MIS Quarterly

SPECIAL ISSUE: IT AND INNOVATION

PLATFORM ECOSYSTEMS: HOW DEVELOPERS INVERT THE FIRM[1]

Geoffrey Parker
Thayer School of Engineering, Dartmouth College, 14 Engineering Drive,
Hanover, NH 03755 U.S.A. {geoffrey.g.parker@dartmouth.edu}

Marshall Van Alstyne
Questrom School of Business, Boston University, 595 Commonwealth Avenue,
Boston, MA 02215 U.S.A. {mva@bu.edu}

Xiaoyue Jiang
School of Business & Engineering, Quinnipiac University, 275 Mount Carmel Avenue,
Hamden, CT 06518 U.S.A. {xiaoyue.jiang@quinnipiac.edu}

For a period starting in 2015, Apple, Google, and Microsoft became the most valuable companies in the world. Each was marked by an external developer ecosystem. Anecdotally, at least, developers matter. Using a formal model of code spillovers, we show how a rising number of developers can invert the firm. That is, firms will choose to innovate using open external contracts in preference to closed vertical integration. The locus of value creation moves from inside the firm to outside. Distinct from physical goods, digital goods afford firms the chance to optimize spillovers. Further, firms that pursue high risk innovations with more developers can be more profitable than firms that pursue low risk innovations with fewer developers. More developers give platform firms more chances at success. Our contribution is to show why developers might cause a shift in organizational form and to provide a theory of how platform firms optimize their own intellectual property regimes in order to maximize growth. We use stylized facts from multiple platform firms to illustrate our theory and results.

Keywords: Open innovation, sequential innovation, platforms, R&D spillovers, intellectual property, network effects, network externalities, bundling, two-sided networks, two-sided markets, vertical integration, standard setting organizations, platform

Introduction and Background

One of the most important questions a firm can ask is how to efficiently create value: Should it produce its own output or should it orchestrate the output of others? In the case of code, the choice increasingly favors orchestration over production. Apple, Google, and Microsoft, for example, became the three most valuable companies in the world in 2015, having passed energy and investment firms Exxon-Mobile and Berkshire-Hathaway.[2]

[1]Satish Nambisan, Kalle Lyytinen, Ann Majchrzak, and Michael Song were the accepting senior editors for this paper. Kalle Lyytinen served as the associate editor.

The appendix for this paper is located in the "Online Supplements" section of the *MIS Quarterly*'s website (http://www.misq.org).

[2]http://finance.google.com, accessed November 9, 2015. Supplemented with Thomson Reuters data.

We argue that developer communities are inverting the firm. That is, firms must now manage value creation that occurs externally just as carefully as they manage the value they create internally. And, this is not just outsourcing. Firms are relinquishing product specifications to third parties that they do not even know. We provide a formal theory of why this is happening and ask what levers platform firms have to encourage external developers to innovate on their behalf. Our particular focus is on digital innovations that developers create to extend platforms. Digital platforms differ from physical systems such as automobiles because the subsystem boundaries can be more loosely defined, which makes recombination of elements less costly, and because information is non-rival. Reusable code, for example, facilitates knowledge spillovers. Interestingly, however, even firms that produce product platforms such as automobiles, tractors, and turbines are adding a digital layer to create an Internet of Things (Evans and Annunziata 2012). The Open Automotive Alliance's mission statement provides an example of the trend toward digital innovation on top of physical products: "The members of the Open Automotive Alliance share a vision for the connected car, and are committed to collaborating around a common platform to make this vision a reality."[3]

The role of developers has become so central in digital ecosystems that firms have developed strategies for "platform evangelism" to manage third party contributions that have become central to platform success.[4] Reasons that developers are so important in digital platforms include well known features of digital technology such as malleability of the code, the low cost of investing in tools to develop code, close to zero cost reproduction, and the potential to profit from application successes while shedding the costs of failures. Put more broadly, developers are key to a platform's ability to scale rapidly because what the platform firm does is not limited by the processes of hiring, training, project selection, and coordination. Instead, these processes are distributed outside of the platform, allowing much more rapid growth (Parker et al. 2016).

Others have observed the importance of digital ecosystems and the strategies that firms have to seek advantage. In an agenda setting paper for the field, Yoo et al. (2010) note that firms need to ask what they should open and what they should close in a digital product platform. Kallinikos et al. (2013) adopt the term "digital artifact" and characterize digital objects as open, reprogrammable, and accessible by other digital objects.

We are far from alone in observing that an historic shift is underway, driven by rapid improvements in network connectivity and computing power (Brynjolfsson and McAfee 2014). Earlier improvements in transportation technology changed the locus of economic activity from vertically integrated firms to firms organized around a nexus of supplier networks. We believe that the current shift goes farther still. Using loosely affiliated ecosystems, firms are able to harness a global network of partners they have never met. These partners can connect through digital networks to innovate on top of a platform's core set of resources, thereby creating highly valuable products and services for ecosystem users.

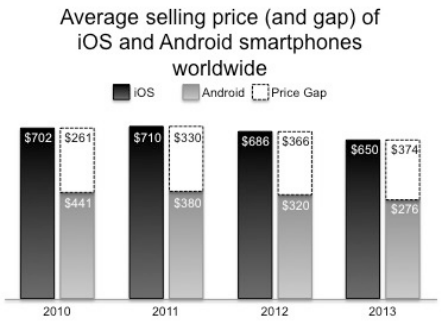
In our context, a relevant innovation is any digital application, including a product or service, that is produced by an ecosystem partner using core platform resources. The rate of innovation is the rate at which developers produce such goods and services. A platform is a "layered architecture of digital technology" (Yoo et al. 2010, p. 725), combined with a governance model (Parker and Van Alstyne 2014 Tiwana 2013). Architecture includes a device layer, network layer, service layer and content layer. Governance includes rules or participation and rewards that organize the ecosystem. The platform owner then influences developer innovation by exposing more platform resources, for example, by opening the architecture (APIs, SDKs, code libraries, templates, etc.), and by offering more favorable standard licensing agreements (SLAs), such as by offering exclusivity for new apps.

The battle between Apple iOS and Google Android provides a useful example of different platform choices around layered architecture and openness. Although Apple courts third party developers, it remains a relatively closed system along the dimensions catalogued in Eisenmann et al. (2009). For example, developers who wish to publish iOS applications must submit to Apple's rigorous quality review and Apple controls the only distribution channel. Eaton et al. (2015) show that tensions have arisen among Apple, its developers who build on top of iOS, and its end-users, many of whom have "bricked" their devices in order to gain access to apps outside Apple's control. In contrast, Google has released Android under an open source license and has attracted more developer attention even though it launched after iOS.[5] Google's Android system is open to hardware manufacturers to the point that Google published reference designs to reduce their fixed costs in creating Android handsets (McAllister 2011). To retain modest control over Android, Google makes API access dependent on a Google Play subscription (Amadeo 2013). Parker and Van Alstyne (2014) describe

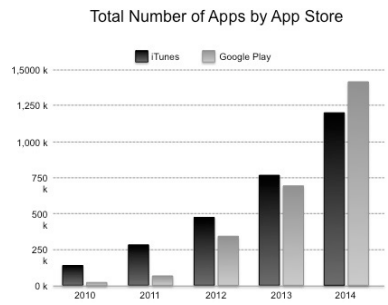
[3]http://www.openautoalliance.net/, accessed November 9, 2015.

[4]https://en.wikipedia.org/wiki/Platform_evangelism.

[5]http://readwrite.com/2014/01/08/app-store-sales-google-play-android.



(a) The more open system has the lower price. Source: Statista using IDC data.



(b) The more open source grows faster. Source: AppFigures

Figure 1. Openness Entails a Tradeoff (it nets less revenue on the core platform but fosters more applications development)

the strategic challenges such firms face when competing as platform ecosystems.

These different decisions have led to different market outcomes for both the price each platform charges and the number of applications provided by developers. As Figure 1a shows, the closed Apple platform is able to charge higher prices for the core platform, giving Apple much higher margins. By contrast, Figure 1b shows that the more open Android platform has attracted more applications development. The number of Android apps passed those of Apple iOS in 2014. Openness therefore entails a tradeoff. It nets less revenue on the core platform but fosters more applications development.

Based upon our interviews with the global heads of platforms at Cisco, Haier, SAP, and Thomson Reuters, we examine two key decisions that platform managers must make. These are (1) how much of the core platform to open in order to spur developer innovation and (2) how long to grant developers the right to benefit from sales on top of the platform before the platform absorbs those innovations into the core. We explore how these decisions are affected by competition, level of vertical integration, number of developers, and the risk of innovation failure.

This paper addresses key topics of this special issue. In particular, we adopt the view that IT platforms can reshape innovation ecosystems. When the number of developers is sufficiently high, a platform with a default contract provides a new architecture for digital innovation. Decisions about how long to protect developer innovations, and especially how much to open the platform are at the intersection of IT architecture, organizational design, and innovation and inform critical decisions about how firms should create value.

Literature

The need to open a platform to facilitate developer innovation is nicely stated by Laursen and Salter (2014): "in order to obtain knowledge, organizations have to reveal some parts of their own knowledge to external actors" (p. 868). In effect, this argues for the benefit of knowledge spillovers. Attaching to and building on others' ideas is easier conditional on access to these ideas. Importantly, in the case of software, Eilhard and Ménière (2009) provide empirical evidence that open code produces significant knowledge spillovers. In a study of 10,553 open source projects on SourceForge, they find evidence of nondecreasing returns to scale. Productivity of developers rises substantially with access to code libraries, especially if modules use the same language. The decision to fold new private code back into an existing public code library thus represents a design parameter of a healthy ecosystem.

Unsurprisingly, given the complexity of platform markets, executives have disagreed over how to manage their developer ecosystems (Libert et al. 2014). At the closed end of the spectrum, TiVo used provisions of the Digital Millennium Copyright Act to lock out industry players who sought to attach to its proprietary systems (Slater and Schoen 2006). This allowed TiVo to charge more for its innovations, yet closure kept its ecosystem small. Similarly, the iPhone was completely closed when Apple introduced it in 2007. Not until hackers broke into it in order to add new features did Apple release an approved System Development Kit (SDK) (Eaton et al. 2015). At the open end of the spectrum, UNIX firms have lost control over application programming interfaces (APIs) to committees and have reduced pricing power as a result (West 2003). RedHat, for example, uses standard GNU Public License terms that give anyone who receives

their code the right to modify and distribute copies of the enhanced code for free. This fosters ecosystem participation, but competitors promptly absorb all valuable innovations.

Despite considerable research on prices, quantities, and network effects, Yoo et al. (2010) note that little formal analysis has investigated broader platform business models. In addition, Tiwana (2013) calls for research on how platform governance affects ecosystem innovation. A growing literature has focused on leadership (Gawer and Cusumano 2008), economics (Bresnahan and Greenstein 1999; Farrell et al. 1998), launch (Bhargava et al. 2013), and strategies for managing platforms (Boudreau 2010; Cusumano 2010). Markovich and Moenius (2009) analyze competitive platform dynamics and show that weak developers can benefit from value added by strong developers. Zhu and Iansiti (2012) show how a platform entrant can overcome an incumbent based on the strength of developers' indirect network effects. Huang et al. (2013) show that developers with stronger property rights can more successfully resist expropriation by the platform. Scholten and Scholten (2011) identify control points that allow the platform sponsor to charge for access. The two-sided literature conceives of platforms as mediating markets with network externalities that cross distinct user groups and shows how subsidies to one group become optimal (Caillaud and Jullien 2003; Eisenmann et al. 2006; Parker and Van Alstyne 2000a, 2000b, 2005; Rochet and Tirole 2003; Rysman 2009).

To build the ecosystem, platform sponsors often embrace modular technologies and encourage partners to supply downstream complements in competitive markets (Baldwin and Clark 2000; Boudreau 2010; Fine 1999). Loose integration promotes layered industries. In the personal computer industry, for example, these layers consist of semiconductor manufacturing, PC assembly, operating system, and application software, among others (Baldwin and Clark 2000; Grove 1996; Shapiro and Varian 1999). The credit card and telecommunications industries are similarly layered (Evans et al. 2006).

Ondrus et al. (2015) study platform openness with a focus on how the openness decisions at the firm level, the technology level, and the user level affect the overall market potential. The key issue they study is the likelihood a platform achieves critical mass. In our study, we approach the openness decision from the opposite direction and focus on how external competition, technical risk, size of developer base, and network effects change the decisions around openness. We also examine the optimal exclusionary period platforms should offer developers in order to foster innovation. Analysis proceeds in the novel context of recursive innovation where developers reuse code from one period to the next.

Extending prior literature, we show the following three results:

(1) Firms with small developer bases and minimal network effects prefer vertical integration to open innovation. Firms with large developer bases or substantial network effects prefer the opposite, and shift their choice of organizational form.

(2) Optimal exclusionary period for developers' intellectual property falls as the number of developers rises and their chances of product success rise. Optimal openness increases in the number of developers until a threshold number of developers after which the platform reduces openness. Openness increases in the chances of product success.

(3) Platform-to-platform competition strictly increases the level of openness as a function of the number of developers. Developer-to-developer competition can increase or decrease optimal openness depending on a threshold total number of developers.

We develop each of these results and discuss their meaning below.

Model with Developer Competition

We extend Parker and Van Alstyne (2017) by explicitly incorporating N_d, the number of developers in an analytical framework, and analyzing developer-to-developer competition as well as platform-to-platform competition. Such an extension enables analysis of how the key platform decisions of exclusionary period t and openness σ are affected by technical risk, pricing power, and network effects. It also permits analysis of how spillovers across developers can change organizational form.

Ecosystem participants include platform sponsors, developers, and consumers. Table A1 in the appendix summarizes variable definitions. V is the value of the core platform before developers add applications. This is what a user would pay for a standalone platform. For example, when the Apple iPhone first shipped, it had network connectivity, an e-mail reader, a web browser, a calendar app, and a handful of basic applications. In 2007, it was closed and did not offer third party applications until 2008 (Eaton et al. 2015).

To capture sequential innovation, let time span two periods of equal length t. At time zero, a platform sponsor makes fraction σ of its platform's value publicly available to

developers, representing free access to reference diagrams, code libraries, APIs, and SDKs. The other $(1 - \sigma)$ portion remains private and is held by the platform sponsor. As a motivating example, consider the "Price Gap" in Figure 1a. This value, represented by public code σV, is not captured by the platform owner. Instead, the platform owner captures only the residual private code $(1 - \sigma)V$. By contrast, a fully closed system would capture V and thus have higher baseline profit.

Developers add value according to a standard Cobb Douglas production function with technology parameter α. We denote the output of an individual developer in each period as y_1 and y_2, respectively. Period one output is $y_1 = k(\sigma V)^{\alpha}$ and period two output is $y_2 = k(y_1)^{\alpha} = k^{1+\alpha}(\sigma V)^{\alpha^2}$. Assume code reuse $k > 0$ and concave technology $0 < \alpha < 1$. Thus second period production depends recursively on first period production. If developer code stays closed, other developers cannot build upon it, there are no spillovers, and third party production has no value. If developer code goes open, a choice the platform sets via the exclusionary period in its contract, then developers can build on each others' code in the next period. Reuse permits code spillovers. Although the production function stays constant and concave, the effect of reuse rises from k to kk^{α} and the effect of technology strengthens from $(\sigma V)^{\alpha}$ to $((\sigma V)^{\alpha})^{\alpha}$. This formulation has the attractive property of capturing knowledge spillovers such as those found in Eilhard and Ménière (2009). Their use of the translog production function for econometric analysis is a direct generalization of the Cobb Douglas form used here.

If code is open, it is public and easily copied. No one can charge for public code. If code is closed, it is private and developers can charge for what they produce. The platform cannot tax developers who have no revenues. Openness thus has a benefit of increasing code spillovers and boosting innovation but at a cost of decreasing ability to charge. To balance these conflicting interests, the platform sets an exclusionary or non-compete period t during which developers can charge for their code. Like a patent, however, the platform will make the code public after the non-compete period expires. We have direct evidence that the non-compete period is a strategic variable whose duration companies choose. For example, SAP publishes an 18 to 24 month roadmap that articulates "white space," alerting developers that they will not copy their innovations until after this time. After that, any developer innovation may be absorbed into the SAP core. Apple reserves the right to appropriate developer innovations and reuse them in its ecosystem.[6] Similarly, Cisco bundles features that have appeared among multiple developer products into its core network operating system where they can be accessed via API calls by ecosystem partners. "Developers don't like it but realize it's good for the ecosystem."[7]

At the time developer innovations become open, the competitive price falls to zero. Knowing apps will be free in the future, a strategic customer optimizes between consuming the innovation at $p = v$ or waiting until time t, which at a conventional interest rate implies a discount of $\delta = e^{-rt}$. The option to wait thus means that a customer will only pay for the incremental value of immediate consumption v relative to the discounted value of future consumption δV. The indifferent consumer thus accepts a price no higher than $p = v - \delta v = (1 - v)\delta$. For simplicity, we assume market size is 1, and that a developer charges the highest price possible to the indifferent consumer, who then enjoys surplus δV.

Deviating from the original setup in Parker and Van Alstyne (2017), we introduce a new parameter, $N_d \geq 1$, to represent the number of developers who compete with one another. This will interact with the production technology and the choice of openness to drive spillovers. The platform sponsor sells the platform, gives away open code to seed development, and splits profits on new development across both periods according to the Nash bargaining solution. With only one developer, platform sponsor profit can then be written as

$$\begin{aligned}\pi_p &= V - \sigma V + \tfrac{1}{2} p y_1 + \tfrac{1}{2}\delta p y_2 \\ &= V(1-\sigma) + \tfrac{1}{2} v(1-\delta) k(\sigma V)^{\alpha} + \tfrac{1}{2}\delta v(1-\delta) k^{1+\alpha}(\sigma V)^{\alpha^2}\end{aligned} \quad (1)$$

Increasing the number of developers N_d raises output in period one such that $\tilde{y}_1 = N_d y_1$. Recursive production then yields $\tilde{y}_2 = N_d^{1+\alpha} y_2$, where the additional $(N_d)^{\alpha}$ follows from production spillovers $y_2 = ((N_d)^{\alpha} y_1)$ of period one developers. Additional developers, however, reduce the pricing power of any given developer in the manner of Cournot competition. The exact formula for price under Cournot competition is $\tilde{p} = \frac{1}{N+1} p$.[8] To make analysis more tractable, simply interpret N_d as $N - 1$, representing the number of *other* developers beyond the first one. This allows us to use the simpler form $\tilde{p} = \frac{1}{N_d} p$. Substituting parameters, we have

[6] "Apple works with many application...developers and some of their products may be similar to or compete with your applications. Apple may also be developing its own...competing applications...or may decide to do so in the future....Apple will be free to use and disclose any licensee [code] on an unrestricted basis without notifying or compensating you," Section 10.3 of the standard iOS license (https://developer.apple.com/programs/terms/ios/standard/ios_program_standard_agreement_20140909.pdf; accessed November 9, 2015.

[7] Authors' interview with Guido Jouret, CTO, Emerging Markets Group, Cisco Systems Inc., September 8, 2006.

[8] See, for example, Tirole (1988, p. 220).

$$\begin{aligned}\pi_p &= V - \sigma V + \tfrac{1}{2}\widetilde{p}\widetilde{y}_1 + \tfrac{1}{2}\delta\widetilde{p}\widetilde{y}_2 \\ &= V(1-\sigma) + \tfrac{1}{2}\tfrac{1}{N_d}pN_d y_1 + \tfrac{\delta}{2}\tfrac{1}{N_d}p(N_d)^{\alpha+1}y_2 \\ &= V(1-\sigma) + \tfrac{1}{2}v(1-\delta)y_1 + \tfrac{\delta}{2}v(1-\delta)(N_d)^{\alpha}y_2 \\ &= V(1-\sigma) + \tfrac{1}{2}v(1-\delta)k(\sigma V)^{\alpha} + \tfrac{\delta}{2}v(1-\delta)k^{1+\alpha}(N_d)^{\alpha}(\sigma V)^{\alpha^2}\end{aligned} \quad (2)$$

which reduces to Eq. (1) when $N_d = 1$.

Model Analysis

To facilitate the analysis below, we introduce a generic parameter $\beta = (N_d)$ and rewrite platform sponsor profit as

$$\pi_p = V(1-\sigma) + \tfrac{1}{2}v(1-\delta)k(\sigma V)^{\alpha} + \beta\delta\tfrac{1}{2}v(1-\delta)k^{1+\alpha}(\sigma V)\alpha^2 \quad (3)$$

Conceptually, we can interpret β as a measure of the spillover effect. We will later see in this and the next section that platforms exposed to developer technical risk and platforms subject to network effects would experience different levels of spillover, say β_{TR}, or β_{NE}. Nonetheless, the dependency of the platform's choice of σ and δ on each β is directionally the same.

Optimization under Competition and Developer Risk

Consider the situation where individual developers face the risk of technical failure (with probability ρ). Given failure in earlier rounds, the number of developers in the ecosystem may decline over time. Assume the number of developers in Stage 1 is N_d and denote the number of developers who survive to Stage 2 as n. n follows a binomial distribution with parameter N_d and ρ. The mean of the total value generated in Stage 1 remains $y_1 = (\sigma V)^{\alpha}$, and in Stage 2, it becomes $E(n^{\alpha})y_2$.

Consequently, the corresponding platform profit can be expressed in terms of the generic form Eq. (3) with $\beta_{TR}(N_d, \rho) = E(n^{\alpha})$. For the special case $\rho = 0$, $n = N_d$, and $\beta_{TR}(N_d, \rho) = \beta_D = (N_d)^{\alpha}$. For the general case $0 < \rho < 1$, it is clear that $0 < \beta_{TR}(N_d, \rho) < \beta_D$, and it monotonically increases in N_d and in $\omega = 1 - \rho$. Denoting $N_r := [E(n^{\alpha})]^{1/\alpha}$, we have $\beta_{TR} = (N_r)^{\alpha}$, and the mean value of platform profit under technical risk can be written as

$$\pi_p = V(1-\sigma) + \tfrac{1}{2}v(1-\delta)k(\sigma V)^{\alpha} + \beta_{TR}(N_d, \rho)\delta\tfrac{1}{2}v(1-\delta)k^{1+\alpha}(\sigma V)^{\alpha^2} \quad (4)$$

$$= V(1-\sigma) + \tfrac{1}{2}v(1-\delta)k(\sigma V)^{\alpha} + (N_r)^{\alpha}\delta\tfrac{1}{2}v(1-\delta)k^{1+\alpha}(\sigma V)^{\alpha^2} \quad (5)$$

In other words, the mean profit of a risk-prone platform is identical to that of a no-risk platform Eq. (1) with a reduced number of developers N_r instead of N_d. Further, define $R := (\alpha V)/(2N_r)$, $U := N_r k/V^{1-\alpha}$, and $\overline{\delta} := [1 - \sqrt{\alpha/(2-\alpha)}\,]/2 < 1/2$. We have the following characterization of the optimal interior solution (σ^*, δ^*). Denote $\overline{N_d}$ as the unique N_r that induces optimal σ at $\overline{\delta}$.

Proposition 1 *Given k, v, V, $N_d > 0$, $0 < \alpha < 1$, $0 \le \rho < 1$. Under condition R, $U < 1$, the optimum (σ^*, δ^*) of the risk-prone platform corresponds to the optimum in the no-risk platform with a reduced number of developers $N_r < N_d$. Consequently, the following results hold true:*

(i) *An interior optimum $\delta^* \in (0, \overline{\delta})$ uniquely exists.*
(ii) *An interior optimum $0 < \sigma^* < 1$ uniquely exists.*
(iii) *Optimum δ^* increases monotonically in N_r. Consequently, it also increases in N_d and in $\omega = 1 - \rho$.*
(iv) *σ monotonically increases in δ when $\delta \le \overline{\delta}$ and in $N_r(N_d, \omega)$ when $N_r \le \overline{N_d}$; and σ monotonically decreases in δ when $\delta \ge \overline{\delta}$ and in $N_r(N_d, \omega)$* when $N_r \ge \overline{N_d}$.

Proof. Please see the appendix ■

To interpret these results, we first note that $N_r < N_d$ suggests that risk damps the spillover effect. Reduced spillovers then shorten the exclusionary period as developers have less to build upon. Conversely, an increased success rate encourages the platform to promote spillovers by shortening the exclusionary period. Note that enlisting additional developers could compensate for technical risk by increasing spillovers. The result that having more developers helps mitigate risk is consistent with both logic and empirical research that finds handheld device platforms opened to more developers precisely to reduce the risk of technological innovation (Boudreau 2010). For the same reason, social network platforms encourage developers to experiment because "much remains unknown concerning preferences and technical approaches to social applications" (Boudreau and Hagiu 2009, p. 11).

The finding that α first increases in N_d and then decreases after a threshold level has an interesting interpretation. At first, the spillover effect that dominates at N_d rises so that the platform opens. Past that threshold, the competition effect dominates so that the platform closes. This also has an important antecedent in the literature. Laursen and Salter (2006, 2014) find a concave relationship between appropriability and openness. To join their context and ours, additional developers can be interpreted as increasing collaboration. An interesting case arises when there is friction in adjusting σ. In this situation, the firm might anticipate growth in the developer base at launch and set a non-optimal σ and then hope that

platform adoption would make the choice better over time. This parallels platform launch as described in Ondrus et al. (2015).

Platform Competition

We now analyze competition among platforms. Continuing with the Green and Scotchmer (1995) approach, we let competition moderate platform pricing power in the same way that it moderates developer pricing power, reducing platform price from V to V/N_p where $N_p \geq 1$ is the number of platform competitors. Note from the proof of Proposition 1 that optimal δ^* is independent of V, but that optimal σ^* depends on σV. We conclude that increasing N_p implies a linear dependence $N_p\sigma := \sigma_p^*$ until σ_p^* hits 1, complete openness, meaning the platform sponsor would give away 100% of its original value. We summarize this discussion formally.

Corollary 1 *Increasing the intensity of platform competition has no effect on t^*, but proportionally increases σ^* to* $\min(N_p\sigma^*, 1)$.

Proof. *The claims follow from Proposition 1 by substituting V/N_p for V.* ■

Holding all else constant, greater platform competition reduces the sponsor's direct platform profits. The sponsor's incentive is, therefore, to open the platform in order to increase indirect profits from developer innovation. In terms of competition policy, the social planner should promote platform competition, which motivates sponsors to open their platforms and seek growth. This result directly parallels empirical findings. Based on case studies of IBM, Sun Microsystems, and Apple, West (2003) concluded that sponsors prefer the higher rents from closing their systems unless their platforms face pressure from rival platforms.

Permissionless Versus Negotiated Platform Access

To this point, our analysis has assumed a platform with default contracts that allow developers to innovate on top. In this setting, it is always strategically optimal to open (i.e., set $\sigma > 0$), at least to some degree. However, a more fundamental question remains to be answered: is opening the platform to all developers the best way to organize for innovation? Might not vertical integration, which implies closing developer access and only working with negotiated partnerships, be better? Vertical integration is a well-known solution to challenges of ecosystem innovaton (Adner and Kapoor 2010).

The two-stage platform model suggests that one mechanism, a network spillover effect, could help open innovation to gain competitive advantage over closed/ negotiated access. In fact, the spillover effect β induced by multiple developers N_d scales up profit in the second stage. Below, we examine how a network effect scales up the ecosystem by mobilizing both developers and users. The generic form of Eq. (3) facilitates analysis via the spillover factor β_{NE} that is implied by a network effect.

Negotiated Platform Access

Numerous mergers and acquisitions are predicated on the theory that a rational firm could improve profits by buying developers to acquire their technology. By only allowing negotiated platform access in our model, the sponsor might gain three advantages over open or permissionless innovation. First, closing the platform saves the open innovation subsidy σV. This increases profits from direct platform sales. Second, negotiated development can build on the *entire* platform, not just the portion opened, so output rises from $y(\sigma V)$ to $y(V)$. Third, application prices rise to monopoly levels $p = v$ because users cannot acquire apps by waiting for them to become an open public good. Thus app profits also rise. Allowing developers to keep half the value of their technology based on Nash bargaining, or simply acquiring them, the platform's profit under vertical integration rises to $\pi_{vi} = V(1-\sigma)|_{\sigma=1} + y_2|_{\sigma=1}$, which simplifies to

$$\pi_{vi} = V + \tfrac{1}{2}vkV^{\alpha} + \tfrac{1}{2}\delta vk^{1+\alpha}V^{\alpha^2} \quad (6)$$

The corresponding platform system with $N_d = 1$ is listed below for comparison.

$$\pi_p = V(1-\sigma) + \tfrac{1}{2}v(1-\delta)k(\sigma V)^{\alpha} + \delta\tfrac{1}{2}v(1-\delta)k^{1+\alpha}(\sigma V)^{\alpha^2} \quad (7)$$

Apparently, vertical integration, Eq. (6), yields higher profit than an open platform with one developer, Eq. (7). It has higher output; it has no subsidy cost; and it has higher prices.[9] We then ask how might profits from open innovation *ever* dominate those from vertical integration? Following Eq. (3) and examining the effects of spillovers β, denoted as $\pi_p(\beta)$, we conclude

Proposition 2 *Vertical integration outperforms open innovation when there is no spillover which reduces the spill-over multiplier to $\beta = 1$. However, there is a unique*

[9]Model analysis can easily extend to subcontracting, an organizational form between vertical integration and open innovation, by choosing different levels of σ.

breakeven spillover parameter $\bar{\beta}$ such that with all other parameters remaining constant, the open platform system outperforms vertical integration if and only if $\beta \geq \bar{\beta}$.

Proof. Notice that the profit in the second stage is the only item depending on β. More specifically, $\pi_p(\beta)$ increases proportionally in β. Thus, increasing the spillover effect β improves the profit of open system $\pi_p(\beta)$, which ultimately outperforms the vertically integrated system π_{vi}. ■

As a practical matter, we posit two distinct reasons why open innovation frequently dominates vertical integration based on "permissionless innovation" (Cerf 2012). One is that there exist developers the sponsor does not know and therefore cannot acquire. The other is that network effects can increase disproportionately under openness. The former might arise if there are numerous small developers who participate if they see an opportunity. Permissionless innovation matters to developers who risk disclosing their novel ideas by identifying themselves or their applications to the platform sponsor. Owning the indispensable asset, the sponsor has bargaining power and needs only the ideas to steal them (Bessen and Maskin 2009; Parker and Van Alstyne 2000a, 2012). Commitment to stay out of the developer's market during the exclusionary period provides the incentive such developers need to step forward. The law literature (Eisenberg 1976) notes that making such a commitment will affect the downstream conduct of other parties whenever the mere act of negotiating reveals sensitive information. This result is clearly in evidence in SAP's platform, for example, where, as noted above, the platform sponsor commits to stay out of "white spaces," functionality that anyone is free to develop, for a minimum period of 18 to 24 months.

The second answer arises because, relative to closed systems, open systems invite more third party participation. Mechanisms by which openness might increase participation include transparency, bug reporting and feedback that can reduce R&D costs and increase platform quality, and user ability to modify open systems (Chesbrough 2003; West 2003). Openness can reduce negotiation costs, facilitate free redistribution (Raymond 1999), and serve as a low price commitment analogous to second sourcing (Farrell and Gallini 1988). It can aid horizontal integration (Farrell et al. 1998). The "two-sided" network literature (Parker and Van Alstyne 2000a, 2005; Rochet and Tirole 2003) specifically demonstrates how openly subsidizing one community (i.e., developers) can increase value to and participation of another community (i.e., end-users). For a variety of reasons, openness can increase both value and participation.

As both answers rely on growing the platform microeconomy, we now modify the earlier open platform model to include classic two-sided network effects across consumers and developers who value one another's participation on the platform (e.g., Parker and Van Alstyne 2005). For tractability, we develop a novel yet simplified version of two-sided network effects to understand how their strength affects a sponsor's choice to provide access to all developers versus working with a select few. Thus we introduce market multiplier, M_i, $i \in (u, d)$, derived from two-sided market feedback, in order to represent the sizes of spillover externalities from content creation and content consumption.

Network Effect

While advantages of vertical integration include eliminating the subsidy, increasing prices, and increasing output, the advantage of open innovation is growing the market. Higher adoption and network effects can then justify open innovation relative to vertical integration. To understand how network effects drive innovation, consider the following mechanism that allows more users to attract more developers and more developers to attract more users. Let an externality spillover e_{ud} attract new developers in proportion to the number of users N_u, increasing baseline developers N_d by $e_{ud}N_u$. Likewise, let an externality spillover e_{du} attract new users in proportion to the number of developers N_d, increasing baseline users N_u by $e_{du}N_d$. These new users attract additional new developers, and vice versa, in amounts $e_{du}e_{ud}N_u$ and $e_{ud}e_{du}N_d$, respectively, a recursion process that defines Cauchy sequences for both groups. Developer size increases according to $N_d(1 + e_{du}e_{ud} + (e_{du}e_{ud})^2 + (e_{du}e_{ud})^3 + \ldots)$ and similarly for users. To keep market size finite, impose the constraint $e_{du}e_{ud} < 1$. These sequences converge to $N_d M_d = N_d \frac{1}{1-e_{ud}e_{du}}$ and $N_u M_u = N_u \frac{e_{du}}{1-e_{ud}e_{du}}$, respectively.

We can now present the platform profit function increased by network effects as follows:

$$\pi_{open} = M_u\left(V(1-\sigma) + \tfrac{1}{2}v(1-\delta)k(\sigma V)^{\alpha} + \tfrac{1}{2}v\delta(1-\delta)(N_d M_d)^{\alpha} k^{1+\alpha}(\sigma V)^{a^2}\right) \quad (8)$$

Introducing network effects has two consequences: (1) on the developer side, the spillover effect increases to the level of $\beta_{NE} := (N_d M_d)^{\alpha}$, (2) on the user side, total platform profit is rescaled proportional to growth in the user population M_u. These two effects jointly cause profits from open innovation to dominate those from negotiated access/vertical integration. The following results parallel those of Proposition 2.

Proposition 3 *For a platform ecosystem with network effects, there exists a monotonically decreasing threshold* $\overline{NM_d}(M_u)$ *such that for any given user-side multiplier* M_u*, the open platform outperforms vertical integration if and only if* $N_d M_d \geq \overline{NM_d}(M_u)$.

Proof. By observing that π_{open}/M_u in Eq. (8) is the standard π_p in Eq. (3) with $\beta = (N_d M_d)^{\alpha}$, the proof follows Proposition 2 for all $M_u > 0$. ■

In the absence of network effects, the platform sponsor should own or contract for all means of production. In fact, negotiated access/vertical integration becomes more attractive as platform value itself grows. This is consistent with a supply side economy of scale in V. By contrast, opening the platform to outside developers becomes more attractive as (1) network effects rise (or the sizes of user or developer pools grow), (2) developer output rises, and (3) content becomes more reusable. Note that the decentralized innovation is achieved without bargaining costs. A default contract with ($\sigma > 0$, $t > 0$) gives developers an option to enter the market without disclosing their ideas to the platform sponsor. Open innovation, with a guarantee of lead-time, preserves the information asymmetry that protects the innovator and prevents a powerful monopoly platform from stealing the full value of the innovation. The importance of third party contributions also becomes clearer as we observe that price effects, which are one-time gains, yield lower returns than production effects, which are recursive gains.

Discussion and Conclusions

Using a formal model of sequential innovation with code spillovers, our contribution is to show why developers might cause a shift in organizational form and to provide a theory of how platform firms optimize their own intellectual property regimes in order to maximize growth. This extends the sequential innovation literature (Chang 1995; Green and Scotchmer 1995; Parker and Van Alstyne 2017) to add developer competition, spillovers, and network effects. From this baseline, we add three new results on optimal openness (σ), optimal IP duration (t), and effects of competition among developers (N_d).

1. We show how a rising number of developers can invert the firm. That is, firms will choose to innovate using open external contracts in preference to closed vertical integration or subcontracts. The locus of value creation moves from inside the firm to outside. Distinct from physical goods, digital goods afford firms the chance to optimize spillovers. If the number of developers is small or network effects are modest, firms prefer vertical integration or closed subcontracts. However, once a threshold developer base N_d is reached, firms prefer to offer an open default contract to any developer who wishes to build upon the platform. Distinct from traditional buyer–supplier networks, this attracts resources from third parties that the platform firm does not even know.

2. Competition has different implications depending upon whether it occurs between platforms or among developers. Platform-to-platform competition strictly increases the level of openness. Because the platform makes less direct profit, openness costs less and the firm prefers to subsidize developer spillovers. By contrast, developer-to-developer competition has a non-monotonic effect. Openness first rises in the number of developers to promote R&D spillovers but then falls due to developer price competition. The result that openness has an invert-U relationship to innovation is consistent with empirical literature (Boudreau 2010; Laursen and Salter 2014).

3. Firms that pursue high risk innovations with more developers can be more profitable than firms that pursue low risk innovations with fewer developers. More developers give platform firms more chances at success. Platforms will increase their openness (σ) as the likelihood of technical success increases until a threshold level of $\bar{N}_d$ after which the platform reduces openness. Further, the more developers, the shorter is the proprietary period (t) offered under an optimal IP policy. The reason is that more developers increase the value of spillovers. Thus it makes sense to subsidize N-1 other developers by making the code of a given developer public in addition to the open code of the platform itself.

Limitations and Research Implications

The model analyzed above requires a number of assumptions for tractability. Key among these are (1) a shared consumer value for the platform V and developer additions, v, (2) use of a Cobb-Douglas formulation for developer output, and (3) no developer entry between periods one and two. A shared value is clearly a simplification since consumer valuations are more likely to follow an exponential distribution. However, Bakos and Brynjolfsson (1998) observe that, based on the central limit theorem, the average value converges rapidly for any bounded distribution as the population of consumers grows or the number of items in a bundle grows.

Analysis should be robust to changing the point-mass valuation because the focus of the analysis is on the behavior of the platform with respect to technical risk, the number of developers, and network effects. If the markets were made up of heterogeneous consumers, then some consumers would be priced out and the served market would be smaller. It would be interesting to examine the impact of the distribution of consumer values on the choices that platforms make. We leave that analysis to future researchers who might need to use simulation tools given the complexity of the analysis.

The Cobb-Douglas output assumption was also made in order that the model be solvable using formal analytic techniques. However, the formulation is widely used and, importantly, enjoys empirical support in work by Eilhard and Ménière (2009), who found the translog generalization fits the data for open source projects in real applications.

Our model of technical risk assumes that developers who fail are assumed to exit. If developers were to enter between period one and two, the impact on the platform's choices of σ and t should be the same as if the number of developers were simply larger in period one. However, the issue of when and why developers join platforms merits further study.

Generally, we note that results in the sequential innovation literature have not, until now, begun to account for recursive R&D spillovers that are feasible with digital goods. By not accounting for digital reuse, prior literature has recommended a patent period that is too long (Chang 1995; Gilbert and Shapiro 1990; Green and Scotchmer 1995). Further, the prior works do not account for the variety of possible subsequent uses, which conditions and shortens these baseline predictions. Each of the results derived from the analysis presented above can also be formulated as a testable hypothesis, which we hope to explore in subsequent empirical research.

Managerial and Policy Implications

There are several managerial and policy conclusions. First, consider a classic mergers and acquisition policy that would recommend acquiring complementary assets. We show that this can be suboptimal in a platform context. More precisely, we show that permissionless innovation can dominate vertical integration in cases where the number of developers becomes large because openness promotes R&D spillovers, which do not occur when the firm internalizes all production. Moreover, the platform owner does not always know which developers will succeed in the market and therefore which assets to acquire. This result implies that a platform strategy has a longer term likelihood of success than a purchasing/subcontracting strategy so long as the developer base reaches a sufficient size. This inverts the firm as it moves production outside. The platform can wait longer to observe and profit from external developers before (if ever) acquiring them.

Second, as noted after Proposition 1, suppose that changing the level of openness is not frictionless. This might be the case due to technological or cultural constraints. Then firms prefer to set openness higher than is initially optimal whenever they anticipate growth of the developer base. Firms can also restrict network growth beyond a given size in order to control for poor developers.

Third, the core platform value can become so great that the owner no longer needs to give it away to stimulate growth. In that case the owner prefers to monetize the platform by reasserting control and further limiting openness. We can observe this in practice based on two examples. MakerBot, a 3D printing company, that gave away designs as well as design specs, was more open at first but has since moved to a more closed strategy by pursuing patents and creating applications that are not governed by open source licenses (Brown 2012). Similarly, Android was more open at first in order to foster growth but has since moved to close the platform by exerting control over application programming interfaces (APIs) and critical applications such as Google maps (Amadeo 2013).

Fourth, the manager's role must change in a platform context. Rather than optimizing the profits of the firm's own product in isolation, the manager needs to optimize the value created by an ecosystem, even if it cannibalizes a core product. In the context of large numbers of developers, this "platform" model is more profitable in the long run. In a real world context, considering developer profit is consistent with the policies of SalesForce and SAP as these firms specifically measure themselves by how much value they create for ecosystem partners. Indeed, addressing the cannibalization problem via a platform strategy is a significant topic addressed in the books *Platform Leadership* (Gawer and Cusumano 2002) and *Platform Revolution* (Parker et al. 2016). Here we show this result formally and prove that it can be more profitable.

Fifth, managing the ecosystem can also be interpreted as setting the optimal growth policy for an intellectual property (IP) regime where code may be reused. This is a unique property of digital and information goods. The period (t) during which a developer can sell an app parallels the duration of patent protection. The end of this period parallels when developer code becomes "public" and useful for R&D spillovers that help other developers. How the reuse of digital code affects optimal IP policy, and in turn is affected by competition, is not analyzed in prior literature.

References

Adner, R., and Kapoor, R. 2010. "Value Creation in Innovation Ecosystems: How the Structure of Technological Interdependence affects Firm Performance in New Technology Generations," *Strategic Management Journal* (31:3), pp. 305-333.

Amadeo, R. 2013. "Google's Iron Grip on Android: Controlling Open Source by Any Means Necessary," ars technica, October 20 (http://arstechnica.com/gadgets/2013/10/googles-iron-grip-on-android-controlling-open-source-by-any-means-necessary/).

Bakos, Y., and Brynjolfsson, E. 1998. "Bundling Information Goods: Pricing, Profits and Efficiency," *Management Science* (45:12), pp. 1613-1630.

Baldwin, C., and Clark, K. 2000. *Design Rules: The Power of Modularity*, Volume 1, Cambridge, MA: The MIT Press.

Bessen, J., and Maskin, E. 2009. "Sequential Innovation, Patents, and Imitation," *The RAND Journal of Economics* (40:4), pp. 611-635.

Bhargava, H., Kim, B., and Sun, D. 2013. "Commercialization of Platform Technologies: Launch Timing and Versioning Strategy," *Production and Operations Research* (22:6), pp. 1374-1388.

Boudreau, K. 2010. "Open Platform Strategies and Innovation: Granting Access versus Devolving Control," *Management Science* (56:10), pp. 1849-1872.

Boudreau, K., and Hagiu, A. 2009. *Platforms, Markets and Innovation*, Cheltenham, UK: Edward Elgar Publishing Limited.

Bresnahan, T., and Greenstein, S. 1999. "Technological Competition and the Structure of the Computer Industry," *The Journal of Industrial Economics* (47:1), pp. 1-40.

Brown, R. 2012. "Pulling Back from Open Source Hardware, MakerBot Angers Some Adherents," c|net, September 27 (http://www.cnet.com/news/pulling-back-from-open-source-hardware-markerbot-angers-come-adherents/).

Brynjolfsson, E., and McAfee, A. 2014. *The Second Machine Age: Work, Progress, and Prosperity in a Time of Brilliant Technologies*, New York: W. W. Norton & Company.

Caillaud, B., and Jullien, B. 2003. "Chicken & Egg: Competition Among Intermediation Service Providers," *RAND Journal of Economics* (34:2), pp. 309-328.

Cerf, V. 2012. "Remarks at the Digital Broadband Migration: The Dynamics of Disruptive Innovation: Internet Speculations," *Journal of Telecommunications and High Technology Law* (10), pp. 21-31.

Chang, H. 1995. "Patent Scope, Antitrust Policy, and Cumulative Innovation," *The RAND Journal of Economics* (26:1), pp. 34-57.

Chesbrough, H. 2003. *Open Innovation: The New Imperative for Creating and Profiting from Technology*, Boston: Harvard Business Press.

Cusumano, M. 2010. "Technology Strategy and Management: The Evolution of Platform Thinking," *Communications of the ACM* (53:1), pp. 32-34.

Eaton, B., Elaluf-Calderwood, S., Sørensen, C., and Yoo, Y. 2015. "Distributed Tuning of Boundary Resources: The Case of Apple's iOS Service System," *MIS Quarterly* (39:1), pp. 217-244.

Eilhard, J., and Ménière, Y. 2009. "A Look Inside the Forge: Developer Productivity and Spillovers in Open Source Projects" (http://ssrn.com/abstract=1316772).

Eisenberg, M. 1976. "Private Ordering Through Negotiation: Dispute-Settlement and Rule- Making," *Harvard Law Review* (89), pp. 637-681.

Eisenmann, T., Parker, G., and Van Alstyne, M. 2006. "Strategies for Two-Sided Markets," *Harvard Business Review* (84:10), pp. 92-101.

Eisenmann, T., Parker, G., and Van Alstyne, M. 2009. *Opening Patforms: How, When and Why?*, Cheltenham, UK: Edward Elgar Publishing Limited.

Evans, D., Hagiu, A., and Schmalensee, R. 2006. *Invisible Engines: How Software Platforms Drive Innovation and Transform Industries*, Cambridge, MA: The MIT Press.

Evans, P. C., and Annunziata, M. 2012. "Industrial Internet: Pushing the Boundaries of Minds and Machine," General Electric.

Farrell, J., and Gallini, N. 1988. "Second-Sourcing as a Commitment: Monopoly Incentives to Attract Competition," *The Quarterly Journal of Economics* (103:4), pp. 673-694.

Farrell, J., Monroe, H., and Saloner, G. 1998. "The Vertical Organization of Industry: Systems Competition Versus Component Competition," *Journal of Economics & Management Strategy* (7:2), pp. 143-182.

Fine, C. 1999. *Clockspeed: Winning Industry Control in the Age of Temporary Advantage*, New York: Basic Books.

Gawer, A., and Cusumano, M. 2002. *Platform Leadership: How Intel, Microsoft, and Cisco Drive Industry Innovation*, Boston: Harvard Business Press.

Gawer, A., and Cusumano, M. 2008. "How Companies Become Platform Leaders," *MIT Sloan management review* (49:2), pp. 28-35.

Gilbert, R., and Shapiro, C. 1990. "Optimal Patent Length and Breadth," *The RAND Journal of Economics* (21:1), pp. 106-112.

Green, J., and Scotchmer, S. 1995. "On the Division of Profit in Sequential Innovation," *The RAND Journal of Economics* (26:1), pp. 20-33.

Grove, A. 1996. *Only the Paranoid Survive*, New York: Currency Doubleday.

Huang, P., Ceccagnoli, M., Forman, C., and Wu, D. 2013. "Appropriability Mechanisms and the Platform Partnership Decision: Evidence from Enterprise Software," *Management Science* (59:1), pp. 102-121.

Kallinikos, J., Aaltonen, A., and Marton, A. 2013. "The Ambivalent Ontology of Digital Artifacts," *MIS Quarterly* (37:2), pp. 357-370.

Laursen, K., and Salter, A. 2006. "Open for Innovation: The Role of Openness in Explaining Innovation Performance Among UK Manufacturing Firms," *Strategic Management Journal* (27:2), pp. 131-150.

Laursen, K., and Salter, A. J. 2014. "The Paradox of Openness: Appropriability, External Search and Collaboration," *Research Policy* (43:5), pp. 867-878.

Libert, B., Wind, Y. J., and Fenley, M. B. 2014. "What Airbnb, Uber, and Alibaba Have in Common," *Harvard Business Review*

(https://hbr.org/2014/11/what-airbnb-uber-and-alibaba-have-in-common).

Markovich, S., and Moenius, J. 2009. "Winning While Losing: Competition Dynamics in the Presence of Indirect Network Effects," *International Journal of Industrial Organization* (27:3), pp. 346-357.

McAllister, N. 2011. "Google Empowers Hardware Hackers with the Android ADK," InfoWorld, May 12 (http://www.infoworld.com/article/2621810/mobile-development/google-empowers-hardware-hackers-with-the-android-adk.html).

Ondrus, J., Gannamaneni, A., and Lyytinen, K. 2015. "The Impact of Openness on the Market Potential of Multi-Sided Platforms: A Case Study of Mobile Payment Platforms," *Journal of Information Technology* (30), pp. 260-275.

Parker, G., and Van Alstyne, M. 2000a. "Information Complements, Substitutes, and Strategic Product Design," in *Proceedings of the 21st International Conference on Information Systems*, Association for Information Systems, pp. 13-15.

Parker, G., and Van Alstyne, M. 2000b. "Internetwork Externalities and Free Information Goods," in *Proceedings of the Second ACM Conference on Electronic Commerce*, New York: Association for Computing Machinerym oo, 197-116.

Parker, G., and Van Alstyne, M. 2005. "Two-Sided Network Effects: A Theory of Information Product Design," *Management Science* (51:10), pp. 1494-1504.

Parker, G., and Van Alstyne, M. 2012. "A Digital Postal Platform: Definitions and a Roadmap," Technical Report, MIT Center for Digital Business.

Parker, G., and Van Alstyne, M. 2014. "Platform Strategy," *Palgrave Encyclopedia of Strategic Management*, M. Augier and D. Teece (eds.), London: Palgrave MacMillan.

Parker, G., and Van Alstyne, M. W. 2017. "Innovation, Openness, and Platform Control," *Management Science* (forthcoming).

Parker, G., Van Alstyne, M., and Choudary, S. 2016. *Platform Revolution*, New York: W. W. Norton & Company.

Raymond, E. 1999. "The Cathedral and the Bazaar," *Knowledge, Technology & Policy* (12:3), pp. 23-49.

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

Rysman, M. 2009. "The Economics of Two-Sided Markets," *The Journal of Economic Perspectives* (23:3), pp. 125-143.

Scholten, S., and Scholten, U. 2011. "Platform-Based Innovation Management: Directing External Innovational Efforts in Platform Ecosystems," Technical Report, SAP Research and Karlsruhe Institute of Technology.

Shapiro, C., and Varian, H. 1999. *Information Rules: A Strategic Guide to the Network Economy*, Boston: Harvard Business Press.

Slater, D., and Schoen, S. 2006. "Who Killed TiVo to Go?," Technical Report, Electronic Frontier Foundation (w2.eff.org/IP/pnp/cablewp.php).

Tirole, J. 1988. *The Theory of Industrial Organization*, Cambridge, MA: MIT Press.

Tiwana, A. 2013. *Platform Ecosystems: Aligning Architecture, Governance, and Strategy*, Waltham, MA: Morgan Kaufmann.

West, J. 2003. "How Open Is Open Enough? Melding Proprietary and Open Source Platform Strategies," *Research Policy* (32:7), pp. 1259-1285.

Yoo, Y., Henfridsson, O., and Lyytinen, K. 2010. "The New Organizing Logic of Digital Innovation: An Agenda for Information Systems Research," *Information Systems Research* (21:4), pp. 724-735.

Zhu, F., and Iansiti, M. 2012. "Entry into Platform-Based Markets," *Strategic Management Journal* (33:1), pp. 88-106.

About the Authors

Geoffrey Parker is a professor of engineering at Dartmouth College. Prior to his appointment at Dartmouth, he was a professor of management science at Tulane University. He is also a visiting scholar and research fellow at the MIT Initiative for the Digital Economy. He has made significant contributions to the economics of network effects as codeveloper of the theory of two-sided networks. Geoffrey's work has been supported by the Department of Energy, the National Science Foundation, and numerous corporations. He advises senior leaders in government and business and is a frequent speaker at conferences and industry events. He received his BS from Princeton and his MS and Ph.D. from MIT.

Marshall W. Van Alstyne is a professor of information systems at Boston University and a visiting scholar and research fellow at the MIT Initiative on the Digital Economy. Marshall has made fundamental contributions to IT productivity and to theories of network effects. In addition, he holds patents in information privacy protection and on spam prevention methods. Marshall has been honored with six best paper awards and National Science Foundation IOC, SGER, iCORPS, SBIR, and Career Awards. He is an adviser to leading executives, a frequent keynote speaker, a former entrepreneur, and a consultant to startups and to Global 100 companies. He received his BA from Yale and his MS and Ph.D. from MIT.

Xiaoyue Jiang is an assistant professor at Quinnipiac University. His research has focused on reliability and maintenance, software reliability, telecommunications performance engineering, supply-chain management with quality of service guarantees, power system modeling and optimization. He has made fundamental methodological contributions to network calculus and envelope modeling. He received his BS from Renmin University of China, and MSC from the Chinese Academy of Science, and M.Eng. and Ph.D. degrees from the University of Toronto.