoft made possible many new applications with graphical user interfaces. Or it might be spurred by the introduction of a new subsidiary platform with APIs supporting derivative options (see equation 6).

For example, many applications in the iPhone ecosystem invoke functions provided by Facebook, Twitter, YouTube and other apps.[14] Facebook, Twitter and the other subsidiary platforms are themselves options: the phone and its operating system will function in their absence. But they are also "second-level" platforms that increase the options available to other application developers and users. Table 16-2 lists the "Top Twenty" subsidiary platforms for the Apple Iphone, circa 2016.

To model the value increase due to a subsidiary platform, let $V_{P(O)}$ represent the value of the entire platform system with the additional options:

$$V_{P(O)} = P_1^{*X} \cdot P_2^* \cdot P_3 \cdot [O_1^{*Y} + \ldots + O_N + O_{N+1}^{*Z} \cdot (o_1 + \ldots + o_j) \ ] \ . \qquad (9)$$

Old Options New Options

The notation indicates that the new options are made possible by the introduction of a subsidiary platform controlled by agent *Z*. O_{N+1}^{*Z} is a binary variable indicating the presence or absence of the subsidiary platform.

[14] Agarwal and Kapoor (20xx).

Table 16-2 "Top Twenty" Subsidiary Platforms in the Iphone Ecosystem

	Platform Application	No. Dependent Applications
1	Facebook	76845
2	Twitter	60703
3	YouTube	20009
4	Gmail	9851
5	Instagram	8223
6	Dropbox	4872
7	Vine	3025
8	GoogleMaps	2627
9	Uber	2331
10	Yahoo	2093
11	Flashlight	2081
12	Amazon	1980
13	Tumblr	1895
14	Photoeditor	1695
15	WhatsApp	1644
16	Pinterest	1327
17	LinkedIn	1275
18	Chrome	1215
19	Indeed	1202
20	Google	1129

Source: Shiva Agarwal, private communication. The platform applications are ranked according to the number of new applications entering the Apps Store between August 2008 and February 2016 that were linked to these platform applications via APIs.

We can rewrite equation (9) as:

$$V_{P(O)} = P_1^{*X} \cdot P_2^{*} \cdot P_3 \cdot (1 + \overline{b^{*Z}}) \cdot \sum_{i=1}^{N} O_i \qquad (10)$$

where:

$$\overline{b^{*Z}} \equiv \frac{O_{N+1}^{*Z} \cdot \sum_1^j o_i}{\sum_1^N O_i} > 0$$ (because the value of every new option is positive).

The superscript on $\bar{b}$ indicates that Agent Z can lay claim to a portion of the value created by the new options created by its subsidiary platform.

Equations (8) and (10) tell us that platform improvements delivering faster processing or more options increase the value surplus, hence the potential profit stream from the platform system. These performance increases are supermodular complements of the pre-existing number of users (subsumed in the O_i variables), thus indirectly supermodular complements of the number and value of pre-existing options. (More

valuable options attract more users, more users increase the value of investments in platform improvement.)

The question is: Who captures the incremental value/profit from platform improvements? The answer depends on timing and property rights.

Capturing the Value Surplus from Platform Improvements

In the simplest instance, suppose the user buys the basic platform, and then purchases a complement from an independent party. In this case, the value surplus will be split between the user and the complementor, and not shared with the platform sponsor. The basic platform is then "doubly open"—open to external complementors and open to installation of complements without payment to the platform sponsor. *The installed base of the original platform is a "free platform" from the perspective of the complementor.*

The original IBM PC and Wintel computers were "doubly open" in this sense, and even today, many users purchase or download complements directly from their creators.

Alternatively, suppose the original platform sponsor controls a retail channel that gives complementors access to users. For example, Apple controls the iPhone's Apps store, and the vast majority of applications pass through this site. Apple charges a 30% commission on revenue from the sale of apps and "in app" purchases of digital content.[15]

Complementors who do not wish to pay this commission can direct the purchases to a mobile web browser, such as Safari or Chrome. Amazon, for example, requires users to do this, to avoid paying Apple's commission.[16]

Apple also does not charge commissions for purchases of physical goods and services, for example Uber rides or travel bookings. Finally, Apple does not claim a share of any subsidiary platform's ad revenues. However, distributed supermodular complementarity (DSMC) means that the sponsor of a basic platform may benefit from complementors' improvements, even if it does not tax them directly. In the next section, I explain how and why this is so.

16.8 Combining Platform Improvements

What happens when different parties improve the platform on more than one dimension? For example, suppose the original platform sponsor improves the throughput (processing speed) of the basic platform and a complementor introduces a subsidiary platform with many derivative options that are valuable to users. The incremental impact

[15] Padhiyar (2018). For this commission, Apple provides payment clearing services, fraud protection for users, local tax calculation and enforcement, a content distribution network, and digital rights management.

[16] Franco (2018). The terms of the Apple store require sellers to set the same end-user prices, rather than charging more for in-apps purchases.

of each improvement on the value of the platform system is captured by the $\bar{a}$ and $\bar{b}$ parameters in equations (8) and (10) respectively.

Without loss of generality, we can write the increase in the value of the platform system with *both* improvements as follows:

$$V_{P(T+O)} = P_1^{*X} \cdot P_2^* \cdot P_3 \cdot (1 + \overline{a^{*X}} + \overline{b^{*Z}} + c^?) \cdot \sum_{i=1}^{N} O_i \quad . \qquad (11)$$

Here $\overline{a^{*X}}$ represents the independent contribution of higher throughput while $\overline{b^{*Z}}$ represents the independent contribution of the additional options associated with *Z's* unique optional platform. By selling a throughput-enhancing innovation, agent *X* can claim a portion of the value created by this improvement. Similarly, agent *Z* can claim a portion of the value created by its unique subsidiary platform.

The variable *c* in equation (11) is the incremental value created when *both* improvements are present. If the underlying improvements are supermodular complements, then by definition, *c* will be positive. The question is, who can lay claim to the extra value created by *both* improvements?

To answer this question, it is worthwhile to work through a specific example. To begin, let us assume that agent *X* introduces a throughput improvement device first, and agent *Z* then introduces the subsidiary platform. For example, suppose Intel introduces an "accelerator" chip which can be added to existing systems. Thereafter, Facebook introduces its social networking platform, which is downloaded by many of the owners of Intel systems who have already purchased the accelerator. (For now, we will ignore Microsoft's role as the second platform sponsor.)

When Intel introduces the accelerator, I assume it can sell it directly to its installed base at a profit, taking a cut of the incremental value $\overline{a^{*X}} \cdot \sum_{i=1}^{N} O_i$. If Facebook then introduces its subsidiary platform, the combined system will have both higher throughput *and* a broader range of options. Thus Facebook can take a cut of the incremental value $(\overline{\Delta b^{*Z}} + c^{*Z}) \cdot \sum_{i=1}^{N} O_i$. As indicated by the superscript, Facebook, acting unilaterally, can claim a share of the supermodular value *of the two improvements in combination*.

However, the story is not over. Given supermodular complementarity, the existence of Facebook increases the value of the accelerator. Thus, if some of Intel's installed base *did not* purchase the accelerator initially, the presence of Facebook on the platform may induce them to do so. In addition, a faster platform with more options may attract new purchasers. The new users will benefit Intel, Facebook, *and Microsoft*. Microsoft, recall, did not contribute to the platform's improvement, but, its control of a strategic bottleneck means that it will in some cases benefit from improvements created by others.

In summary, a platform sponsor does not need to tax every new option or improvement introduced into the platform system. If the options and improvements are

supermodular complements, their introduction will increase demand for the basic platform and thus enhance the sponsor's profit streams. Finally, the magnitude of these indirect benefits depends on the parameter *c* in equation (11). The higher is *c* in relation to $\bar{a}$ or $\bar{b}$, the more the value of the platform system will depend on *joint and coordinated* action on the part of the platform sponsors at various levels of the platform hierarchy.

What if $\bar{a}$ and $\bar{b}$ are very small or zero?

In the extreme, $\bar{a}$ and $\bar{b}$ may be very small or zero, so that essentially all the value of improvements comes from the *c* term. In these cases, the separate inputs are strong complements and *there is little or no reward to unilateral action*. In that case, the platform value function is not separable, and the first necessary condition for distributed supermodular complementarity (DSMC) is not satisfied. In theory, an open platform system cannot be sustained under these circumstances: a single agent under unified governance is needed to ensure that both improvments take place in a timely way.

This is quite close to the "chicken-and-egg" dilemma which is thought to be endemic to the early stages of platform evolution.[17] A platform with no complements and complements with no platform have no value. There are various ways to solve this dilemma. One way is for a lead agent to create an initial technical architecture, solve the first set of technical bottlenecks, and arrange for a good set of complements to be created *ex ante*, so that users can purchase a fully functional system from Day 1. IBM played this leading role in the introduction of the PC, securing two strategic bottlenecks in the process. (See Chapter 15.)

However a lead system integrator is not always needed to get an ecosystem up and running. What is essential is for the stand-alone rewards, $\bar{a}$ and $\bar{b}$, to be large enough to cover the costs of members of the ecosystem. For example, in the late 1970s, a microcomputer ecosystem without central coordination. (See Chapter 15.) The ecosystem was made up of hobbyists and entrepreneurs united by a shared interest in very small computers. Members of the ecosystem were not linked through a single platform, but instead were connected by diverse bilateral market transactions.[18]

The microcomputer ecosystem was able to evolve as an open system *because the individual products had separable values, hence $\bar{a}$* and *$\bar{b}$ were relatively large*. Every component was sold in the market, with revenues from sales covering the sellers' costs. In this fashion, an open ecosystem containing many competing platforms came into existence without central planning or overt coordination.

[17] Rochet and Tirole (2003); Evans, Hagiu and Schmalensee (2006); Parker, Van Alstyne and Choudary (2016).

[18] Friedrich Hayek (1945) famously argued that changing prices provide a parsimonious mechanism capable of coordinating distributed agents in the absence of central planning. The early microcomputer ecosystem might be called a "Hayekian" ecosystem, while the IBM PC ecosystem might be called a "sponsored" ecosystem.

16.9 Platform Disintermediation

The natural result of actions by many autonomous agents in an open platform system is a complex network of technological artifacts linked by varying degrees of technical dependency and supermodular complementarity. In this network, certain key nodes will emerge as strategic bottlenecks at different levels of the platform hierarchy.

Strategic bottlenecks are positions of power in an open platform system. Complementors dependent on a strategic bottleneck component must split their profits with the bottleneck owner. Thus they have good reason to remove the bottleneck or its owner if they can. This section deals with strategies of platform *distintermediation*, defined as the purposeful removal of a strategic bottleneck.

In the value structure equations above, let us focus on the leading platform component, P_1^{*X}, which is unique and owned by *X*. There are basically four ways to disintermediate this platform component:

(1) Replace the *entire* platform system with another technology (substitution);
(2) Remove *X* (the owner) via reverse engineering;
(3) Remove the * (uniqueness of the component) by designing options to be "platform independent;" or
(4) Remove P_1 (the platform module) by "enveloping" its functionality into another platform component.

Below, I give examples of each of these methods of platform disintermediation.

Platform Substitution

It is conceptually simple, but often practically difficult, to replace one platform with another technology that provides the same goods and services. The most common instances of disintermediation involve transaction platforms. On a transaction platform, once a buyer and seller are matched, they may conduct their further dealings directly, without going through the platform or another intermediary.[19]

Replacing a standards-based platform, once it is established, is more difficult. As we have seen, standards become embedded in the designs of complements in the form of instructions. Creating a new platform means starting over with new instructions and a corresponding loss of backward compatibility and legacy assets. The new platform must create enough new value to compensate users for these losses.

For example, when the original IBM PC was introduced, three operating systems were advertised as PC-compatible. (See Chapter 15.). However, two were not ready to ship on the launch date. A few months later, users had become accustomed to using MS-DOS and developers had embedded DOS commands in their application code. After only six months, switching to a different operating system was unthinkable for many users.

[19] Gu and Zhu (2018).

(The lock-in to DOS continued for decades. When Microsoft introduced Windows™, the company made sure that the new operating system was fully backward compatible with older versions of DOS.)

Reverse Engineering

The second way to disintermediate a standards-based platform is to use the same instructions, but give them many owners. In Chapter 15, I described the legal practice of "clean-room reverse engineering," which Compaq and others used to replicate the BIOS instructions of the IBM PC. After reverse engineering, the instructions (inputs) were the same and the resulting behavior (outputs) were the same, but the code used to instantiate the instructions did not violate IBM's copyright. In other words, the visible information (standards) continued to be unique, but ceased to be owned by the original copyright holder: $P_1^{*X} \rightarrow P_1^*$.

The main impediments to reverse engineering are (1) patents; and (2) complexity. Patents give the patentee rights to a family of designs that solve a technical problem in a specific way: they protect the *idea* behind the solution. In contrast, copyrights do not protect ideas, but only the way the idea is expressed. Patented designs cannot be legally reverse engineered unless the new solution achieves the stated goal in a demonstrably different way from the original. Thus patents offer a broader form of intellectual property protection than copyrights. However, software patents are not recognized in every jurisdiction, and are generally difficult to obtain.

Reverse engineering also requires the exhaustive testing and cataloging of cause and effect within a system that must be treated as a black box. When the target of reverse engineering is complex—for example, a large codebase—the various paths through the system grow exponentially increasing the cost and reducing the reliability of the reverse engineering effort.

"Platform Independent" Complements and Emulation

Complementors must decide whether to design their products for one platform or several.[20] The impact of platform independence is subtly different from reverse engineering. The targeted platform component becomes "non-unique," as well as "not owned:" $P_1^{*X} \rightarrow P_1$. However, only the complement whose owner undertakes the expense of becoming platform independent is affected by the change.

Notwithstanding this fact, if an alternate platform attracts enough complementors *and users*, complementors aspiring to dominate a particular niche may have no choice but to design products for both platforms. In that case, the first platform will cease to be a strategic bottleneck.

[20] Single-platform design is sometimes called "single-homing" and multi-platform design "multi-homing."

A variant on this strategy occurs when a consortium of complementors collaborate to support a common alternative platform that is controlled by a not-for-profit organization.[21] The consortium-backed standards are generally made available for free or at a low cost, placing even more pressure on the incumbent platform sponsor. However, as we will see in later chapters, the sponsor may encourage industry members and/or open source communities to take charge of some essential platform components, while continuing to make other components the basis of its strategic bottleneck.

A technical method of achieving platform independence is through "cross-platform emulation." Here instructions from one machine or program are translated into instructions understood by a different machine or program. In contrast to reverse engineering, the objective of emulation is not to replicate the behavior of a given platform component, but to provide a different pathway for the execution of instructions. For example, the reverse-engineered IBM BIOS was meant to send instructions to Intel microprocessors. An emulator would allow programs written for IBM-compatible PCs to run on Apple hardware.

The main drawback to emulation is poor performance, i.e., the translated code generally runs more slowly than the original code.

Platform Envelopment

The last method of platform disintermediation is an operation known as "platform envelopment," first analyzed by Thomas Eisenmann, Geoffrey Parker and Marshall Van Alstyne.[22] The move is often initiated by an incumbent platform as a defense against the establishment of a second strategic bottleneck and/or the threat of disintermediation.

Envelopment occurs when the owner of one platform component integrates the functionality of another platform component into its own technical architecture such that the components cannot be separated. The components become one module (with one price), instead of two.[23] The enveloper then generally sets the price of the combined components at a steep discount to what users would pay for the two original components. It may also withdraw its earlier product from the market, offering only an integrated product. Users are then confronted with the choice between (1) purchasing the fully functional integrated product only vs. (2) purchasing the integrated product plus a redundant, stand-alone product from the second vendor.

Microsoft famously used this tactic to counter the inroads made by Netscape's Navigator, one of the first Internet browsers. The so-called "browser wars" that ensued are discussed in the next chapter. There we will also see examples of "partial

[21] Consortium-backed standards are generally known as "industry standards" or "open standards."

[22] Eisenmann, Parker and Van Alstyne (2011).

[23] In antitrust economics and law, this practice is known as "bundling."

envelopment" where the enveloper continues to offer stand-alone products as well as the integrated product.

16.10 Conclusion—How Technology Shapes Organizations

Through the magic of complementarity, open platforms and ecosystems are capable of creating a great deal more value than can be gleaned from their separate pieces. The question then arises: which agents and what types of organizations are most likely to capture larger shares of the system's surplus value?

In this chapter, I combined the theories of technical and strategic bottlenecks and flow production in order to:

- Identify the most likely points of value capture in an open platform system; and
- Predict how a platform system might evolve in terms of modular structure, property rights, and the zone of authority of the platform sponsor.

I first focused on the strategic bottlenecks in the platform and among the options. Platforms typically consist of many different components, all of which are essential to the platform's proper functioning. Strategic bottlenecks are platform components that are unique and owned by a for-profit enterprise. The owner (or owners) of such platform components will split the system-level surplus with the owners of the options provided on the platform. The amount of the surplus and the fraction claimed by each owner depend on the details of the product market and the owners' property rights.

However, within any platform there exists a sequence of steps the platform must perform. In particular, digital platforms must carry out a series of instructions arranged in the form of programs. The speed and throughput of instruction processing are key determinants of the platform's value in the eyes of users.

Speeding up throughput generally requires integrating steps and reducing buffers and distance on the "main path" of instructions. Sponsors must ensure that their property rights and zone of authority allow them to modify the "core of the platform" to make the entire system run faster.

Platform improvements include not only actions that increase throughput, but also actions that increase the range of options, perhaps by creating a subsidiary platform with its own unique instructions. By packaging improvements in separate modules, innovators can generally claim a share of the incremental value they create.

However, supermodular complementarity among innovations means that a given innovation will increase the value of others' investments, rebounding to the benefit of other platform sponsors and/or complementors. These indirect benefits make the split of value among users, platform sponsors, and complementors messy and indeterminate. They also mean that platform sponsors do not have to "tax" every possible profit stream.

Indeed, other things equal, the untaxed profit streams will garner more investment by third parties, attracting more users, and increasing all profit streams.

Platform disintermediation is the purposeful removal of a strategic bottleneck at any level of a platform system. The most common methods of platform disintermediation are: (1) replacing one platform with another (substitution); (2) reverse engineering visible information so that correct instructions can be supplied by many agents without violating the creator's property rights; (3) making optional complements "platform independent" so that a given platform component is no longer unique; and (4) "enveloping" a platform within a larger platform.

The next chapter considers how Intel and Microsoft managed to solve emerging technical bottlenecks in the Wintel platform while protecting their respective strategic bottlenecks.

References

Adner, R. and Kapoor, R. (2016) Innovation ecosystems and the pace of substitution: Re-examining technology S-curves. *Strategic Management Journal*, 37(4):625-648.

Arora, Ashish and Robert P. Merges (2004) "Specialized Supply Firms, Property Rights and Firm Boundaries," *Industrial and Corporate Change*, 13(3):451-475.

Asanovic, K., Bodik, R., Catanzaro, B. C., Gebis, J. J., Husbands, P., Keutzer, K., ... & Yelick, K. A. (2006). *The landscape of parallel computing research: A view from Berkeley* (Vol. 2). Technical Report UCB/EECS-2006-183, EECS Department, University of California, Berkeley.

Boudreau, K. J., & Jeppesen, L. B. (2015). Unpaid crowd complementors: The platform network effect mirage. *Strategic Management Journal*, *36*(12), 1761-1777.

Boudreau, K.J., 2018. *Amateurs Crowds & Professional Entrepreneurs as Platform Complementors* (No. w24512). National Bureau of Economic Research.

Eisenmann, T., Parker, G., & Van Alstyne, M. (2011). Platform envelopment. *Strategic Management Journal*, *32*(12), 1270-1285.

Ethiraj, Sendil K. (2007) "Allocation of Inventive Effort in Complex Product Systems," *Strategic Management Journal*, 28(6):563-584.

Evans, D. S., Hagiu, A., & Schmalensee, R. (2006). Invisible engines: how software platforms drive innovation and transform industries. Cambridge: MIT Press.

Franco, J. (2018) Why You Can't Buy Books from the Kindle app on iPhone or iPad in 2018, *Techspot.com* (April 5, 2018) https://www.techspot.com/article/1597-how-to-buy-kindle-book-iphone-ipad/ (viewed 10/26/19).

Gu, G., & Zhu, F. (2018). Trust and disintermediation: Evidence from an online freelance marketplace. *Harvard Business School Technology & Operations Mgt. Unit Working Paper*, (18-103).

Gross, T. R., Jouppi, N. P., Hennessy, J. L., Przybylski, S., & Rowen, C. (2016). A Retrospective on "MIPS: A Microprocessor Architecture". *IEEE Computer Society*, *36*(4), 73-76.

Hannah, D. P., & Eisenhardt, K. M. (2018). How firms navigate cooperation and competition in nascent ecosystems. *Strategic Management Journal*, *39*(12), 3163-3192.

Hart, Oliver and John Moore (1990) "Property Rights and the Nature of the Firm," *Journal of Political Economy,* 98(6):1119-1158.

Hayek, Friedrich A. (1945) "The Use of Knowledge in Society," *American Economic Review* 35(4):519-530.

Hennessy, John L. and David A. Patterson (1990) *Computer Architecture: A Quantitative Approach,* San Mateo, CA: Morgan Kaufmann.

Hughes, T. P. (1993). Networks of power: electrification in Western society, 1880-1930. Johns Hopkins University Press.

Jacobides, Michael G., Thorbjorn Knudsen and Mie Augier (2006) "Benefiting from Innovation: Value Creation, Value Appropriation and the Role of Industry Architecture," *Research Policy,* 35(8):1200-1221.

Kapoor, R. (2018). Ecosystems: broadening the locus of value creation. *Journal of Organization Design*, *7*(1), 12.

Kapoor, R., & Agarwal, S. (2017). Sustaining superior performance in business ecosystems: Evidence from application software developers in the iOS and Android smartphone ecosystems. *Organization Science*, *28*(3), 531-551.

Mead, Carver and Lynn Conway (1980) *Introduction to VLSI Systems*, Addison-Wesley, Reading, MA.

Padhiyar, J. (2018) Why Apple's App Store is Charging 30% Fees and How is it Justified? *IGeeksBlog.com* (August 23, 1018) https://www.igeeksblog.com/why-app-store-is-charging-30-percent-commission/ (viewed 10/26/19)

Parker, G., & Van Alstyne, M. (2017). Innovation, openness, and platform control. *Management Science*, *64*(7), 3015-3032.

Parker, G.G., Van Alstyne, M.W. and Choudary, S.P. (2016) *Platform revolution: How networked markets are transforming the economy--and how to make them work for you*. WW Norton & Company.

Patterson, David A. and John L. Hennessy (1994) *Computer Organization and Design: The Hardware/Software Interface,* San Mateo, CA: Morgan Kaufmann.

Rochet, Jean-Charles and Jean Tirole (2003) "Platform Competition in Two-sided Markets," *Journal of the European Economic Association*, 1(4): 990-1029.

Rosenberg, N. (1969). The direction of technological change: inducement mechanisms and focusing devices. *Economic development and cultural change*, *18*(1), 1-24.

Rosenberg, N. (1976). *Perspectives on technology*. Cambridge University Press.

Teece, D. J. (2018). Profiting from innovation in the digital economy: Enabling technologies, standards, and licensing models in the wi

Teece, David J. (1986).“Profiting from Technological Innovation: Implications for Integration, Collaboration, Licensing and Public Policy,” *Research Policy*, 15(6): 285-305.

Von Hippel, E., 2016. *Free innovation*. Cambridge, MA: MIT press.