Received: 28 February 2018 | Revised: 28 March 2020 | Accepted: 16 April 2020 | Published on: 17 July 2020

DOI: 10.1002/smj.3191

SPECIAL ISSUE ARTICLE

STRATEGIC MANAGEMENT JOURNAL WILEY

Platform governance matters: How platform gatekeeping affects knowledge sharing among complementors

Yuchen Zhang[1] | **Jingjing Li**[2] | **Tony W. Tong**[3]

[1]Freeman School of Business, Tulane University, New Orleans, Louisiana

[2]McIntire School of Commerce, University of Virginia, Charlottesville, Virginia

[3]Leeds School of Business, University of Colorado Boulder, Boulder, Colorado

Correspondence
Tony W. Tong, Leeds School of Business, University of Colorado Boulder, 995 Regent Drive, Boulder, CO 80309.
Email: tony.tong@colorado.edu

Abstract

Research Summary: Orchestrating complementors' value creation activities is critical to platform owners but is challenging. Emerging literature on platform governance suggests that platform access control can shape complementors' contributions to platforms. We extend this literature by using the coopetition framework from strategic management to examine the relationship between platform gatekeeping, a prominent policy for governing platform access, and knowledge sharing among complementors. Exploiting the iOS 7 jailbreak as an exogenous shock to Apple's gatekeeping policy and tracing iOS and Android app developers' knowledge sharing activity on an online forum, we find causal evidence that a lapse in gatekeeping reduces knowledge sharing among iOS app developers. Further, this effect is mitigated among developers with greater knowledge complexity but magnified among those with greater knowledge routineness.

Managerial Summary: Platform owners can use governance policies, such as gatekeeping, to control complementors' platform access and shape their value creation activities. This study examines how platform access control affects the interactions among complementors in the form of knowledge sharing. We find that iOS app developers share knowledge less frequently after a lapse in Apple's gatekeeping policy, suggesting that strict platform access control can facilitate complementors' interactions with one another. Further, the decrease in knowledge sharing is more pronounced among app developers with routine knowledge and less so among those with complex knowledge, indicating that complementors' characteristics also matter. This study highlights the value of understanding the impact of the design and deployment of platform governance policies on complementors' interactions and value creation activities more generally.

KEYWORDS

complementor, gatekeeping, innovation, knowledge sharing, platform governance

1 | INTRODUCTION

Complementors' value creation activities are critical to the success of platforms (Jacobides, Cennamo, & Gawer, 2018; McIntyre & Srinivasan, 2017). Such activities, ranging from developing innovative complements to exchanging product or market information, can increase the attractiveness of a platform (Adner & Kapoor, 2010; Gawer, 2011). For example, Apple's App Store and Google Play are more attractive to mobile device users than Windows Phone Store because millions of active developers publish a wide variety of mobile apps on those two platforms. However, orchestrating complementors' value creation activities is challenging to platform owners for at least two reasons. First, complementors have heterogeneous incentives to contribute to platforms (Boudreau & Jeppesen, 2015), making it difficult for platform owners to coordinate with complementors. For example, complementors have both competitive incentives to secure proprietary knowledge for creating unique complements, and cooperative incentives to share knowledge for collectively attracting potential users (Wareham, Fox, & Giner, 2014). Second, conventional means, such as hierarchical control, are less applicable to the management of such incentives in a platform context, because complementors are often not under the platform owner's direct control (Adner, 2017).

A burgeoning stream of research revolves around the idea of orchestrating complementors' value creation activities through platform governance policies (Constantinides, Henfridsson, & Parker, 2018). This stream suggests that the design and deployment of platform governance rules, including hard regulations and soft nudges, have substantial influences on complementors' actions and decisions (Claussen, Kretschmer, & Mayrhofer, 2013; Kretschmer & Claussen, 2016; Rietveld, Schilling, & Bellavitis, 2019). Within this stream, platform access control receives a great deal of attention (Boudreau, 2010). Platform gatekeeping, a prominent policy for governing platform access—defined as a set of predetermined acceptance criteria that control what (complement) or who (complementor) is allowed on a platform (Tiwana, 2013), is suggested to play a significant role in shaping complementors' value creation activities. However, studies to date have often focused on how complementors' interactions with platforms, such as release of new software applications on a platform, are in response to platform governance rules, neglecting how complementors' interactions among themselves may also be shaped by platform access control such as gatekeeping.

We aim to address this gap by focusing on the impact of platform gatekeeping on one important type of such interactions among complementors, their knowledge sharing activities, using the coopetition framework from strategic management. Knowledge sharing, according to the coopetition literature (Hoffmann, Lavie, Reuer, & Shipilov, 2018; Khanna, 1998; Khanna, Gulati, & Nohria, 1998), is driven fundamentally by agents' competitive and cooperative incentives. In the platform context, for example, when complementors are less burdened by competitive pressure, they are more likely to cooperatively exchange knowledge, such as users' needs or preferences, which increases their contributions to the platform as a whole (Wareham et al., 2014). Complementors' knowledge sharing is especially relevant to innovation platforms on which complementors create and release innovative products. Knowledge sharing can uncover new interdependencies among knowledge embedded in different complementors, triggering novel knowledge recombination, empowering complementors' innovation attempts, and subsequently enhancing the platforms' attractiveness and value (Boudreau, 2010). Hence, examining the implications of a prominent platform governance policy (i.e., platform gatekeeping) on an important type of interaction among complementors (i.e., knowledge sharing) can extend our understanding of how platform governance shapes complementors' competitive and cooperative incentives and value creation for the platform.

Building on recent applications of the coopetition framework to the ecosystem context (Hannah & Eisenhardt, 2018), we argue that knowledge sharing among complementors is affected by the balance between their focus on common benefits (i.e., expanding platforms' overall user base) and private benefits (i.e., increasing their respective market share). By managing platform access, platform gatekeeping may affect complementors' competitive pressure and concerns regarding private benefits, thus shifting complementors' coopetitive balance (Khanna, 1998; Khanna et al., 1998). This shift in coopetitive balance may change complementors' knowledge sharing as a result. Moreover, we suggest that the magnitude of this change may hinge on complementors' characteristics. Extant research highlights the competitive implications of two characteristics of complementors: knowledge complexity and knowledge routineness (Fleming, 2001; Jehn, 1995; Rivkin, 2001; Wong, 2004). A complementor's knowledge complexity characterizes her mastery of knowledge of heavily interdependent subcomponents, and a complementor's knowledge routineness indicates her reliance on frequently used knowledge. We argue that complementors with high knowledge complexity are concerned less about competitive pressure, whereas those with high knowledge routineness are concerned more about it. As a result, these differences may lead to differential effects of gatekeeping on complementors' knowledge sharing.

Research suggests that like other governance policies, platform gatekeeping evolves (Claussen et al., 2013), varies with several shades of gray, and may change accidentally or even illicitly against platform owners' desires (Jacobides et al., 2018). This idea suggests a promising way to identify the effects of gatekeeping on knowledge sharing. Specifically, we focus on an exogenous lapse of platform gatekeeping, an externally caused weakening in gatekeeping that grants platform access to complementors who are otherwise rejected. We predict that such lapse will tilt incumbent complementors' coopetitive balance toward competition and reduce knowledge sharing; further, the extent of the reduction in knowledge sharing will vary across complementors with different degrees of knowledge complexity and routineness.

To test these predictions empirically, we examine changes in iOS app developers' knowledge sharing with each other after an exogenous lapse in Apple's gatekeeping policy—the jailbreak of iOS 7. The iOS platform is well-known for its strict app review processes that deny platform access to imitating apps and developers in order to encourage app innovation (Koetsier, 2017). The jailbreak of iOS and the subsequent introduction of pirated apps that closely imitated or directly copied the features and functionalities of existing apps triggered so-called business stealing and user attrition, exerting significant competitive pressure on incumbent app developers (Koetsier, 2017; Lahiri & Dey, 2013; Rasch & Wenzel, 2013). Hence, the jailbreak of iOS serves as an appropriate empirical context for our purpose. Knowledge sharing among app developers is difficult to observe, track, and measure, despite it being so central to software development (von Krogh & von Hippel, 2006). To address this challenge, we use detailed user activity logs from StackOverflow.com, an active knowledge-sharing forum for app developers worldwide (Abdalkareem, Shihab, & Rilling, 2017). To capture the causal effects of the jailbreak on knowledge sharing among iOS app developers, we employ difference-in-differences analysis that uses the jailbreak-immune Android app developers as a control group.

This study makes three contributions. First, we contribute to research on two-sided platforms by using the coopetition framework to theorize that complementors' competitive and cooperative incentives represent an inherent trade-off, and that platform governance policies can tilt the balance between the two incentives. This perspective considers complementors' competition and cooperation incentives simultaneously, extending existing platform research that commonly considers these incentives separately (cf., Hannah & Eisenhardt, 2018). Second, this study goes beyond prior platform governance research that focuses on the interactions between complementors and platforms (Claussen et al., 2013; Kretschmer & Claussen, 2016; Rietveld et al., 2019) by examining the interactions among complementors themselves. The focus on complementors' interactions points to a new direction of platform governance research. Third, in addition to discouraging complementors' contributions to platforms, as extant research suggests (Boudreau, 2012), this study highlights another implication of relaxed platform access control—a reduction of knowledge sharing among complementors. We further show that this effect varies across complementors, highlighting the value of incorporating complementors' heterogeneous characteristics to advance platform strategy research (McIntyre & Srinivasan, 2017).

2 | RELATED LITERATURE

2.1 | Platform governance

Platform governance broadly concerns the design and deployment of governance choices, including decision rights, incentive structures, and control mechanisms (Tiwana, 2013). A burgeoning stream of research has revealed valuable insights into how platform governance choices can shape complementors' incentives and activities in contributing to platforms. For example, Kretschmer and Claussen (2016) found that a new gaming console's backward compatibility, considered a hard regulation, negatively affected game developers' incentives in submitting new game titles. Soft incentive structures (Claussen et al., 2013) and the timing of their deployment (Rietveld et al., 2019) also nudged developers' incentives in creating software applications.

Among various platform governance designs documented in the literature, control of platform access receives particular attention. Platform access refers to the degree (e.g., scale or scope) to which complementors can access a platform. Extant research has revealed nuanced influences of platform access on complementors' incentives to contribute to platforms. For example, selectively granting platform access to hardware developers with heterogeneous skills and experiences significantly improves complementors' incentives for designing new handheld devices (Boudreau, 2010). By contrast, granting platform access widely and “letting a thousand flowers bloom” causes competitive crowding, hurting complementors' incentives for developing software applications (Boudreau, 2012).

A prominent policy for governing platform access is platform gatekeeping, which can be used to screen out imitating complementors and their offerings, thereby limiting competitive crowding and encouraging complementors to contribute (Boudreau & Jeppesen, 2015). For example, gatekeeping can screen out imitating mobile apps with deceptive icons to mitigate competitive tensions created by knockoffs (Wang, Li, & Singh, 2018). Maintaining a smaller set of complements with substitutive or overlapping features through gatekeeping may also attract new users to the platform, further alleviating competitive pressure among complementors (Casadesus-Masanell & Halaburda, 2014). Software platforms such as the Apple App Store are known for having strict app review processes that deny platform access to imitating apps to encourage original submissions from developers.

Although extant research has offered valuable insights into the implications of platform governance, research on how governance affects interactions among complementors remains scarce. Studies often view complementors as portfolio-like assets, focusing on the implications of platform governance for complementors' interactions with platforms, such as developers' release of software or games on a platform. Nevertheless, shaping strategic interactions among complementors (e.g., exchange of knowledge and ideas) through governance policies can also be important to platform owners, because such interactions may also affect complementors' value added to the platform (Adner & Kapoor, 2010).

Despite a scarcity of evidence on complementors' interactions, extant research suggests that such interactions may be affected by their competitive and cooperative incentives (Boudreau & Jeppesen, 2015). For example, Wareham et al. (2014) show that complementary developers of an enterprise resource planning software platform may share their code libraries to improve the usefulness of the platform collectively, yet when competition intensifies, they may opt to protect their source code to secure a competitive advantage of their own software. To address this gap in the platform governance literature, we use the coopetition framework from strategic management to guide our theoretical argument.

2.2 | Coopetition framework

The coexistence of agents' competitive and cooperative incentives has been studied extensively in the coopetition literature, primarily in an inter-firm context. The coopetition framework suggests that agents' competitive and cooperative incentives originate from the pursuit of private and common benefits (Khanna, 1998; Khanna et al., 1998). Common benefits accrue to each agent from the collective actions of all agents, and private benefits represent those an agent can extract unilaterally from other agents. For example, in an alliance, partnering firms can earn common benefits from the collective application of skills that synergistically create value for the alliance. Meanwhile, a firm can also earn private benefits by learning from its partners and applying the acquired knowledge to its own operations. The simultaneous presence of common and private benefits leads to the coexistence of agents' competitive and cooperative incentives.

Moreover, agents' competitive and cooperative incentives are shaped by a balance between their focus on common and private benefits. While partnering firms may enjoy common benefits that result from synergies otherwise unavailable when each firm operates alone, such cooperative relationships may shift toward competition when an opportunistic partner prioritizes extraction of private benefits by misappropriating a counterpart's resources. For example, in an alliance, when a young firm attempts to extract an established firm's knowledge opportunistically (Larsson, Bengtsson, Henriksson, & Sparks, 1998), the established firm may restrict knowledge sharing, reducing potential synergies and ultimately hurting alliance success (Lavie, 2006). Hence, the coopetition framework suggests that when the emphasis on private benefits increases, agents tend to act more competitively; by contrast, when the emphasis on common benefits increases, cooperation outweighs competition (Khanna et al., 1998). Consequently, competitive and cooperative incentives represent an inherent trade-off, which can shift between each other depending on environmental and organizational conditions (Hoffmann et al., 2018). A central issue in managing agents' coopetitive incentives, then, is to balance the agents' pursuit of common and private benefits.

Much of the coopetition literature has centered around knowledge sharing because knowledge is a source of competitive advantage (Grant, 1996), and because knowledge sharing reflects agents' competitive and cooperative incentives (Khanna, 1998). When competitive incentives increase, agents increasingly resort to knowledge protection to secure their proprietary knowledge (Lavie, 2006). When competitive incentives decrease, agents are more likely to form cooperative relationships to facilitate knowledge exchange and build knowledge stocks for innovation (Gnyawali & Park, 2011; Larsson et al., 1998). In summary, agents' coopetitive incentives determine their decisions to share knowledge, and knowledge sharing is more likely when they are more cooperative than competitive (Tsai, 2002).

While extant coopetition literature provides important insights regarding firms' competitive and cooperative incentives for strategic interactions, it rarely applies the coopetition framework to the platform context (cf. Hannah & Eisenhardt, 2018). This omission may be because conventional means, such as contractual agreements or hierarchical control, are less applicable to platforms. By examining the implications of platform gatekeeping for complementors' coopetitive incentives and knowledge sharing, we help expand the application scope of the coopetition framework.

3 | HYPOTHESES DEVELOPMENT

In keeping with recent work connecting platform research with strategic management research, we use the coopetition framework to examine how gatekeeping in an innovation platform may affect complementors' coopetitive incentives and shape their knowledge sharing activities. We consider platform gatekeeping an organizational condition that affects the interplay of competitive and cooperative incentives among complementors (Hoffmann et al., 2018). Platform gatekeeping, like any platform governance rule, may change strategically or accidentally. Examples of such changes abound, including the opening of IBM's System/360 to other companies and

Apple's iOS jailbreak giving platform access to unauthorized apps.[1] In this study, we focus on a lapse in a gatekeeping policy caused by an exogenous event that grants platform access to imitating complementors who are otherwise rejected.

Combining the literatures of coopetition and platform governance, we consider expanding the user base (i.e., the number of customers on the other side of the platform) to be common benefits to all complementors. The size of the user base associates positively with a platform's overall market size, and thus a larger user base benefits all complementors by attracting potential customers and increasing prospective gains. Conversely, we consider increasing the individual market share of users to be a private benefit for a complementor, because a complementor's materialized gain associates more with acquired attention from actual users and less with the number of potential users (Aral & Van Alstyne, 2011). Following this logic, to collectively pursue the common benefits of expanding the user base, complementors may have incentives to cooperate to attract additional users. To pursue the private benefits of enlarging or maintaining market share individually, complementors may have incentives to compete with each other for existing users (Aral & Van Alstyne, 2011). Since it reflects both complementors' competitive and cooperative incentives, knowledge sharing becomes more difficult to accomplish when complementors' pursuit of individual market share and the resulting competitive incentives increase.

To examine changes in complementors' knowledge sharing after an exogenous lapse in platform gatekeeping, we follow extant research in both coopetition and platform literatures, focusing on the coopetitive incentives of incumbent complementors who offer digital products with novel features or functionalities on innovation platforms (Wen & Zhu, 2019). We argue that a lapse in gatekeeping shifts the incumbent complementors' coopetitive balance toward competition for two reasons. First, such a lapse may directly intensify competition among complementors by granting unrestricted platform access to entrants who seek to imitate incumbent complementors and offer complements with overlapping or substitutive features (Parker & Van Alstyne, 2018). An extreme case occurs when some entrants opportunistically misappropriate market-proven complementary offerings by directly copying the designs and functionalities used by concept originators (e.g., copycat developers or software crackers) (Wang et al., 2018), yet conventional intellectual property protections may not easily extend to the protection of visual aspects, functionality, and sequences of such offerings (e.g., software applications) (Boudreau, 2012). Platform users thus end up consuming imitating offerings (Belleflamme & Peitz, 2019), lowering the user count for original ones. This situation may induce a strong sense of business stealing and concerns about a decrease in the market share of the incumbent complementors, shifting their coopetitive incentives toward competition.

Second, a lapse in gatekeeping may also indirectly intensify competition among complementors by reducing overall platform attractiveness (Casadesus-Masanell & Halaburda, 2014). For one reason, imitating complements (e.g., cracked apps) often suffer from quality problems (Lahiri & Dey, 2013). However, with the juxtaposition of original complements and low-quality knockoffs, it is difficult for users to assess the quality *ex ante*, resulting in a "lemons" problem (Belleflamme & Peitz, 2019). For another, platform users may overly explore (due to a lack of perfect foresight about others' choice) original and imitating complements, weakening the overall utility of complements to users (Casadesus-Masanell &

[1]IBM System/360 introduced the idea of computer compatibility, allowing machines across a product line to work with each other. Customers could buy a small mainframe and add to it as the computing system grew. Companies other than IBM could make peripheral equipment that worked with the System/360.

Halaburda, 2014; Halaburda, Piskorski, & Yildirim, 2018). For example, the distraction of imitating software may dilute user feedback such as bug reports (Arora, Caulkins, & Telang, 2006; Jiang, Sarkar, & Jacob, 2012), hindering quality improvements of such software and damaging the overall user experience (Cennamo & Santalo, 2013). Consequently, platforms may become less attractive to users after a lapse in gatekeeping. Since a reduction in platform attractiveness tends to associate with potential losses of existing and prospective users, incumbent complementors' concerns about a decrease in their market shares may become more prominent, shifting their coopetitive incentives toward competition.

Taken together, with a lapse in gatekeeping, imitating entrants may free ride on the incumbent complementors' knowledge to develop imitation complements, resulting in a potential reduction of incumbent complementors' market shares (i.e., private benefits). Such unilateral knowledge extraction and reduced private benefits may shift incumbent complementors' coopetitive balance toward competitive incentives. Thus, incumbent complementors may reduce knowledge sharing to tighten their control over proprietary knowledge and secure a competitive advantage when facing such a lapse. Therefore:

Hypothesis 1 *A lapse in platform gatekeeping reduces knowledge sharing among incumbent complementors on an innovation platform.*

Similar to the notion that organization design has to take into account the underlying characteristics of the firm's knowledge base (Birkinshaw, Nobel, & Ridderstrale, 2002), platform governance design must consider characteristics of complementors shaped by their knowledge base. Hence, although a lapse in platform gatekeeping may reduce complementors' knowledge sharing on average, the effect will likely vary across complementors with a heterogeneous knowledge base. Since the reduction in knowledge sharing is related to increased competitive pressure caused by imitating complementors, complementors' knowledge characteristics that have implications for imitation and competition are particularly relevant. Prior research on knowledge and innovation suggests that complementors' knowledge complexity and knowledge routineness fit this context. Specifically, complex knowledge is difficult to transfer and learn (Zander & Kogut, 1995), and thus innovators with complex knowledge may be less concerned about competitors' imitation and the associated competitive pressure (Fleming & Sorenson, 2001; Rivkin, 2001). Frequently or routinely used knowledge, by contrast, is easy to communicate and absorb, and the associated innovation is easy to replicate (Jehn, 1995; Wong, 2004). These two characteristics of complementors thus represent essential boundary conditions for the proposed coopetition mechanism.

Seminal research defines knowledge complexity as the degree of interdependence inherent in the subcomponents of knowledge (Fleming, 2001; Simon, 1996). Due to potential conflicting constraints, intractable interconnections, and tight functional interdependencies among knowledge subcomponents, complex knowledge and associated innovations are difficult for competitors to imitate (Grant, 1996; Rivkin, 2001), a notion that applies to innovations on platforms. For example, software developers with complex knowledge often have rich programming experience with managing a complicated array of technical constraints that interact with different layers of platform architectures (Cennamo, Ozalp, & Kretschmer, 2018; Cusumano, 2004). Hence, software applications created by such developers rely on the diverse and interdependent subsystems of the platform, making it challenging for other programmers to imitate their work (Kapoor & Agarwal, 2017). Relatedly, due to their technical complexity and difficulty for imitation, innovative complements based on complex knowledge are often limited in supply

(Fleming, 2001), and therefore platform users may be forced to join or stay with a platform because of such complements. For example, digital gamers may choose to stay with PlayStation or Xbox consoles to enjoy technologically sophisticated and exclusive game titles (Gil & Warzynski, 2015). Thus, complementors who master complex knowledge may be less concerned about the increase in competitive pressure associated with the relaxed platform access to imitating complementors. As a result, to the extent that imitating complementors are given platform access due to a lapse in gatekeeping, incumbent complementors with greater knowledge complexity will be less sensitive and experience a smaller shift of the coopetitive balance toward competition, mitigating the negative main effect in Hypothesis . Thus:

Hypothesis 2 *A lapse in platform gatekeeping reduces knowledge sharing to a smaller extent among incumbent complementors with greater knowledge complexity on an innovation platform.*

Building on extant research on task routineness, knowledge routineness represents the degree to which knowledge is being applied repetitively, and the limited functional variability regarding the application of the knowledge (Jehn, 1995; Wong, 2004). Because knowledge of high routineness is more commonly seen or repetitively used, complementors are more likely to form a common ground of understanding (Wong, 2004), facilitating the transfer and learning of such knowledge and subsequent imitation of the resulting innovation (Zander & Kogut, 1995). Furthermore, the low functional variability of commonly-used knowledge (Jehn, 1995) indicates that such knowledge often serves specific but ordinary purposes and can be recombined or applied only in limited ways, making commonly-used knowledge particularly vulnerable to the innovation exhaustion problem and reverse engineering attempts (Fleming, 2001; Leiponen & Helfat, 2010). Taken together, complementors relying on routine knowledge are more likely to offer ordinary products that are relatively easy to imitate and abundant in supply. Thus, complementors with greater knowledge routineness may be particularly concerned about the increase in competitive pressure associated with the relaxed platform access to imitating complementors. Consequently, to the extent that imitating complementors are given platform access due to a lapse in gatekeeping, incumbent complementors with greater knowledge routineness will be more sensitive and experience a larger shift of the coopetitive balance toward competition, accentuating the negative main effect in Hypothesis . Thus:

Hypothesis 3 *A lapse in platform gatekeeping reduces knowledge sharing to a larger extent among incumbent complementors with greater knowledge routineness on an innovation platform.*

4 | DATA AND METHODS

4.1 | Research design and identification strategy

To test the hypotheses, we focus on Apple's iOS app platform, a leading platform for mobile software applications, and examine iOS developers' knowledge sharing after an exogenous lapse in platform gatekeeping due to the jailbreak of iOS 7. The iOS jailbreak is a hacking that exploits loopholes to remove iOS built-in restrictions. After the jailbreak, apps that were previously rejected by iOS's app review became available to users, representing a lapse in the gatekeeping of iOS. In a typical jailbreak, a few elite hackers jointly developed an automated jailbreaking program and made it available to the public for free.[2] After installing the program, iOS users could easily jailbreak their devices and install apps that were unavailable in the Apple App Store (i.e., jailbreak apps). A major type of jailbreak apps is pirated apps (i.e., copycats or cracked apps), which are explicitly discouraged by the app review.[3] Pirated apps typically closely imitate or directly copy the functionalities of existing legitimate apps but are offered to users for a discount or even free (Geng & Lee, 2013), causing significant revenue losses to iOS app developers (Koetsier, 2017; Rasch & Wenzel, 2013).[4] Some pirated apps are unreliable and may even contain malicious content, which could damage customer experience and result in user attrition (Lahiri & Dey, 2013). Consequently, the business stealing and user attrition induced by jailbreak apps could intensify the competitive pressure faced by iOS developers and affect their knowledge sharing. For this reason, iOS jailbreak provides an appropriate context in which to test our hypotheses.

The jailbreak of iOS 7 especially fits current analyses because the timing of the jailbreak was unexpected. The iOS 7 upgrade notably improved system security against jailbreaks. According to Apple's security reports and the U.S. National Vulnerability Database, known exploits (i.e., loopholes) used in previous jailbreaks were patched in the iOS 7 system, making jailbreaking especially challenging. Moreover, unlike previous jailbreaks that usually documented hackers' progress on social media sites, iOS 7's jailbreak was hidden from the public.[5] Thus, the technical community at that time was pessimistic about the possibility of a jailbreak of iOS 7 (Anthony, 2013). Overall, the timing of the jailbreak of iOS 7 was considered uncertain, given the technical complexity and the opaque hacking progress.[6] The unusually long time for the iOS 7 jailbreak to be released (i.e., 95 days)[7] also gives us sufficient time to observe changes in knowledge sharing activities before and after the treatment. We thus used a quasi-experimental design to compare the knowledge sharing activity of iOS developers (treatment group) with that of "shock-immune" Android app developers (control group) before and after the jailbreak, as illustrated in Figure 1. This difference-in-differences (DD) approach helps reduce endogeneity and strengthen the causality of findings (Angrist & Pischke, 2009).

[2]The jailbreak was a major event for the iOS platform, and it received extensive coverage by IT news media. As a result, the lag between the release of a jailbreak program and the program being widely available was short and negligible. For details, see: http://www.forbes.com/sites/anthonykosner/2013/02/10/what-7-million-jailbreaks-are-saying-is-apple-listening

[3]Another major type of jailbreak apps is "tweaks", which adds additional functionalities to existing iOS apps. This app type is less influential to the current context for two reasons. First, the programming skills required for tweaks and existing iOS apps are similar. Thus, tweaks are not significantly superior to existing apps. Second and more importantly, sales of tweaks (millions) are much smaller than that of pirated apps (billions).

[4]iOS app developers rarely develop jailbreak apps because, once discovered, their developer licenses can be revoked.

[5]The history of the hacking team's Twitter account and posts on the jailbreak subreddit (a subforum on Reddit for jailbreak discussions) yielded no information regarding the iOS jailbreaking progress. Even the creator of Cydia, the de facto marketplace for jailbreak apps, received no updates about the progress. On social media he stated, "So, I got no lead time on evasi0n7 (the hacking team), nor was I asked for an official iOS Cydia; I was not given builds, nor was I asked for things to test."

[6]We suspected that new developers who did not experience the impact of previous jailbreaks directly and who focused on familiarizing themselves with Apple's coding syntax/environment were less likely to track the news about the jailbreak. Thus, the jailbreak of iOS 7 was likely exogenous to these developers. Following this reasoning, we identified 260 developers of this type by looking for StackOverflow users who created accounts after the jailbreak of iOS 6 and tested our main hypothesis with this subgroup of developers who had never experienced a jailbreak before the iOS 7 jailbreak, finding qualitatively similar results.

[7]We report details regarding past iOS jailbreaks in the Supporting Information.

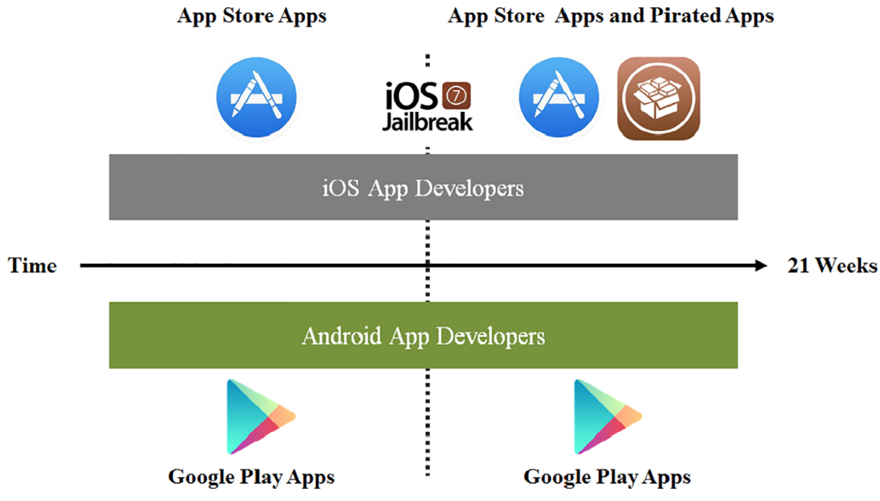

FIGURE 1 Graphical illustration of the research design. *Note:* Figure 1 depicts the quasi-experimental research design. Before the jailbreak of iOS 7 (the treatment), app developers of iOS 7 ("treatment group" on the top left part) are relatively less concerned about pirated apps because of the enforced gatekeeping by Apple. By contrast, Android app developers ("control group" on the bottom left part) do not experience such gatekeeping by Apple. After the treatment, Apple's gatekeeping is weakened by iOS 7 jailbreak, and pirated apps are accessible to jailbroken device users and pose competitive pressure to incumbent app developers of iOS 7. By contrast, Android app developers (on the bottom right part) should not experience such changes because the jailbreak does not affect them

We used Android app developers as a control group for three reasons. First, they are unaffected by jailbreaks because the Android platform allows users to "sideload" unauthorized apps by default. Second, other than the difference in platform access control, extant research suggests that Android developers are comparable to iOS developers (Kapoor & Agarwal, 2017). Our study of StackOverflow corroborates this notion by showing that these two groups of developers demonstrate similar knowledge sharing behaviors on online forums. Third, within this study's timeframe, we failed to find other material changes in platform access control across the Android and iOS platforms.[8]

After constructing the treatment and control groups above, we conducted coarsened exact matching (CEM) to create a matched sample to strengthen the assumptions of a quasi-experimental design. CEM offers advantages over alternative matching methods because of its incorporation of monotonic balance bounding (Blackwell, Iacus, King, & Porro, 2009) that reduces model-dependence, bias, and inefficiency (Iacus, King, & Porro, 2012), and it is used increasingly in strategy research in recent years (Younge, Tong, & Fleming, 2015).

[8]In the Android ecosystem, there is a concept called "rooting", which is sometimes confused with jailbreaking. A jailbreak allows a user to use software that Apple does not authorize on iOS devices (e.g., iPhones). By contrast, rooting gives a user access to largely the entire operating system. For example, a user can remove the whole operating system and replace it with a user-created one that contains tweaks and enhancements. Apart from such technical differences, the magnitude of the impact also differs between the two. A jailbreak often has an ecosystem-wide impact—once a jailbreak is achieved, most devices running the affected iOS versions can be jailbroken. However, rooting is often manufacturer- or device-specific. For example, a Huawei device may need specific software for rooting, but an LG device may not need third-party software to be rooted. For these reasons, rooting in the Android ecosystem has little impact on ecosystem-wide changes in app developers' activities.

4.2 | Complementors' knowledge sharing on online forums

Knowledge sharing among complementors is difficult to observe and measure, because platform-provided tools or information (e.g., page views, submission frequency, or product ranking) rarely capture such activity (Greenstein & Nagle, 2014). Knowledge sharing is also usually qualitative in nature, thus presenting a measurement challenge (Stavrianou, Andritsos, & Nicoloyannis, 2007). Fortunately, third-party online forums that complementors visit or use frequently provide rich information about their knowledge sharing activity (Faraj, Jarvenpaa, & Majchrzak, 2011), an observation that guided our search for this study's context. Specifically, our analyses leveraged a dataset that tracks a large sample of iOS and Android app developers (i.e., complementors of the iOS and Android platforms) and their knowledge sharing (i.e., posting) activity on a major online forum that IT professionals often visit—StackOverflow.

This empirical context is appropriate for our study for three reasons. First, app developers actively use StackOverflow to share knowledge and learn from peers (Abdalkareem et al., 2017).[9] Second, StackOverflow represents a large, diverse set of app developers from different countries and regions. Web traffic data show that app developers who actively participate on StackOverflow come from major countries and regions, including the United States, Europe, Japan, and India (Vasilescu, Filkov, & Serebrenik, 2013). Third, these developers have diverse backgrounds (e.g., company- and self-employed developers), reducing concerns that some types of programmers self-select into this forum.

4.3 | Sample construction

4.3.1 | StackOverflow sample

We obtained the primary data from online data collection at StackOverflow. Although researchers have rarely used this data source in strategic management, similar data sources have been used in extant management research (von Krogh & von Hippel, 2006). To construct the base sample, we began with all StackOverflow users who posted at least one question and received at least 500 view counts in the iOS or Android subforum within the 52 weeks before the release date of iOS 7 on September 18, 2013. We excluded old and inactive users, and users who posted rarely viewed, trivial questions, because such users might not represent active app developers.

We collected the activity log for the app developers for 21 weeks (10 weeks before and 10 weeks after the treatment, and during the treatment week) from October 13, 2013 to March 8, 2014, leading to a weekly panel dataset before and after the jailbreak event. We chose this observation window for two reasons. First, there might have been an unusually high volume of discussions regarding app development during the first few weeks after the release of iOS 7. Therefore, we started observations about 4 weeks after the release on September 18, 2013. Second, the average time to develop an app is around 10 weeks.[10] Hence, allowing a 10-week observation window before and after the jailbreak of iOS 7 on December 22, 2013 covers typical app developing cycles. Following extant literature (Kuk, 2006), we retained app developers who

[9]Our interviews with active app developers confirmed that developers use StackOverflow frequently to acquire or share programming-related knowledge.
[10]For more details, see: http://www.accella.net/iphone-app-development-timeline

were in the top 10% of active posters, based on the posting frequency distribution in the study's observation window, to exclude developers who did not consider this forum to be a primary channel of knowledge sharing.

We also accounted for the presence of multi-homing app developers. Some skilled developers might create apps for both the iOS and Android platforms, and including such developers to a study might introduce biases. Nevertheless, on the individual level, multi-homing app development is challenging and should apply only to a minute proportion of developers, due to many notable differences, such as application programming interfaces (APIs) and programming syntaxes, between the two platforms. Thus, we excluded 55 multi-homing app developers who posted questions on both the iOS and Android subforums.[11] Consequently, the sample consisted of 418 iOS and 841 Android app developers.

In a final step, we conducted CEM to create a matched sample. To achieve greater balance between the treatment and control groups, we matched on all covariates that might affect app developers' knowledge sharing activity within our observation window, and up to 52 weeks before the iOS 7 release to account for historical patterns. This step eliminated 357 developers from our sample.

4.4 | Variables and measurement

4.4.1 | Unit of analysis

Following extant studies (Kuk, 2006; Sun & Zhu, 2013), the unit of analysis was individual app developer by week. We constructed a panel-structured dataset that recorded the weekly knowledge sharing activities of app developers based on their iOS- and Android-related discussions on StackOverflow.

4.4.2 | Dependent variable

The extent of app developers' knowledge sharing was captured by the frequency of knowledge sharing (i.e., *Post Count*), measured as the total number of questions and answers posted by an iOS or Android app developer in a week.[12]

4.4.3 | Explanatory variables

Following the quasi-experimental design, the DD "treatment group" variable, *iOS*, was a dummy variable that took a value of 1 for iOS app developers, and zero for Android app developers. The DD "after" variable, *After*, was a dummy variable that took a value of 1 for the weeks after the jailbreak event, and zero for the weeks before the event, including the focal week of the jailbreak. The interaction term *DD* (*iOS*After*) was calculated to identify the treatment effect of the lapse in gatekeeping (H1).

[11]We also tested the main hypothesis by using the multi-homing sub-sample, finding qualitatively consistent results.

[12]We also measured the quality of posts using average user-provided ratings, aggregated to the individual-week level. Results were consistent with the main analysis on *Post Count*.

Testing H2 required a measure of developers' knowledge complexity. Following the prior literature on knowledge and innovation, we used Kauffman's NK framework to construct this measure (Fleming, 2001; Fleming & Sorenson, 2001; Kauffman, 1993). This framework suggests that the complexity of a system is determined by the degree of interdependence among its subcomponents. In the context of knowledge, a piece of knowledge is considered complex when its subcomponents have high interdependence. To capture the interdependence among subcomponents of app developers' knowledge, we used the tags assigned to each StackOverflow question. A tag is a keyword used to categorize questions based on app development knowledge. A single StackOverflow question may have multiple tags, and thus a set of tags could represent knowledge subcomponents. Based on tags associated with each question, we calculated app developers' knowledge complexity in three steps, adapted from the complexity measure created by Fleming and Sorenson (2001) in the patent literature (Zhang & Tong, 2020). In the first step, we calculated, at the tag level, an "ease of recombination" measure, ET_i, for each tag i in StackOverflow's tag pools to capture the overall "ease of recombination" between a particular tag and other tags:

$$\text{Ease of recombination Tag } i \equiv \mathrm{ET}_i = \frac{\text{Count of tags previously combined with tag } i}{\text{Count of questions associated with tag } i} \tag{1}$$

In the second step, we calculated, at the question level, a complexity measure, K_q, for question q[13]:

$$\text{Complexity of Question } q \equiv K_q = \frac{\text{Count of tags associated with question } q}{\sum\limits_{q \in \mathrm{i}} \mathrm{ET}_i} \tag{2}$$

where $\mathrm{ET_i}$ is the degree of "ease of recombination" for tag i calculated in Equation (1). This measure represented the inverse of the average "easiness" to recombine a group of technical tags associated with a question, or the average level of complex and hard-to-achieve interdependence of the tags (Fleming & Sorenson, 2001).

Finally, given that it takes many years for developers to accumulate programming knowledge and experience, and that the observation window was only 21 weeks, developers' knowledge complexity in this context is likely to be relatively stable and time invariant. Hence, in the third step, we averaged all question-level complexity measures to represent knowledge complexity at the developer level.

Testing H3 required a measure of developers' knowledge routineness, which relates to the application frequency and functional variability of knowledge. Following extant literature (Jehn, 1995; Wong, 2004), we used tag frequency to capture knowledge routineness of app developers. When a tag is mentioned frequently, the associated knowledge is likely to be routinely and invariably used during app development. We calculated knowledge routineness in

[13]A developer's knowledge sharing may include knowledge solicitation (questions) and provision (answers). Since developers' answers to questions were not tagged on StackOverflow, we were not able to calculate answer-level complexity and routineness measures directly. Answers often related closely to affiliated questions, so in addition to the questions posted by developers, we included the questions to which they provided answers when calculating developer-level knowledge complexity and routineness measures.

three steps. In the first step, we calculated tag-level routineness by capturing overall tag use frequency:

$$\text{Routineness of Tag } i \equiv RT_i = \text{Log}(\text{Count of questions associated with tag } i) \tag{3}$$

In the second step, we calculated, at the question level, a routineness measure, R_q, for question q:

$$\text{Routineness of Question } q \equiv R_q = \frac{\sum_{q \in i} RT_i}{\text{Count of tags associated with question } q} \tag{4}$$

where RT_i is the routineness measure for tag i calculated in Equation (3). Finally, to calculate the individual-level *Knowledge Routineness* measure, we averaged all question-level routineness values for a developer in the observation window.[14]

4.4.4 | Other covariates

We included several covariates to control for other differences between the treatment and control groups. We included the average count of question views (*Question View Count*) and average count of question comments (*Question Comment Count*) for all questions posted by an app developer in a given week, because these question-related variables may also explain the variance in the dependent variable (Sun & Zhu, 2013). In addition, prior research suggests that knowledge sharing activities may correlate with other social activities in an online community (Faraj & Johnson, 2011). Thus, social interactions in the form of editing posts, commenting on others' posts, bookmarking posts, and replying to posts may affect app developers' knowledge sharing activity. We therefore included app developers' editing frequency (*Edit Activity*), commenting frequency (*Comment Activity*), favorite-adding frequency (*Add-Favorite Activity*), and non-app-related post frequency and quality (*Non-app Post Count and Non-app Post Quality*), to control for these time-varying factors related to app developers and their use of online communities (Roberts, Hann, & Slaughter, 2006). We measured the quality of non-app-related posts as the average score of the posts.[15]

In addition, we incorporated a full set of individual app developer fixed effects to remove effects due to unobserved, time-invariant characteristics of the app developers (e.g., individual intelligence or background) (Wooldridge, 2010). We also included a full set of week fixed effects to control for environmental factors or week-by-week variations that may affect results. In all regressions, robust standard errors were clustered at the individual app developer level (Wooldridge, 2010). Panel A of Table 1 provides a list of all variables, their definitions, and their descriptive statistics, and Part B reports correlation coefficients for all variables.

[14] Although measures of knowledge routineness and knowledge complexity were both calculated based on question tags, the two measures represent distinct constructs. Knowledge complexity builds on the difficulty of tags' collective and interdependent use, whereas knowledge routineness is based on the tags' application frequency, without considering their co-appearance with other tags.

[15] StackOverflow uses a scoring system that allows other users to determine the quality of a given post by voting collectively. When a user finds a post insightful or important, she can upvote it, increasing the total score the post receives; she can also choose to downvote a post if she finds the information unhelpful. Thus, higher-quality posts tend to receive higher scores.

TABLE 1 Variable definition, summary statistics, and correlation coefficients

Panel A: Variable definition and summary statistics

Variables	Definition	Mean	*SD*	Min.	Max.
Post count	Number of posts that are related to iOS or Android from an app developer in a week	1.67	3.08	0	62
iOS	A dummy variable that equals to 1 for an iOS app developer, and 0 for an Android app developer	0.33	0.47	0	1
After	A dummy variable that equals to 1 for weeks after the jailbreak of iOS 7, and 0 otherwise	0.50	0.50	0	1
Complexity	Average level of technical interdependence among tags used by an app developer	0.58	0.70	0.10	2.91
Routineness	Average level of tag frequency for tags used by an app developer	3.46	2.18	0.42	11.76
Question view count	Average counts of views for all questions posted by an app developer in a week	91.19	348.87	0	3,579
Question comment count	Average counts of comments for all questions posted by an app developer in a week	0.89	1.80	0	25
Edit activity	Number of edits proposed by an app developer in a week	0.11	1.62	0	106
Comment activity	Number of comments proposed by an app developer in a week	2.65	6.07	0	212
Add-favorite activity	Number of questions that are labeled as favorites by an app developer in a week	0.33	1.47	0	46
Non-app post count	Number of posts that are unrelated to iOS or Android from an app developer in a week	2.85	3.32	0	51
Non-app post quality	Average score of posts that are unrelated to iOS or Android from an app developer in a week	0.37	1.23	−10	82

Panel B: Correlations

Variables	1	2	3	4	5	6	7	8	9
1. Post count									
2. Complexity	0.06								
3. Routineness	0.30	0.28							
4. Question view count	0.24	0.03	0.09						
5. Question comment count	0.35	0.05	0.19	0.18					
6. Edit activity	0.03	0.02	0.04	0.02	0.03				
7. Comment activity	0.22	0.04	0.12	0.06	0.17	0.10			
8. Add-favorite activity	0.09	0.03	0.07	0.03	0.06	0.07	0.13		
9. Non-app post count	0.39	0.10	0.26	0.16	0.22	0.03	0.21	0.09	
10. Non-app post quality	0.17	0.03	0.04	0.25	0.06	0.01	0.04	0.03	0.08

Note: Panel A repots the summary statistics of the variables in the full sample. Please refer to the Data and Methods section for detailed operationalization. Panel B repots the correlation coefficients among the variables in the full sample. $n = 26,439$.

4.5 | Model specification

Using the difference-in-differences research design, we estimated the following model specification to examine the treatment effect of a lapse in gatekeeping (jailbreak) on knowledge sharing among app developers:

$$Y_{it} = \beta_0 + \beta_1 DD_{it} + \delta_2 X_{it} + T_t + I_i + \varepsilon_{it} \quad (5)$$

$$Y_{it} = \beta_0 + \beta_1 DD_{it} + \beta_2 DD_{it} * Complexity_i + \delta_1 D_{it} * Complexity_i + \delta_2 X_{it} + T_t + I_i + \varepsilon_{it} \quad (6)$$

$$Y_{it} = \beta_0 + \beta_1 DD_{it} + \beta_3 DD_{it} * Routineness_i + \delta_1 D_{it} * Routineness_i + \delta_2 X_{it} + T_t + I_i + \varepsilon_{it} \quad (7)$$

where Y_{it} is the dependent variable for app developer i in week t; β_0 is the intercept; β_1 in Equation (5) identifies the treatment effect of DD_{it} (*iOS*After*); β_2 and β_3 in Equations (6) and (7) identify the moderating effects (DD_{it}**Complexity* and DD_{it}**Routineness*), respectively; D_{it} are the first-order terms of the DD_{it} term; X_{it} are covariate controls; T_t is a vector of week fixed effects; I_i is a vector of individual app developer fixed effects; and ε_{it} is the error term.[16]

5 | RESULTS

5.1 | Assessment of the comparability assumption for the research design

A key assumption of our research design requires that iOS and Android app developers are comparable. We assessed this assumption by visually examining the "parallel trend" of developers' posting activities (Angrist & Pischke, 2009). As shown in Figure 2, we followed prior research (Beattie, Durante, Knight, & Sen, 2019) by plotting the weekly *DD* estimates from a splined regression of iOS and Android developers' posting activities before and after the treatment. Figure 2 shows that before the iOS 7 jailbreak, the posting patterns of iOS (treatment group) and Android (control group) app developers had a similar trend.[17]

5.2 | Main results

Results of DD regressions that tested H1–H3 appear in Table 2. Results in Column 1 suggest that when the jailbreak of iOS 7 weakened the gatekeeping policy, incumbent iOS app developers shared knowledge less frequently. The negative coefficient on the DD term indicates that iOS app developers posted 0.12 (p = .008) fewer posts (about a 7% drop in terms of economic

[16]We ran both linear and count models for *Post Count*. We considered linear models a baseline that offers easy-to-interpret results. Cameron and Trivedi (1998, p. 89) suggest that in large sample analysis, linear models "in practice give results qualitatively similar" to non-linear models. When we followed Allison and Waterman (2002) to fit a negative binomial model with fixed effects to our data, we obtained similar results (see results in the Supporting Information).
[17]We further assessed the "parallel tread" assumption, the comparability of general programing knowledge, and covariate balance between iOS and Android app developers, reporting the findings in the Supporting Information.

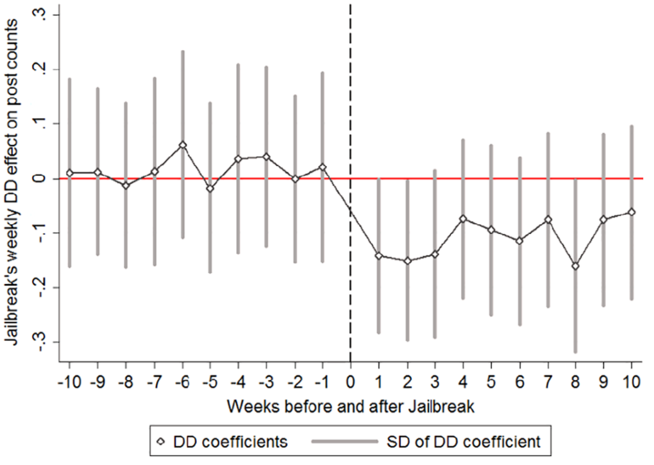

FIGURE 2 Parallel trend graphs. *Note:* Figure 2 examines the "parallel trend" assumption in the pre and post treatment periods using the estimates from a splined regression. The graph shows that before the jailbreak of iOS 7, the posting patterns of iOS app developers ("treatment group") and Android app developers ("control group") are relatively comparable in the pre-treatment period

magnitude) on StackOverflow after the jailbreak, compared with Android app developers. Thus, H1 was supported.[18]

H2 suggests that the main effect in H1 is mitigated for app developers with greater knowledge complexity. The results in Column 2 support this prediction. The positive coefficient (p = .048) on the triple-DD term (*DD*Complexity*) in Column 2 indicates that iOS app developers with greater knowledge complexity experienced a smaller drop in posting frequency on StackOverflow after the jailbreak event, compared with Android app developers. H3 posits that the main effect in H1 is magnified for app developers with greater knowledge routineness. The results in Column 3 also support this prediction. The negative coefficient (p = .058) on the triple-DD term (*DD*Routineness*) in Column 3 indicates that iOS app developers with greater knowledge routineness experienced a larger drop in posting frequency.

Since app developer-specific characteristics may correlate with their activities (e.g., Boudreau & Jeppesen, 2015), we considered allowing the covariates to vary for app developers with different degrees of knowledge complexity or routineness, to gain further insights into the moderation effects. For succinctness and practicality (Bertrand & Morse, 2011), we followed extant research (Sleptsov, Anand, & Vasudeva, 2013) and split our sample according to complementors' knowledge complexity and routineness. Table 3 reports results of this analysis. First, we created subsamples of app developers within the upper and lower levels of each moderator. With the subsample of app developers who were low in knowledge complexity (Column 1 of Panel A in Table 3), we found that after the jailbreak, iOS app developers posted 0.222 fewer posts on average (p < .001). On the contrary, with the subsample of app developers who were high in knowledge complexity (Column 2 of Panel A), the average post count after the jailbreak decreased only by 0.029 (p = .642). These findings suggest that app developers with low knowledge complexity were affected more by the iOS jailbreak, while app developers with high knowledge complexity were insensitive to it. Similarly, using the subsample of app developers who were low in knowledge routineness (Column 3 of Panel A), we found that these app developers posted, on average, 0.106 fewer posts after the jailbreak (p = .111). Using the subsample of app developers who were high in knowledge routineness (Column 4 of Panel A), we found

[18]Since online knowledge sharing involves knowledge solicitation and knowledge provision, we performed additional analyses to examine the implications of a lapse in gatekeeping on the quantity and quality of the two steps of complementors' knowledge sharing, finding consistent results. Details appear in the Supporting Information.

TABLE 2 Difference-in-differences results for complementors' knowledge sharing

Variables	(1)	(2)	(3)	(4)	(5)	(6)
	CEM sample			Full sample		
	Post count	Post count	Post count	Post count	Post count	Post count
Question view count	0.000	0.000	0.000	0.000	0.000	0.000
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
Question comment count	−0.022	−0.023	−0.042	−0.010	−0.012	−0.035
	(0.009)	(0.008)	(0.013)	(0.008)	(0.007)	(0.011)
Edit activity	−0.031	−0.031	−0.031	−0.020	−0.015	−0.019
	(0.009)	(0.008)	(0.021)	(0.006)	(0.005)	(0.010)
Comment activity	0.071	0.071	0.091	0.050	0.051	0.070
	(0.007)	(0.006)	(0.010)	(0.007)	(0.006)	(0.008)
Add-favorite activity	0.052	0.061	0.056	0.049	0.055	0.046
	(0.026)	(0.026)	(0.040)	(0.018)	(0.018)	(0.025)
Non-app post count	0.798	0.807	0.811	0.812	0.819	0.843
	(0.020)	(0.018)	(0.025)	(0.019)	(0.017)	(0.021)
Non-app post quality	0.514	0.470	0.547	0.397	0.388	0.390
	(0.080)	(0.066)	(0.120)	(0.068)	(0.056)	(0.078)
DD (iOS*after)	−0.120	−0.213	0.024	−0.114	−0.197	0.033
	(0.045)	(0.053)	(0.145)	(0.045)	(0.053)	(0.144)
After*complexity		−0.016			−0.025	
		(0.031)			(0.035)	
DD*complexity		0.099			0.103	
		(0.050)			(0.051)	
After*routineness			0.013			0.003
			(0.014)			(0.012)
DD*routineness			−0.045			−0.044
			(0.024)			(0.023)
Constant	−0.790	−0.896	−1.122	−0.790	−0.933	−0.997
	(0.072)	(0.050)	(0.104)	(0.072)	(0.051)	(0.104)
Observations	18,942	18,942	18,942	26,439	26,439	26,439
R-squared	0.680	0.681	0.696	0.667	0.670	0.683
Number of users	902	902	902	1,259	1,259	1,259
Week FE	Yes	Yes	Yes	Yes	Yes	Yes
Individual FE	Yes	Yes	Yes	Yes	Yes	Yes

Note: This table reports results for the baseline and moderation hypotheses. Columns 1–3 report the results using the CEM sample. Columns 4–6 report the results using the full sample. Specifically, Columns 1 and 4 report results for H1, Columns 2 and 5 for H2, and Columns 3 and 6 for H3. Standard errors clustered on the individual level are reported in parentheses.

TABLE 3 Split sample analyses for the moderating effects

Panel A: DD results for developers within the upper/lower level of knowledge complexity/routineness

	(1)	(2)	(3)	(4)
	Complexity low	**Complexity high**	**Routineness low**	**Routineness high**
Variables	**Post count**	**Post count**	**Post count**	**Post count**
DD (iOS*after)	−0.222	−0.029	−0.106	−0.172
	(0.056)	(0.063)	(0.067)	(0.053)
Observations	11,424	7,518	5,943	12,999
R-squared	0.685	0.673	0.704	0.668
Number of users	544	358	283	619
Other controls	Yes	Yes	Yes	Yes
Week FE	Yes	Yes	Yes	Yes
Individual FE	Yes	Yes	Yes	Yes

Panel B: DD results for developers with different combinations of levels of knowledge complexity and routineness

	(1)	(2)	(3)	(4)
	Complexity low routineness low	**Complexity high routineness low**	**Complexity low routineness high**	**Complexity high routineness high**
Variables	**Post count**	**Post count**	**Post count**	**Post count**
DD (iOS*after)	−0.209	0.084	−0.234	−0.069
	(0.078)	(0.122)	(0.077)	(0.072)
Observations	4,326	1,617	7,098	5,901
R-squared	0.709	0.692	0.671	0.668
Number of users	206	77	338	281
Other controls	Yes	Yes	Yes	Yes
Week FE	Yes	Yes	Yes	Yes
Individual FE	Yes	Yes	Yes	Yes

Note: Panel A reports results of the split sample analyses with subsamples of app developers within the upper and lower levels in the CEM sample. Model specifications are the same as the main analysis in Table 2. Standard errors clustered on the individual level are reported in parentheses. Panel B reports results of the split sample analyses with subsamples of app developers with different combinations of levels of knowledge complexity and routineness in the CEM sample. Model specifications are the same as the main analysis in Table 2. Standard errors clustered on the individual level are reported in parentheses.

that these app developers posted 0.172 fewer posts (p = .001). These findings suggest that although app developers with highly routine knowledge were sensitive to the jailbreak, some app developers with low knowledge routineness also tended to post less frequently after the jailbreak. While such complementors' knowledge may be rarely used or applicable only to a niche domain (i.e., low knowledge routineness), some of such knowledge may still be easy to imitate; thus they may also have been concerned about the iOS jailbreak.

Second, since complementors' knowledge complexity and routineness are distinct moderators, it was possible to have app developers with different combinations of the two characteristics. To investigate how these characteristics may shape app developers' knowledge sharing after the iOS 7 jailbreak, we created four subsamples based on the degrees of knowledge complexity and routineness—high complexity and high routineness, high complexity and low routineness, low complexity and high routineness, and low complexity and low routineness. We then reran our primary model using each subsample. Reported in Panel B of Table 3, the results show that knowledge complexity was more influential than routineness in shaping complementors' knowledge sharing after the jailbreak. For instance, app developers with low knowledge complexity (Columns 1 and 3 of Panel B) were sensitive to the jailbreak in general regardless of their degree of knowledge routineness ($p = .007$ and $p = .002$, respectively), but app developers with high complexity (Columns 2 and 4 of Panel B) exhibited little change in posting frequency overall ($p = .491$ and $p = .338$, respectively). In particular, app developers with the most advantageous position (i.e., those with complex and rare knowledge; Column 2 of Panel B) were indifferent to the jailbreak, and even posted slightly more frequently after the jailbreak. On the contrary, app developers with the most disadvantageous position (i.e., low complexity and high routineness; Column 3 of Panel B), demonstrated the largest drop in posting frequency after the jailbreak. The findings also partially corroborate our supposition that some app developers with low knowledge routineness were also concerned about the jailbreak due to low knowledge complexity. Overall, this split sample analysis not only lends general support to H2 and H3, but also enriches our interpretation of the moderating effects of complementors' knowledge complexity and routineness.

5.3 | Robustness checks

We examined the robustness of our results. First, we included the results using the full sample in Table 2. As Columns 4–6 in Table 2 show, we continued to find coefficients on the DD variable to have signs, magnitudes, and p-values that were qualitatively similar to the results using the CEM sample, suggesting that our results are robust to both unmatched and matched samples.[19]

Second, we tested the robustness of our results using two alternative sample constructions. In the first robustness test, we used a different control group. Because of the intense competition between the iOS and Android ecosystems, a significant shock to the iOS ecosystem might have affected the dynamics on the Android platform and the activities of its app developers indirectly. To address this concern, we created another sample that consisted of Windows Phone OS app developers on StackOverflow, which served as an alternative control group. Applying the same model specifications as in Columns 1–3 of Table 2 and using this new control group, we again found qualitatively similar statistics for the DD variable and the three-way interaction terms, as shown in Columns 1–3 of Table 4.

[19]Results were robust to samples of app developers with alternative activity levels. See the Supporting Information for details.

We also used an alternative data source to create a new sample for robustness tests. Despite many of the advantages it offers, StackOverflow is not the only active forum that developers visit for app development-related discussions. Thus, we also used Reddit.com, another active online forum, as a secondary source of data to examine the robustness of our findings. This data source contains the posting history of Reddit, and it is maintained on Google Bigquery (Reddit, 2016). Since Reddit data did not come with post view counts, we started with all Reddit users who posted at least once within the 52 weeks before September 18, 2013 in the subreddits related to iOS and Android app development.[20] We tracked users' posting activities using Reddit data within the same 21-week timeframe as the StackOverflow sample. Like the construction of the StackOverflow sample above, we retained only those developers within the top 10% of active posters based on posting frequency on Reddit. Unlike the StackOverflow data, comprehensive technical tag information and other time-varying control variables were unavailable from the Reddit data. Hence, with the Reddit sample, we focused on the robustness check of the main effect. As shown in Column 4 of Table 4, we continued to find qualitatively consistent evidence.[21]

6 | DISCUSSION

6.1 | Research contributions

This study makes three contributions to platform strategy research. First, our study contributes to research on two-sided platforms by incorporating foundational strategy research on coopetition. Using the coopetition framework, this study deepens existing understanding of complementors' competition and cooperation incentives in a platform context. Although both incentives are crucial to complementors' value creation and platform success (McIntyre & Srinivasan, 2017), extant research often treats these incentives separately (c.f., Hannah & Eisenhardt, 2018). For example, researchers consider Amazon's initial cooperation with third-party sellers and subsequent competition with them to be separate incentives (Wen & Zhu, 2019). However, coopetition literature highlights that agents' competitive and cooperative incentives may exist simultaneously and represent an inherent tradeoff (Khanna, 1998). Hence, by theorizing complementors' competitive and cooperative incentives as a dialectic tension, this study offers a new perspective to consider complementors' coopetition dynamics.

Second, this study addresses a gap in the platform governance literature—the implications of platform governance for interactions among complementors. Extant research has paid extensive attention to the effects of platform governance policies on the interactions between

[20]For iOS, we used the subreddit *iOSProgramming*, a major and comprehensive subreddit for iOS app development related discussions. According to its community description, this is "a subreddit to share articles, code samples, open source projects and anything else related to iOS, watchOS or tvOS development. Swift or Objective-C." For Android, we used the subreddit *androiddev*, which is described as "News for Android developers—Thoughtful, informative articles—Insightful talks and presentations—Useful libraries—Handy tools—Open source applications for studying." By focusing on development related subreddits, we may better filter general discussions unrelated to development.

[21]We performed two other robustness tests. First, we reexamined our main hypothesis using the eight, nine, and eleven weeks before and after the iOS 7 jailbreak as alternative timeframes, finding consistent results. Second, we excluded iOS 6 related discussions. Specifically, we first identified 28 questions marked with the iOS 6 tag from six developers in the sample and then removed all discussions posted by these six developers, reran the main analysis, and found consistent results.

TABLE 4 Robustness test: Alternative sample constructions

	StackOverflow sample: Windows as control group			Reddit sample
	(1)	(2)	(3)	(4)
Variables	Post count	Post count	Post count	Post count
Question view count	0.000	0.000	0.000	
	(0.000)	(0.000)	(0.000)	
Question comment count	−0.016	−0.001	−0.019	
	(0.011)	(0.010)	(0.011)	
Edit activity	−0.024	−0.030	−0.028	
	(0.012)	(0.012)	(0.013)	
Comment activity	0.068	0.059	0.069	
	(0.011)	(0.009)	(0.011)	
Add-favorite activity	0.095	0.086	0.099	
	(0.021)	(0.021)	(0.021)	
Non-app post count	0.736	0.729	0.727	
	(0.024)	(0.022)	(0.024)	
Non-app post quality	0.467	0.485	0.569	
	(0.086)	(0.086)	(0.113)	
DD (iOS*after)	−0.137	−0.187	0.139	−0.051
	(0.061)	(0.091)	(0.111)	(0.025)
After*complexity		0.045		
		(0.063)		
DD*complexity		0.299		
		(0.178)		
After*routineness			0.044	
			(0.019)	
DD*routineness			−0.068	
			(0.022)	
Constant	−0.599	−0.564	−0.587	0.375
	(0.098)	(0.080)	(0.102)	(0.028)
Observations	13,377	13,377	13,377	10,815
R-squared	0.644	0.648	0.649	0.202
Number of users	637	637	637	515
Week FE	Yes	Yes	Yes	Yes
Individual FE	Yes	Yes	Yes	Yes

Note: This table reports results of robustness checks using an alternative control group and data source, respectively. Columns 1–3 report results using Windows Phone OS app developers as an alternative control group, based on the StackOverflow sample. Column 4 reports results using Reddit as an alternative data source. Model specifications are the same as the main analysis in Table 2 except that the comprehensive technical tag information and time-varying control variables are unavailable for the Reddit sample. Standard errors clustered on the individual level are reported in parentheses.

complementors and platforms (e.g., developers' software release on a platform) because such interactions are associated directly with platform value creation (Kretschmer & Claussen, 2016; Rietveld et al., 2019). However, interactions among complementors may be equally important to value creation because such interactions can trigger novel knowledge recombination by unlocking new interdependencies among knowledge embedded in complementors. As a result, Boudreau and Jeppesen (2015) suggest that platform owners should consider management interventions to grow social interactions among complementors. By demonstrating that platform gatekeeping affects complementors' knowledge sharing, this study offers new ideas regarding how platform governance shapes complementors' social interactions, identifying a new direction for future platform governance research.

Third, our study informs extant research on platform access in two ways. First, we highlight an additional consequence of relaxed access control on innovation platforms. Extant research suggests that granting platform access to imitating complementors may intensify competition and discourage incumbent complementors' innovation (Boudreau & Jeppesen, 2015). Our results suggest another consequence—a drop in the incumbent complementors' knowledge sharing. Since knowledge sharing is critical to innovation, platform owners may find it important to design and enforce platform access policies carefully to limit imitation and encourage complementors' knowledge sharing and innovation initiatives. This view is consistent with the idea that platform owners should encourage complement innovations in less crowded domains (Gawer, 2011). Second, we echo recent calls to examine how complementors' heterogeneous characteristics influence their decisions and activities in a platform context (McIntyre & Srinivasan, 2017). Our study shows that the aforementioned drop in knowledge sharing is contingent upon two complementors' characteristics—knowledge complexity and routineness. The finding highlights the importance of considering complementors' heterogeneous characteristics to reveal boundary conditions for platform access control.

6.2 | Practical implications

Our research has several implications for practice. First, our findings suggest that managing complementors' competitive and cooperative incentives and shaping their knowledge sharing through platform governance is important to innovation platforms. Intense competition may discourage complementors from sharing knowledge, and overemphasizing competition may lead complementors to knowledge silos, fragmented innovations, and wasted efforts to reinvent the wheel (Wareham et al., 2014). When designing, deploying, and enforcing governance policies, platform owners should therefore consider possible impacts on complementors' competitive and cooperative incentives. One way to study these latent incentives, according to our research, is to track and analyze complementors' knowledge sharing activities.

Second, our findings suggest that owners of innovation platforms might consider instituting a stratified platform access policy to shape complementors' competitive and cooperative incentives. Similar to the idea of selective promotion of complementors (Rietveld et al., 2019), platform owners may dynamically adjust the criteria or enforcement strength of gatekeeping for sub-markets of their platform. For example, when a sub-market (e.g., gaming apps category) is overly crowded with substitutive complements, platform owners could tighten platform access to reduce the number of complementors, mitigating competitive incentives and encouraging cooperative interactions among complementors.

Finally, we find that an unrestricted platform access policy that attracts imitating complementors may lead to less active knowledge exchange among complementors. For complementors who are concerned about knowledge misappropriation and appreciate the benefits of reciprocal knowledge sharing, the finding suggests that they should prioritize platforms with well-designed and strictly enforced platform access policies that discourage imitating complementors.

6.3 | Limitations and future research

The contributions of this study need to be considered in light of its limitations. Although we found that an exogenous lapse of gatekeeping impairs complementors' knowledge sharing after controlling for complementor fixed effects and many covariates, we acknowledge that complementors' activities may also be shaped by other environmental factors, such as community cultures or etiquette on online forums (Faraj & Johnson, 2011). This suggests fruitful opportunities to examine the effect of community-related motivations on complementors' knowledge sharing. In addition, while we recognized complementor heterogeneity by studying the moderating role of app develops' knowledge complexity and routineness, other characteristics of app developers may also affect their behavioral patterns. Future work should therefore examine other characteristics of complementors in this context. Finally, although the literature suggests that complementors' knowledge sharing relates positively to the effectiveness or attractiveness of a platform, our data did not allow a direct analysis of platform effectiveness as a dependent variable. Future research that links complementors' interactions such as knowledge sharing to platform effectiveness will be particularly valuable.

7 | CONCLUSION

This study extends the coopetition framework to the platform context to examine how platform governance affects interactions among complementors. Our analysis focuses on platform access control in the form of gatekeeping as a driver of complementors' coopetitive incentives that shape their knowledge sharing. By exploiting the jailbreak of iOS 7 as an exogenous shock to Apple's gatekeeping policy, we show that a lapse in gatekeeping causes a drop in iOS app developers' knowledge sharing, and that this effect varies across developers, depending on characteristics of their knowledge base. Our arguments and findings, therefore, suggest that the strategic design and deployment of platform governance policies can be a valuable instrument for platform owners to orchestrate complementors' interactions such as knowledge sharing. As platforms become increasingly prominent in the digital economy, we hope that this study will stimulate more future research on the relationship between platform governance, complementors' value creation activities, and platform effectiveness.

ACKNOWLEDGEMENT

We would like to thank Peter Gray, Shane Greenstein, Wesley Koo, Steve Johnson, Sharon Matusik, Jeff Reuer, PK Toh, Carmen Weigelt, Jeff York, and Feng Zhu for their constructive comments on the development of this paper. An earlier version of the paper was presented at the *SMJ* Special Issue Conference on Platform Ecosystems and the 2018 NBER Summer Institute on IT and Digitization, and we appreciate the helpful comments provided by the participants. We acknowledge the detailed guidance and comments offered to us by the *SMJ* Special Issue editors and reviewers.

ORCID

Yuchen Zhang https://orcid.org/0000-0003-0220-1757
Tony W. Tong https://orcid.org/0000-0001-6760-6101

REFERENCES

Abdalkareem, R., Shihab, E., & Rilling, J. (2017). On code reuse from StackOverflow: An exploratory study on Android apps. *Information and Software Technology*, *88*, 148–158.

Adner, R. (2017). Ecosystem as structure: An actionable construct for strategy. *Journal of Management*, *43*(1), 39–58.

Adner, R., & 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), 306–333.

Allison, P., & Waterman, R. (2002). Fixed–effects negative binomial regression models. *Sociological Methodology*, *32*(1), 247–265.

Angrist, J. D., & Pischke, J. (2009). *Mostly harmless econometrics: An Empiricist's companion*, Princeton: Princeton University Press.

Anthony, S. (2013). iOS 7 jailbreak for iPhone and iPad: When, if ever, will the jailbreak be ready?.

Aral, S., & Van Alstyne, M. (2011). The diversity-bandwidth trade-off. *American Journal of Sociology*, *117*(1), 90–171.

Arora, A., Caulkins, J. P., & Telang, R. (2006). Research note – Sell first, fix later: Impact of patching on software quality. *Management Science*, *52*(3), 465–471.

Beattie, G., Durante, R., Knight, B., & Sen, A. (2019). *Advertising spending and media bias: Evidence from news coverage of car safety recalls* (NBER Working Paper No. 23940).

Belleflamme, P., & Peitz, M. (2019). Managing competition on a two-sided platform. *Journal of Economics & Management Strategy*, *28*(1), 5–22.

Bertrand, M., & Morse, A. (2011). Information disclosure, cognitive biases, and payday borrowing. *The Journal of Finance*, *66*(6), 1865–1893.

Birkinshaw, J., Nobel, R., & Ridderstrale, J. (2002). Knowledge as a contingency variable: Do the characteristics of knowledge predict organization structure? *Organization Science*, *13*(3), 274–289.

Blackwell, M., Iacus, S., King, G., & Porro, G. (2009). cem: Coarsened exact matching in Stata. *Stata Journal*, *9*(4), 524–546.

Boudreau, K. J. (2010). Open platform strategies and innovation: Granting access vs. devolving control. *Management Science*, *56*(10), 1849–1872.

Boudreau, K. J. (2012). Let a thousand flowers bloom? An early look at large numbers of software app developers and patterns of innovation. *Organization Science*, *23*(5), 1409–1427.

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

Cameron, A., & Trivedi, P. (1998). *Regression analysis of count data*, Cambridge: Cambridge University Press.

Casadesus-Masanell, R., & Halaburda, H. (2014). When does a platform create value by limiting choice? *Journal of Economics & Management Strategy*, *23*(2), 259–293.

Cennamo, C., Ozalp, H., & Kretschmer, T. (2018). Platform architecture and quality trade-offs of multihoming complements. *Information Systems Research*, *29*(2), 461–478.

Cennamo, C., & Santalo, J. (2013). Platform competition: Strategic trade-offs in platform markets. *Strategic Management Journal*, *34*(11), 1331–1350.

Claussen, J., Kretschmer, T., & Mayrhofer, P. (2013). The effects of rewarding user engagement: The case of Facebook apps. *Information Systems Research*, *24*(1), 186–200.

Constantinides, P., Henfridsson, O., & Parker, G. G. (2018). Platforms and infrastructures in the digital age. *Information Systems Research*, *29*(2), 381–400.

Cusumano, M. A. (2004). *The business of software: What every manager, programmer, and entrepreneur must know to thrive and survive in good times and bad*, New York, NY: Simon and Schuster.

Faraj, S., Jarvenpaa, S. L., & Majchrzak, A. (2011). Knowledge collaboration in online communities. *Organization Science*, *22*(5), 1224–1239.

Faraj, S., & Johnson, S. L. (2011). Network exchange patterns in online communities. *Organization Science*, *22*(6), 1464–1480.

Fleming, L. (2001). Recombinant uncertainty in technological search. *Management Science*, *47*(1), 117–132.

Fleming, L., & Sorenson, O. (2001). Technology as a complex adaptive system: Evidence from patent data. *Research Policy*, *30*(7), 1019–1039.

Gawer, A. (2011). *Platforms, markets and innovation*, Cheltenham: Edward Elgar.

Geng, X. J., & Lee, Y. J. (2013). Competing with piracy: A multichannel sequential search approach. *Journal of Management Information Systems*, *30*(2), 159–184.

Gil, R., & Warzynski, F. (2015). Vertical integration, exclusivity, and game sales performance in the US video game industry. *Journal of Law Economics & Organization*, *31*, 143–168.

Gnyawali, D. R., & Park, B. J. (2011). Co-opetition between giants: Collaboration with competitors for technological innovation. *Research Policy*, *40*(5), 650–663.

Grant, R. M. (1996). Toward a knowledge-based theory of the firm. *Strategic Management Journal*, *17*, 109–122.

Greenstein, S., & Nagle, F. (2014). Digital dark matter and the economic contribution of Apache. *Research Policy*, *43*(4), 623–631.

Halaburda, H., Piskorski, M. J., & Yildirim, P. (2018). Competing by restricting choice: The case of matching platforms. *Management Science*, *64*(8), 3574–3594.

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

Hoffmann, W., Lavie, D., Reuer, J. J., & Shipilov, A. (2018). The interplay of competition and cooperation. *Strategic Management Journal*, *39*(12), 3033–3052.

Iacus, S. M., King, G., & Porro, G. (2012). Causal inference without balance checking: Coarsened exact matching. *Political Analysis*, *20*(1), 1–24.

Jacobides, M. G., Cennamo, C., & Gawer, A. (2018). Towards a theory of ecosystems. *Strategic Management Journal*, *39*(8), 2255–2276.

Jehn, K. A. (1995). A multimethod examination of the benefits and detriments of intragroup conflict. *Administrative Science Quarterly*, *40*(2), 256–282.

Jiang, Z. R., Sarkar, S., & Jacob, V. S. (2012). Postrelease testing and software release policy for enterprise-level systems. *Information Systems Research*, *23*(3), 635–657.

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.

Kauffman, S. (1993). *The origins of order*. New York, NY: Oxford University Press.

Khanna, T. (1998). The scope of alliances. *Organization Science*, *9*(3), 340–355.

Khanna, T., Gulati, R., & Nohria, N. (1998). The dynamics of learning alliances: Competition, cooperation, and relative scope. *Strategic Management Journal*, *19*(3), 193–210.

Koetsier, J. (2017). App Developers Losing $3–4 Billion Annually Thanks To 14 Billion Pirated Apps. In *Forbes*.

Kretschmer, T., & Claussen, J. (2016). Generational transitions in platform markets – The role of backward compatibility. *Strategy Science*, *1*(2), 90–104.

Kuk, G. (2006). Strategic interaction and knowledge sharing in the KDE developer mailing list. *Management Science*, *52*(7), 1031–1042.

Lahiri, A., & Dey, D. (2013). Effects of piracy on quality of information goods. *Management Science*, *59*(1), 245–264.

Larsson, R., Bengtsson, L., Henriksson, K., & Sparks, J. (1998). The interorganizational learning dilemma: Collective knowledge development in strategic alliances. *Organization Science*, *9*(3), 285–305.

Lavie, D. (2006). The competitive advantage of interconnected firms: An extension of the resource-based view. *Academy of Management Review*, *31*(3), 638–658.

Leiponen, A., & Helfat, C. E. (2010). Innovation objectives, knowledge sources, and the benefits of breadth. *Strategic Management Journal*, *31*, 224–236.

McIntyre, D. P., & Srinivasan, A. (2017). Networks, platforms, and strategy: Emerging views and next steps. *Strategic Management Journal*, *38*(1), 141–160.

Parker, G., & Van Alstyne, M. (2018). Innovation, openness, and platform control. *Management Science*, *64*(7), 3015–3032.

Rasch, A., & Wenzel, T. (2013). Piracy in a two-sided software market. *Journal of Economic Behavior & Organization*, *88*, 78–89.

Reddit. (2016). 1.7 billion reddit comments loaded on BigQuery.

Rietveld, J., Schilling, M. A., & Bellavitis, C. (2019). Platform strategy: Managing ecosystem value through selective promotion of complements. *Organization Science*, *30*(6), 1232–1251.

Rivkin, J. W. (2001). Reproducing knowledge: Replication without imitation at moderate complexity. *Organization Science*, *12*(3), 274–293.

Roberts, J. A., Hann, I. H., & Slaughter, S. A. (2006). Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the apache projects. *Management Science*, *52*(7), 984–999.

Simon, H. A. (1996). *The architecture of complexity*. Cambridge, MA: MIT Press.

Sleptsov, A., Anand, J., & Vasudeva, G. (2013). Relational configurations with information intermediaries: The effect of firm-investment bank ties on expected acquisition performance. *Strategic Management Journal*, *34*(8), 957–977.

Stavrianou, A., Andritsos, P., & Nicoloyannis, N. (2007). Overview and semantic issues of text mining. *Sigmod Record*, *36*(3), 23–34.

Sun, M., & Zhu, F. (2013). Ad revenue and content commercialization: Evidence from blogs. *Management Science*, *59*(10), 2314–2331.

Tiwana, A. (2013). *Platform ecosystems: Aligning architecture. Governance, and strategy*, Waltham: Morgan Kauffmann.

Tsai, W. P. (2002). Social structure of "coopetition" within a multiunit organization: Coordination, competition, and intraorganizational knowledge sharing. *Organization Science*, *13*(2), 179–190.

Vasilescu, B., Filkov, V., & Serebrenik, A.. (2013). *StackOverflow and GitHub: Associations between software development and crowdsourced knowledge*. 2013 Ase/Ieee International Conference on Social Computing (Socialcom): 188–195.

von Krogh, G., & von Hippel, E. (2006). The promise of research on open source software. *Management Science*, *52*(7), 975–983.

Wang, Q., Li, B. B., & Singh, P. V. (2018). Copycats vs. original mobile apps: A machine learning copycat-detection method and empirical analysis. *Information Systems Research*, *29*(2), 273–291.

Wareham, J., Fox, P. B., & Giner, J. L. C. (2014). Technology ecosystem governance. *Organization Science*, *25*(4), 1195–1215.

Wen, W., & Zhu, F. (2019). Threat of platform owner entry and complementor responses: Evidence from the mobile app market. *Strategic Management Journal*, *40*(9), 1336–1367.

Wong, S. S. (2004). Distal and local group learning: Performance trade-offs and tensions. *Organization Science*, *15*(6), 645–656.

Wooldridge, J. M. (2010). *Econometric analysis of cross section and panel data*, Cambridge: MIT Press.

Younge, K. A., Tong, T. W., & Fleming, L. (2015). How anticipated employee mobility affects acquisition likelihood: Evidence from a natural experiment. *Strategic Management Journal*, *36*(5), 686–708.

Zander, U., & Kogut, B. (1995). Knowledge and the speed of the transfer and imitation of organizational capabilities: An empirical test. *Organization Science*, *6*(1), 76–92.

Zhang, Y., & Tong, T. W. (2020). How vertical integration affects firm innovation: Quasi-experimental evidence. *Organization Science*.

SUPPORTING INFORMATION

Additional supporting information may be found online in the Supporting Information section at the end of this article.

How to cite this article: Zhang Y, Li J, Tong TW. Platform governance matters: How platform gatekeeping affects knowledge sharing among complementors. *Strat Mgmt J*. 2022;43:599–626. https://doi.org/10.1002/smj.3191