From invisible hand to visible hand: platform governance and institutional logic of independent Mac developers, 2001-2012

Completed Research Paper

Yixin Qiu
Robert H. Smith School of Business, University of Maryland
joqiu1999@gmail.com

Il-Horn Hann
Robert H. Smith School of Business, University of Maryland
ihann@rhsmith.umd.edu

Anandasivam Gopal
Robert H. Smith School of Business, University of Maryland
agopal@rhsmith.umd.edu

Abstract

Amongst the research of institutional logics, field-level logics have continuously gained interest among scholars. A cultural emergence model of field-level logics was proposed in the latest development of the institutional logics perspective (Thornton et al. 2012). This study aims to validate a section of the model: the relationship between resource environment and emergence and evolution of field-level logics, and do so in the context of Apple’s independent Mac OS X developers – Mac indies. Through a qualitative interpretive study, and a combination of narrative and content analysis, we identify the resource environment of Mac indies: platform governance and developers’ own economy, and examine a critical platform governance change from Apple – its role shift from being primarily a technological platform to one that includes a market exchange, and show that a software ecosystem logic prevailed for Mac developers prior to the change, and a platform ecosystem logic emerged after that.

Keywords: Institutional theory, platform ecosystem, qualitative research
Introduction

During the past two decades, the theory of institutional logics has undergone substantial theoretical and empirical development. Broadly conceptualized as the “socially constructed organizing principles that shape individual preferences and organizational interests as well as the repertoire of behaviors by which interests and preferences are attained” (Friedland and Alford 1991: 232), institutional logic has been instrumental in explaining the heterogeneity in organizational decisions and routines (Thornton and Ocasio 1999, Thornton 2002, 2004, Lounsbury 2002, 2007), contestation and resistance behavior at organizational and individual levels (Townley 1997, 2002, Reay and Hinings 2005, Marquis and Lounsbury 2007, Qiu, Gopal and Hann 2011), and change and evolution of organizational fields (Nigam and Ocasio 2010, van Gestel and Hillebrand 2011).

Friedland et al. (1991) argued that institutional logics exist at multiple levels in a nested manner, and following on this reasoning, Thornton and her colleagues developed the concept of institutional logic at the level of the industry or field (Thornton and Ocasio 1999, 2008, Thornton 2002, 2004). In a more recent advancement, Thornton, Ocasio and Lounsbury (2012) articulate a cultural emergence model of formation of field-level institutional logics, and explicate the mechanisms through which cross-level effects operate. Adopting a linguistic approach, the model illustrates how societal level and external logics as well as resource environment contribute to forming field-level logics through shaping both the material practices and symbolic representations in the field. While the theoretical relationships and processes are documented in the model, which will be explained in detail later, further empirical work is needed to validate elements of this model. In addition to logic emergence, the model also theorizes about the change and evolution of the field-level institutional logic wherein the authors urge that great care must be exercised in future research in expressing what aspects of the field-level practices are changing and how changing practices reflect changing symbolic meanings and institutional logic (Thornton et al. 2012: p169). In this study, we examine exactly this gap in the literature.

The objective of this study is to investigate the effect of resource environment on the emergence and change of field-level institutional logics in the field of consumer software industry, and particularly of Apple's desktop platform and its independent third-party developers. The resource environment, according to Thornton et al. (2012), can include market and other forms of governance, such as governments, corporations, and information networks (p.157). Third-party developers constitute a software platform's key external innovative assets since they build complementary applications based on platform's core technologies (Tiwana, Konsynski and Bush 2010, Gawer 2010). Most software platforms exercise governance and control over developers through technological and non-technological means (Tiwana et al. 2010). Thus, the platform constitutes a key resource environment for such developers. At the same time, developers operate in a free market and create their own economy together with other market participants on the platform. This market is influenced by the platform in terms of its installed base and customer characteristics; yet it is still a self-contained economy with its own internal coherence. The market conditions therefore form the second part of the resource environment for developers. How then do resource environments lead to the emergence of a field-level logic of independent Mac developers? This is the first research question we address in this work. In addition to the formulation of institutional logics, literature has documented the effect of change in resource environments, such as environmental jolts, shocks, or critical events on institutional change (Nigam et al. 2010, Sine and David 2003, Hoffman 1999). For third-party developers, changes in their resource environment include direct changes implemented by the platform, such as changes in platform governance, as well as indirect changes, which may be affected by the platform governance change and occur in developers’ own economy. This leads to the second research question we study – how do changes in resource environments impact third-party developers’ institutional logic? We examine a critical platform governance change from Apple – the opening of the iOS App Store and subsequently, the Mac App Store, and hence the shift in Apple’s role from being primarily a technological platform to one that includes a market exchange. We study how the content and composition of Mac developers’ institutional logic changes from before to after these changes in their resource environments.

This research first of all contributes to a deeper understanding of the emergence and evolution of institutional logics in the field of consumer software industry. This research also contributes to the software platform governance and software ecosystem literature by emphasizing the role of third-party developers. Through an institutional field conceptualization, we argue that as developers write software
for a platform, they constantly interpret and make meanings of governance mechanisms from the platform, which explains their subsequent practices and strategies. Our study hence adds a symbolic and cultural lens to the software platform ecosystem literature, which currently is composed mostly of perspectives on architecture-modular design (Baldwin and Clark 2000, Baldwin and Woodard 2009), economic explanations (Katz and Shapiro, Rochet and Tirole 2003, 2006, Parker and Van Alstyne 2005, 2012, Eisenmann, Parker and Van Alstyne 2009), and organizational considerations (Gawer and Henderson 2007, Cusumano and Gawer 2002). Apple’s context is uniquely appropriate for the research questions in the study. In recent years, the consumer IT industry witnesses a growing number of platforms adopting the “App Store” model, such as in software (mobile and desktop), social networking, web browsing and e-publishing. We aim to use Apple’s ecosystem as a starting point to study the impact of platform’s governance of technology and market distribution on third-party developers.

Theoretical background

Emergence of field-level institutional logics

Building on the seminal work by Friedland and Alford (1991), Thornton and Ocasio (1999) define institutional logic as “the socially constructed, historical patterns of cultural symbols and material practices, including assumptions, values, and beliefs, by which individuals and organizations provide meaning to their daily activity, organize time and space, and reproduce their lives and experiences.” (P804). Over the years, Thornton and her colleagues have developed institutional logic at the level of field (e.g. Thornton et al. 1999, Thornton 2002, 2004). Their notion of the field encompasses the boundary of an industry or a profession, and they argue for the influence of societal-level logic on the formation of field logic by positing that “field-level logics are both embedded in societal-level logics and subject to field-level processes that generate distinct forms of instantiation, variation, and combination of societal logics” (Thornton et al. 2012: p148). For instance, the fiduciary logic in public accounting is a hybridization of logic of the profession and religion; the aesthetic logic in architecture a hybrid of professional and market logic, and the editorial logic in higher-education publishing a variant of professional logic (Thornton, Jones and Kury 2005). These field-level logics are instantiations of societal-level logics, and the specific historical, cultural and material contingencies in the field lead to field-specific variations in practices (Thornton et al. 2012: p149).

In more recent work on the cultural emergence model on field-level logics mentioned above, Thornton et al. (2012) delineate the mechanisms through which these cross-level effects operate on specific field-level logics. Building on the premise that institutional logics are both symbolic and material (Friedland et al. 1991), they expand it by taking a linguistic turn to explain the construction of field-level logics. Because institutional logics reflect cognitive, normative and material forces (Thornton et al. 2012), they are embodied in the vocabularies and communication of members of social groups (Loewenstein and Ocasio 2009). As narratives create new systems of categories that link category labels to field-level organizing practices (Thornton et al. 2012:159), or change meanings of existing categories (Ruef 1999), distinct institutional logics emerge. Their theoretical model is shown in Figure 1a and a brief summary of the constructs and the processes in the model is as follows. The authors argue that societal logics, or external logics, defined as “the institutional logics developed in other institutional fields”, are building blocks for the formation of field-level institutional logics. Providing opportunities and constraints for field-level practices, the resource environment affects the emergence of field-level institutional logics through material forces, as well as cognitive, cultural and political factors. Vocabularies of practice, defined as “systems of labeled categories used by members of a social collective to make sense of and construct organizing practices”, provide a critical linchpin which brings together symbolic representations, in the form of theories, frames and narratives and practices in formulating field-level institutional logics (Thornton et al. 2012: 150-161). An important part of this model pertains to how resource environments affect the development of field-level logics, and empirical work testing these linkages is still absent in the literature. This relationship, depicted in Figure 1b, is particularly examined in the current study.

Field level institutional logic change

In addition to logic emergence at the field level, institutional logic change has also continued to be of interest to scholars. In accord with the cultural emergence model by Thornton et al. (2012), exogenous
changes in societal logics and changes in the resource environment can both lead to evolution and change in field-level institutional logics. In examining financial intermediaries in the U.S., Lounsbury (2002) documented that the stable regulatory logic established in the 1930s, composed of government legislation and rules, was replaced by the market logic, reflected in retail-orientation and competitiveness, due to a deregulation act in 1980. He argued that this industry-level logic shift may be better viewed as an outcome of the general cultural shift from a regulatory to a market logic that unfolded gradually over the period since midcentury in the society (p257). New government policies are another frequently-theorized form of changing resource environment and their impact on existing logics is discussed. For instance, researchers document responses from universities and museums to a new government policy of business and performance measure (Townley 1997, 2002), and physicians’ responses to provincial government’s structural change to the healthcare system in Alberta (Reay and Hinings 2005, 2009). These new policies challenge the incumbent logics with their distinct symbolic meanings and rationalities and bring forth new status in field-level logics, such as logic shift or co-existence. In this study, we examine the governance policies in the consumer software industry and the effect of their change on the logics in the industry.

Structure of software industry and third-party developers’ resource environment

Because the focus of this study is institutional logic and the role of resource environment in the field of consumer software industry, we review elements constituting developers’ resource environment. Highly tied to the structure of the software industry, the first component of the resource environment is software platform governance, defined as “who makes what decisions about a platform” (Tiwana et al. p679). A typical example of two-sided networks, the software platform aims to bring both sides, developers and users, on board and grow the ecosystem through two-sided network externalities (Rochet and Tirole 2003, 2006, Cusumano et al. 2002, Cusumano 2010, Gawer 2010). Developers rely on a platform’s services such as Software Development Kit (SDK) and Application Programming Interface (APIs) in order to obtain access to the hardware and operating system and write complementary application or services (Evans and Schmalensee 2007, Ghazawneh and Henfridsson 2011). These services form the foundation for a platform’s technology-based governance over third-party developers, a classic consideration of which is the level of intellectual property openness on a platform’s success (West 2003, Boudreau 2010, Eisenmann et al. 2009).

At the same time, third-party developers operate in a market environment with typical economic conditions, where strategic behaviors are desired. Boudreau (2012) demonstrates that increased number of developers tend to reduce innovation incentives in the PDA market, hence an evidence of the crowding-out effect in a competitive market. Through a study of enterprise software industry, Huang, Cuccagnoli, Forman and Wu (2013) show that third-party developers with a greater stock of formal intellectual property rights (such as patents and copyrights), and those with stronger downstream capabilities (as measured by trademarks and consulting services) are more likely to protect themselves against the threat of platform expropriation. In studying relationship between first mover advantage and environmental characteristics in the iPhone app market, Srinivasan and Suarez (2009) show that early entrants outperform late entrants, and entry timing is more important in higher growth rate genres and for incumbent rather than new developers. These findings suggest that there is a second component of the developers’ resource environment, which is formed by developers’ market environment. Furthermore, these two parts of the resource environment are interconnected – the platform’s policies and market performance can influence third-party developers’ market dynamics, competitor types, and customer base. Therefore, it would be more precise to say that developers operate in a semiautonomous economy. To the best of our knowledge, no studies have discussed the effect of platform governance and developers’ own economy at the same time, thus identifying these would be a contribution of this research.

Methodology

Research context

To examine the questions on consumer software industry’s institutional logic and the impact of resource environment, we chose to study independent Mac OS X developers: Mac indies, and their relationship with Apple during the 2001 to 2012 time span. 2001 marked the year when Mac OS X, Apple’s new
desktop operating system was introduced to the world after founder Steve Jobs returned to Apple. The time also coincided with the dot-com bust, which left many software developers unemployed, but also a relatively mature online payment infrastructure grown during the Internet bubble. The availability of a cool new technology, free development tools, and an online distribution channel sparked a wave of entrepreneurship where small or individual developers formed business by writing Mac apps and selling them through the Internet (Meeteren 2008). Consisting of a large portion of third-party developers for the Mac OS X platform, Mac indies share attributes with the shareware developers in terms of firm size (Hui, Yoo and Tam, 2008): many of Mac indies are individuals or small firms. Yet they also differ in one key dimension: instead of relying on a donation base as shareware developers do, Mac indies sell software as commercial products. As alluded to earlier, Apple’s change in its governance policies: its announcement of the iPhone App Store on March 6, 2008 (later the iOS App Store) and the Mac App Store (MAS) for its desktop applications on October 20, 2010, makes the platform a perfect context to study the research questions.

Data and analysis

Following the argument that narrative and vocabularies reflect the underlying process for the emergence and change of institutional logic (Thornton et al. 2012), and based on the role of historical research in analyzing field-level institutional logics (e.g., Thornton and Ocasio 1999, Thornton 2002, Lounsbury 2002, 2007, Marquis and Lounsbury 2007), we rely on publicly available online data sources of the Mac indie community, iOS-turned Mac developers, sources on Apple’s culture and policies, and existing studies on Mac and iOS developers for this research. We use qualitative approach with an interpretive philosophy (Klein and Myers 1999) to derive Mac indies’ institutional logics. Because institutional logic is concerned with social actors’ practices and meaning system, the interpretivism epistemology fits perfectly. Furthermore, we follow Nigam et al. (2010) by combining a narrative and content analysis in studying the research questions. Specifically, we derive ideal types of developers’ institutional logics before and after change in the resource environment, explicate characteristics of developers’ resource environment and use content analysis to quantitatively show the logic pattern over time. The analytic approach is summarized in the process chart in Figure 2.

Ideal types, established by Max Weber (1904), is a typological construct for theory building and modeling (Doty and Glick 1994), and they are the commonly used formal analytic models to compare empirical observations across institutional order; therefore, they are best developed at least in pairs, if not multiple characterizations (Thornton et al. 2008: 119). Ideal types of institutional logics are characterized by elemental categories, which represent the cultural symbols and material practices particular to a logic. These elemental categories are established social-science concepts, some of which are derived from Weber (1922/1978). In explicating the cultural emergence model of field-level institutional logics, Thornton et al. (2012) maintain that the key constructs in the model, such as symbolic representations, practices, and vocabularies of practice are all categorical elements of institutional logics (150). For the current study, we use ideal types to construct field-level logics for the industry we study - the consumer software industry.

We draw on data sources related to incumbent Mac developers, Mac culture and existing research on Mac developers (e.g. Meeteren 2008) to construct ideal types for the logic prior to platform’s governance change. While most data sources cover the period between 2004 and 2012, an ethnographic study on Mac indies by Meeteren (2008) and an oral history project on Mac culture provide data between 2001 and 2004. For ideal types for the new logic, we examine data sources related to incumbent Mac developers’ changes, iOS-turned Mac developers, and existing studies on iOS developers (e.g., Qiu et al. 2011, Meeteren 2009, Bergvall-Kåreborn et al. 2010). The assumption for the latter is that Mac developers’ new institutional logic is composed of iOS-turned Mac developers’ practices and belief systems, as well as incumbent Mac developers’ changes in response to both the iOS App Store and the Mac App Store (MAS).

Our first step was coding one of the data sources – discussions topics on the major Mac indie listserv: MacSB on Yahoo! Groups. We traced the discussion from its inception: 1/29/2004 to 12/31/2012, the end of our data collection, in a 9-year span. We divided listserv threads into 3 time periods in accord with Apple’s governance change to examine the temporal shift in developers’ attention and discussions. The first phase is from 1/29/2004 till 3/5/2008, the second phase is from 3/6/2008, when the iOS App Store was announced, till 10/19/2010; and the third-phase is between 10/20/2010, when the Mac App Store was announced, and 12/31/2012. We expect developers’ logic change to appear in phase 2, and blossom in
phase 3. We coded the first message in a given thread and used that to represent the subject discussed in that thread. We tried to mitigate the limitation of this approach with the large number of threads coded. In total 1,264 threads were coded. This number is comparable to that in existing IS literature which uses listserv as data source (c.f. Orlikowski and Yates, 1994, Kuk 2006, Kudaravalli and Faraj 2008).

To map the comprehensiveness of the elemental categories in ideal types, we adopt the stakeholder perspective (Donaldson and Preston 1995, Jones and Wicks 1999, Agle, Mitchell and Sonnenfeld 1999) as a general coding framework to identify developers’ practices and interpretations. A much cited definition of stakeholders is “those groups without whose support the organization would cease to exist” (Stanford Research Institute (SRI) 1963, quoted in Donaldson and Preston 1995: 72). Our use of the stakeholder perspective is mainly at the descriptive and empirical level, which suits exploration of the new areas (Donaldson et al. 1995: 70-71), such as our case. We also went a step further to analyzing the focal organizational entity itself – developers’ own identity and practices. Together, five categories emerged. Three of them are related to developers’ stakeholders: the platform, customers and infrastructure service providers. The rest two are market competition and developers’ entrepreneurial strategies, and developers’ identity and routine tasks. Figure 3 shows the stakeholder coding framework.

We combined the deductive and inductive coding approach in analyzing the listserv threads. For the deductive part, we relied on the platform governance literature and ethnography about Mac indies by Meeteren (2008) to code developers’ relationship with the platform and developers’ identity and drew on studies on iOS app developers (Qiu et al. 2011, Meeteren 2009, Bergvall-Kåreborn et al. 2010) to guide our coding on the new logic. We also allowed themes to emerge during the coding process. To start off coding, the first author sampled about 800 threads out of 1,264 across three phases to construct a coding scheme. To explicate the emergence of new logic, we used themes generated in the 1st phase as a baseline, and then carefully compared the meaning of developers’ messages in phase 2 and 3 with the existing themes. Following previous literature (e.g., Ruef 1999, Reay and Hinings 2005), which maintains that the new logic can reflect in the changing meaning of an existing theme or in a new theme, we formed a coding rule where any new meanings attached to the existing themes or brand new themes were classified as the new institutional logic attributed to the resource environment change. For instance, theme “platform’s entry into developers’ turf” means “market and product clash” before the change, while it means “distribution channel clash” afterwards. Theme “platform’s rules and regulations” mainly indicates legal rules regarding the appropriate use of the technology before the change, but means administrative, business and technical rules afterwards. It is worth pointing out that not all themes change meaning after the platform change strikes, and not all App Store-related discussions fall under the new logic. In addition, the original logic continued to exist in phase 2 and 3. This is because developers were either discussing apps released in their traditional outlets, or they were still following the original logic even for App Store apps. Based on the characteristics of the logics, the logic before the platform governance change was named “software ecosystem logic”, and the one after the change was named “platform ecosystem logic”. Due to space constraints, we do not include a full description of the themes developed through the coding but the complete coding scheme is available upon request from the authors.

After the first round was completed, the research team discussed the coding scheme, clarified different interpretations and adjusted the scheme. Then, the first author re-coded all threads one more time. The coding scheme continued to evolve until all threads were coded. To validate the coding framework, a research assistant coded 10% of the threads in each of the three phases. Cohen’s kappa was calculated for the inter-coder reliability, because it takes into account the agreement occurring by chance (Viera and Garrett 2005). The final Cohen’s kappa is 0.75, which indicates good agreement and above the threshold of 0.70 suggested for content analysis (Neuendorf 2002, Krippendorff 2004).

To triangulate the listserv data, we drew on supplemental online archival sources. Snowball and theoretical sampling were used to obtain these data (Miles and Huberman 1994, Strauss and Corbin 1998). We started with blog posts mentioned in the listserv discussions, blogs by established Mac or iOS developers in the community, and well-known industry press, and then expanded data sources from there. We then used theoretical sampling, where we selected blog posts from developers who differ in opinions, strategies and performances on the Mac App Store. We also searched data based on critical issues about platform’s policies which stirred heated debate among the developer community and the press, such as Apple’s issuance of the App Store review guidelines, Apple’s changing policies towards Adobe Flash, and Apple’s policies of Sandboxing and Gatekeeper. This search resulted in 77 additional...
pieces of text. These texts were coded using the same coding scheme derived from the listserv discussion. Finally, we mapped themes developed from the listserv discussion and supplemental texts, and existing studies on Mac and iOS developers to the elemental categories to construct the ideal types.

In addition to ideal type construction, we also used coding results from the listserv discussions to quantitatively demonstrate logic’s temporal shift. Due to data availability, while multiple sources about incumbent Mac developers, iOS-turned Mac developers and existing studies were used to construct the ideal types of the two logics, only listserv discussions were used to quantitatively capture the logic shift over time. Therefore in the content analysis, the new logic (platform ecosystem logic) only reflects incumbent Mac developers’ change, and does not include practices and belief systems of iOS-turned Mac developers. As mentioned earlier, based on the stakeholder coding framework, we derived five broad categories of developers’ stakeholder relationship and their identity and practices. We examine developers’ logic shift in two aspects: change of distribution of two logics in each category over time, and change of percentage of each category over time. For each phase in each of the five categories, we calculated frequency of each theme for the two logics, and we calculated the percentage score for the two logics in each phase. For instance if in phase 2 of any given category, the frequency for software ecosystem logic was 40, and that for platform ecosystem logic was 10, then software ecosystem logic constituted 80% of developers’ discussion topics in phase 2 for this category, and platform ecosystem logic constituted 20%. Over time from phase 1 to phase 3, we thus observed the changing distribution between the two logics for any given category due to resource environment change. We also calculated the percentage score of each category in each phase. For example, if 100 instances were coded for each of the 5 categories in phase 1, then each category equally constituted 20% of developers’ discussion topics in phase 1. Over time, the percentage score revealed the changing pattern of developers’ attention on their stakeholder relationship as well as their identity and strategies because of resource environment change.

Findings

Below we first discuss two ideal types of Mac indies’ institutional logics before and after platform’s governance change and their relationships with the resource environment, and then corroborate the logic change with results from the content analysis.

Ideal types of Mac developers’ institutional logics

Table 1 presents the ideal types of two logics. We integrate discussion of the resource environment with that of the institutional logic. In the end of the description of each logic, relationship between resource environment and institutional will be summarized.

Individual developers started to write and disseminate their software almost three decades ago. Developers distributed their software as “shareware”, initially through dial-up bulletin boards or via disks given away with computer magazines, and later via the Internet. The first piece of shareware was initially distributed free, but as maintenance and administrative costs increased, the author asked users to donate a small amount of money on a voluntary basis (Hui et al. 2008: 580). While making a living out of shareware had been hardly attainable under a rudimentary distribution and payment system (Takeyama 1994), the dream of writing commercial applications and selling directly to users was made possible at the turn of the 21st century, especially among Mac developers.

Gradually from 2001 to 2008, a software ecosystem logic was formed for Mac indies. This logic is a hybridized mix of the professional and market logic, and is guided by the personal and market capitalism. The market logic here is characterized by indie developers’ specialist position in the market through exploitation of periphery of the resource space, or the niche market (Carroll 1985, Carroll and Swaminathan 2000, Swaminathan 2001). In comparison, companies like Microsoft and Adobe, including the platform firm Apple itself produce generalist software, which target center of the market. Under the software ecosystem logic, developers’ identity is characterized by their awareness of being third-party complementors to a platform, and their value they confer on being an independent business owner. A popular analogy among the Mac indie community is that they are the “sharecropper” to Apple. Developers’ app sales are highly reliant on platform’s market share and installed base, and they face the risk of platform potentially entering their product turf. Developers would produce a power user version of Apple’s app in order to avoid market clash (Meeteren 2008), which essentially drives developers to
further target periphery of the market – hard-core Mac users, consumers with specialized needs or developers themselves.

Mac indie's identity is at the same time strongly defined by the independence aspect. Developers cherish the autonomy of being their own boss, having a big say over the business, and resisting the bureaucracy of working for a “BigCo” (developer presentation). They normally bootstrap the business without taking external funding, and command the freedom in choosing the kind of technology for writing apps. Sources of legitimacy come from quality of apps, reputation of developers and platform recognition. Mac indie’s emphasis on quality is highly influenced by platform Apple’s meticulous attention to detail on aesthetics and user interface design. Such artistic pursuit was passed down from the design philosophy of the original Macintosh. As Andy Hertzfeld, one of the original Macintosh team members recalled, “The Macintosh was driven by artistic values, oblivious to competition, where the goal was to be transcendentally brilliant and insanely great. We wanted the Macintosh to be a technical and artistic tour-de-force that pushed the state of the art in every conceivable dimension.”

Customers are also an important force in shaping app quality. Mac users, like developers themselves, tend to be immersed in the cultural meanings of Mac experience and have a sense of taste (Meeteren 2008). Their high bar for quality thus pushes developers to hone in the app experience even further. In the Mac community, famous developers are frequently quoted and their success stories are widely shared. Legitimacy also comes from platform’s recognition, such as Apple’s yearly Apple Design Awards (ADA) at yearly developer conference: WWDC. For Mac indies, the authority includes their skills and capabilities in producing apps with high engineering and design qualities, as well as the level of market acceptance towards an app. As a community of practice (Brown and Duguid 2001), developers form norms regarding how members should behave during the socializing process. Following the principle of reciprocity, a developer earns credit in the community by helping others that later can be “exchanged” if he or she needs help (Meeteren 2008). Mac indies pay particular attention to the etiquette in dealing with competitors in public. Developers have a general consensus that their products are differentiating and not competing with each other. Competition on price incentives is not encouraged and direct product imitation is frowned upon (Meeteren 2008).

When indies start the business, their mission is to do something they love, keep the business sustainable and increase sales. To do so, they focus their attention on resolving entrepreneurial challenges, implementing infrastructure services best suited for the business and adapting to platform’s system progress and technology change. Under the software ecosystem logic, developers are accountable for their relationship with stakeholders, namely, customers, infrastructure providers and the platform. Small company size allows Mac indies to add a personal touch in tech support. In the meantime, developers also try to balance the personal and professional side of the business. Indie business relies heavily on market service providers, such as app aggregators, e-commerce and payment providers, hosting services, update and publishing service providers. Under the software ecosystem logic, developers’ attention towards Apple is mainly centered around the impact of platform’s system change and upgrade on third-party development. For instance, developers need to decide between backward compatibility to support customers of old operating systems and embracing the new OS to leverage more platform features. The biggest technical change developers encountered during the observation period was Apple’s switching from IBM PowerPC chips to Intel chips in 2005. This caused huge uproar among the developer community: developers were concerned that Mac would lose its character and become just like any mass-consumed Wintel machines. A bigger impact however, was that developers had to convert their code to be compatible with the new system throughout the transition period.

Under the software ecosystem logic, basis of strategy is an organic growth model. Developers write apps targeting serious customers for long-term use. They maintain an average price range of $20 to $50 per piece of software, because developers attach great importance on price as it represents the value of their work. After the first version of an app is released, developers continue to fix bugs and enhance features to both strengthen the existing customer base and cultivate new ones.

The above ideal-type attributes of software ecosystem logic demonstrate that both societal-level professional and market logic are in effect here. Mac indie developers need to solve engineering and design problems and their legitimacy system rewards apps with high quality. At the same time, developers target specialist market and need to resolve entrepreneurial challenges. Under software ecosystem logic, Apple as resource environment provides the hardware, operating system, development tools and design
guidelines to developers. It also influences developers with its innovative spirit and artistic pride in software design. Resource environment within developers’ own economy comprises the Internet as a distribution channel, proliferation of infrastructure service providers and software technologies. Developers’ customer base is mainly composed of those who have the need for and knowledge about specialized third-party software, such as power users, Mac enthusiasts or other developers. This resource environment not only facilitates Mac indies to run viable software business, it also helps developers to build a strong identity of independence. Here, developers’ customer base is partially influenced by the potential threat of platform entering third-party developers’ product market, which indicates that the platform component of the resource environment influences the developers’ economy component of the resource environment. Under software ecosystem logic, platform’s impact on third-party developers’ market dynamics, such as market competition, and their market strategies is minimal, if not none.

The first substantial change in developers’ resource environment occurred on March 6, 2008. Along with the announcement of the iPhone SDK (software development kit), Apple also introduced the App Store, the exclusive market place to distribute the iPhone and later the iPad apps. This platform’s governance change marked the beginning of Apple as a technology platform entering the domain of software distribution, enacting the role as a market exchange owner. On October 20, 2010, Apple announced the Mac App Store (MAS), a similar market place as the iOS App Store but focused on desktop applications. Unlike the iOS App Store, the MAS is not the exclusive channel. These two changes were the harbinger of a new and differing developers’ logic – platform ecosystem logic, which we discuss below.

The platform ecosystem logic is a hybridization of corporate and market logic, and it is guided by the managerial and market capitalism. Here the corporate logic stems from platform’s technical and administrative policies for developers. The market logic is characterized by indie developers’ generalist position on the market, and this is strongly influenced by the design and attributes of a platform-controlled market place. Platform Apple has in three ways assisted Mac indie developers in taking on the role as a generalist without actually being one – indie developers are still of small-scale, but they are able to leverage center of the resource space as generalist firms do with the help of the platform. First, the App Store has legitimized and popularized software consumption through increasing average consumers’ knowledge about and demand for software. Shopping for software is no longer an activity only for power users; it becomes a mass market phenomenon. Because these new users are mostly average users, they tend to have mainstream preferences for apps as opposed to preferences that reflect power users or programmers. Therefore they form the center of the resource space which appeals to generalists. Secondly, the App Store creates a mass distribution channel for developers to reach a much larger user base that they were not able to in the past – the App Store essentially solves the scale issue critical for a firm to function as a generalist. Thirdly, the App Store design, especially its top charts feature, incentivizes developers to write mass-appeal apps and target center of the market space.

Under this logic, developers’ identity is that of subordinate third-party developer entrepreneur. Apple, as an exchange place owner, created administrative and technical rules that developers need to adhere to. From App Store developer enrollment status to app review process, Mac indies experienced unprecedented organizational bureaucracy in releasing an app to the market. Apple performs “centralized quality certification” (Hagiu 2009) through a review process. However, unlike game console companies, who filter out developers before a game is created, Apple rejects apps after they are created. In addition to the inconsistency of the review policies, Apple’s rules led to much frustration among developers. They feel that they cannot trust the platform and risk losing control over their apps and even their business. Furthermore, developers need to abide by Apple’s technology requirement for producing apps, be it programming languages or APIs. Many Mac indie’s existing applications cannot be sold through the App Store simply because they will violate the guidelines. Developers expressed their chagrin about their increased dependency on the platform: “Is this worth it? Nimble development has always been something I’ve taken pride in, but the App Store process is the very antithesis of nimble. It’s slow, bureaucratic, and opaque. And for what? Is the sales increase worth it?” “So, the big question is, will you see RA applications in the Mac App Store? Right now, we don’t know. With Apple’s onerous guidelines, most of our applications would not be approved. Even if they would be, however, are the benefits good enough to give up being a truly independent software developer?” (Listserv discussion)

Under platform ecosystem logic, developers’ legitimacy is derived from App Store chart ranking, platform recognition and end users’ reviews and ratings. Sources of authority stem from platform’s review and end
user interests. On the App Store, every app goes through a review process before being published, and as alluded to earlier, Apple employs its power in deciding app qualifications. Developers convey a sense of fear about Apple’s potential disapproval of apps, and they frequently exchange information with each other about the suitability of an app before submission. The other source for authority is end user interests. On the scale with consumers and developers on each side, Apple as a platform tilts the scale towards consumers. Many of Apple’s policies are designed to protect user experience and security, but they do not always benefit developers. For instance, Apple’s App Store review guidelines have detailed items on banning apps related to personal attacks, with objectionable content and pornography, and those offending religion, culture, and ethnicity. This is certainly commendable from a user’s standpoint, but Apple reviewers’ subjectivity and inconsistency often results in unfairness, and thus hurts developers’ interest. Under platform ecosystem logic, basis of norms is developers’ self-interest and dependency on the platform. Developers’ mission is to build competitive position of apps on top charts, and write apps to be favored by the platform. Some developers deliberately update apps according to platform’s OS upgrade cycle with a hope that their apps, which show off the latest platform technology, will be featured on the App Store.

Under platform ecosystem logic, developers’ basis of attention is standing out from competition, embracing the mass app users, and adapting to platform’s changing technological and administrative policies. Because of App Store’s design, competition appears much closer to each other than on the web. Therefore developers need to design app graphics and descriptions to quickly grab user’s attention. As noted earlier, the App Store has attracted a large number of users who have never made software purchase part of their lives. Unlike power users who are familiar with third-party software experience, these new users exhibit certain “immature” behaviors. For instance, they would complain about features that developers do not claim to have or blame developers for issues they are not liable for, etc. Learning to interact with average users is thus a new task for developers. Furthermore, the Mac App Store removed much flexibility that developers had enjoyed in app transactions on their web stores. For instance, the App Store licensing scheme makes sharing review and testing copies less easy; a commonly used time-limited demo approach by Mac indies is forbidden on the App Store; paid upgrades are not allowed on the App Store; to name just a few. To accommodate these changes, developers employ technical methods to address the licensing and app monetization challenges; and they try to use in-app purchase through modular app design or new app release to solve the no-paid-upgrade issue, although they both have limitations.

Under the platform ecosystem logic, developers’ basis of strategy is “hit” growth (Qiu et al. 2011). The “hit” here means striking app charts and constant new app releases. This is again mainly associated with design of the App Store and platform’s review process. Because ranking on top charts gives apps legitimacy and increases visibility, and sales volume determines chart position, developers tend to lower app prices and engage in pre-release buzz marketing in order to climb onto the chart during the initial launch. Writing apps with mass-appeal and modeling after or even mimicking store-trending apps is frequently pursued on the App Store, because this increases the chance for apps to hit the charts. In addition, since Apple evaluates apps after they are created, in order to minimize the risks of app rejection, developers release smaller-scale apps and save development cost. Consequently, developers diversify app portfolio with many small apps and accumulate profits from each of them.

The above ideal-type attributes of platform ecosystem logic exhibit the effect of societal-level corporate and market logics. The corporate logic is manifested in developers’ increased reliance on the platform for technology and business needs. The market logic is reflected in developers’ opportunities and incentives to function as a generalist company – in adoption of a “hit” strategy and embracing the mass users. Under platform ecosystem logic, Apple as the resource environment provides extensive rules and regulations on app qualification, licensing, monetization and versioning. These send a strong message of control and bureaucracy to Mac indie developers. Platform’s entry into developers’ distribution channel significantly alters the other part of developers’ resource environment – the market economy in which developers operate. Through bringing in large number of new users and developers, and a store design which favors frequent new app release, shorter development cycle and lower pricing, Apple creates a platform-controlled market environment, which denotes a very different kind of competition dynamics. These changes in the resource environment serve as a catalyst for incumbent Mac developers to redefine valuation of their software and labor, and their relationship with the platform.
Logic evolution of incumbent Mac indie developers: a stakeholder view

In the previous section, we illustrate and contrast the ideal types of software ecosystem logic and platform ecosystem logic along the elemental categories before and after change in the resource environment: namely, platform’s governance change and the subsequent change in developers’ economy. As noted earlier, Mac developers’ platform ecosystem logic is composed of iOS-turned Mac developers’ practices and belief systems, and incumbent Mac developers’ changes in practices and belief systems in response to both the iOS App Store and the Mac App Store (MAS). In this section, we only focus on the incumbent Mac developers, and demonstrate how their logics shifted in a temporal structure using results of the content analysis based on Mac developers’ listserv discussion. As explained in the methods section, we divided the listserv discussion into three phases. While instances of platform ecosystem logic are already present in phase 1, they began to take shape in phase 2 and flourish in phase 3. Recall that a stakeholder perspective was adopted in the listserv threads coding. Five categories emerged: developers’ relationship with the platform, customers and infrastructure service providers, developers’ entrepreneurial strategies, and developers’ identity and routine tasks. This approach allows us to observe which part of software logic changed meaning and morphed into platform logic due to platform’s governance change, and which part did not.

Table 2 shows the changing pattern of each of the five categories over time. With platform’s governance change, developers’ discussion topics related to the platform and customers have increased significantly. This is mainly because platform’s new review process, manifested in the sub-theme “Platform’s rules and regulations” under “Platform’s coordination with developers”, and platform’s policies on apps’ versioning, upgrade and licensing, reflected in the theme “Versioning and upgrade” and “Licensing” attracted much of developers’ attention. In contrast, developers’ discussions on infrastructure providers have reduced dramatically. This is not surprising, given most of the functionality is replaced with the App Store itself and developers’ dependency on them is decreasing. Developers’ attention on entrepreneurial strategies, their identity and business routines have roughly remained the same. An interesting observation is that from phase 1 to phase 3, developers’ discussions on market environment and entrepreneurial strategies, as well as their identity and routines consistently exceeded those on other categories. It indicates that how to compete on the market, how to appraise oneself and behave in a community, and how to operate the business has always been developers’ top focus. The third-place changed from infrastructure service-related discussions in phase 1 to platform relationship in phase 3. It suggests that as the platform is involved in third-party applications’ distribution, developers’ attention towards the platform increased substantially.

Table 3 exhibits the changing distribution of software and platform logics over time in five categories. Except for relationship with infrastructure providers, which does not entail platform logic, other categories all witness a migration in discussion topics from software logic to platform logic due to platform’s governance change. Interestingly, in categories on relationship with the platform and customers, platform logic-related discussion significantly surpassed that on software logic and became dominant in developers’ discussion in phase 3. In contrast, software logic still prevailed in phase 3 for developers’ discussion on market and strategies as well as their identity and business routines. To explain the latter phenomenon, let us recall that software logic in phase 2 and 3 is composed of two types of discussions: issues related with developers’ traditional distribution channel: the web store, or same set of issues also applicable to App Store apps. Percentage distribution among themes in these two categories in phase 3 particularly suggests that “entrepreneurial decisions”, together with developers’ operational routines, including “creation of apps other than coding” and “small business operations”, are what developers focus on the most regarding the web-store distribution issues. They are also less sensitive to platform governance change and are more likely to share in practices under two distinct distribution channels. The means across all categories shows that in phase 3, software ecosystem logic consists of close to 60% of developers’ entire listserv discussions, and platform ecosystem logic a little over than 40%. This suggests that the two logics co-exist with each other for now; however, the future coexistence of the two logics does not necessarily follow and represents a promising avenue for future work.
Discussion and conclusion

This study set out to investigate the relationship between resource environments and field-level logic in the context of independent Mac OS X developers (Mac indies). Following the cultural emergence model of field-level logics in Thornton et al. (2012), we attempt to answer two research questions: how do resource environments lead to the emergence of field-level logic of independent Mac developers? How do changes in resource environments impact third-party developers’ institutional logic? The findings reveal that two layers of resource environments are present for third-party developers given the structure of platform-oriented software industry: the platform governance and developers’ own economy. Both of them influence developers’ logic through material practices and symbolic meanings. In addition, developers’ economic environment is also affected by platform’s governance. Exploiting a critical change of the resource environment – Apple extending the role from a technology platform to a market exchange owner, and through a narrative and content analysis, we show that a software ecosystem logic prevailed prior to the change, while a new logic: platform ecosystem logic emerged afterwards. Constructing the ideal types of the two logics via a stakeholder perspective, we demonstrate that these two field-level logics are both combinations of societal-level logics, and are also subject to the distinct processes of the software industry. A content analysis of incumbent Mac indies’ discussion also illustrates the temporal logic shift pattern evidenced in the changing field-level practices and symbolic meanings, and the current status of the two logics suggests that they co-exist for the time being.

This study contributes to the institutional logic theory mainly through a validation of the theoretical model of field level logics in Thornton et al. (2012). We illustrate both the material and symbolic aspect of Mac indie’s logic through analyzing the discourse of developers and trade press. This study also contributes to the literature on software platform ecosystem by emphasizing the role of developers in the ecosystem and discussing the effect of platform governance and developers’ own economy at the same time. This adds a fresh perspective on the extant platform ecosystem literature, which tends to focus on the platform itself. This study has several practical implications. It indicates that it is crucial for a platform to develop better understanding about third-party developers, especially about where they come from. For Apple, they face the risk of losing incumbent Mac indie developers on the Mac App Store because its technology policies inconvenience this group of developers. For a technology platform transitioning to becoming a market exchange owner, it is also critical to spend more energy in designing better store experience to facilitate buyer-seller transaction. For instance, lack of direct communication between developers and users has been complained about by developers for a long time. Apple could design mechanisms to enhance such communication experience while retaining its control as a platform.

This study has several limitations. We suggest that the shift in developers’ practices and meaning system is due to platform’s governance change; and yet the method used for the content analysis cannot rule out alternative explanations. For instance, discussion topics could be related to the type of people who post messages, the list guidelines enforced by the list moderator, the auto-correlation among threads topics, etc. However, the goal of the content analysis is not to argue for causality; rather, it is to demonstrate a general trend in the changing pattern of the two logics. Secondly, we only coded the first message in a thread and this could leave out certain dynamics within a given thread. This is less of an issue for the ideal type construction, because of the use of supplemental data sources. However, this could potentially influence the results of the content analysis, although we try to mitigate this limitation with the large number of threads coded.

This study suggests several directions for future research. Given unique attributes of each platform and their governance mechanisms, it would be interesting to examine the resource environment in other software platforms or technology platforms which also incorporate a distribution channel in their governance model, and the impact of the resource environment on third-party developers’ institutional logic. For instance, Google and Microsoft, while both adopt a similar “app store” approach, have policies which differ significantly from each other and from Apple. The implications that these governance policies have on developers’ economy, and consequently their practices and meaning systems would be interesting to study. In addition, it is argued that “the conceptual scheme of the ideal types offers a guide for developing hypotheses about the effects of institutional change on the attributes likely to affect the dependent variables of interest” (Thornton 2004: 25). Thus, the ideal types derived in this study can potentially be developed into hypotheses about Mac indie developers and demonstrate that the effect on dependent variables of interest is institutionally contingent.
Figure 1a. Cultural emergence model of field-level institutional logics in Thornton, Ocasio and Lounsbury (2012: 151)

Figure 1b. Relationships examined in the current study

Figure 2. Process chart of analytical approach in deriving ideal types and temporal logic shift. # indicate incumbent Mac indies and iOS-turned Mac indies respectively.
Figure 3. Stakeholder-view as a general coding framework. In oval shapes are developers’ relationships with stakeholders; in rectangles are developers’ identities and practices.

Table 1. Two ideal types of Mac developers’ institutional logics

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Software ecosystem logic</th>
<th>Platform ecosystem logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Societal-level logics</td>
<td>• Market (specialist) and Profession</td>
<td>• Market (generalist) and corporation</td>
</tr>
<tr>
<td>Economic system</td>
<td>• Market capitalism and personal capitalism</td>
<td>• Market capitalism and managerial capitalism</td>
</tr>
<tr>
<td>Sources of identity</td>
<td>• Independent third-party developer entrepreneur</td>
<td>• Subordinate third-party developer entrepreneur</td>
</tr>
<tr>
<td>Sources of legitimacy</td>
<td>• Quality of apps, Reputation of developer, Platform recognition via awards</td>
<td>• Chart ranking of apps, Platform recognition via featuring, End users’ reviews and ratings</td>
</tr>
<tr>
<td>Sources of authority</td>
<td>• Software engineering and design prowess, Market acceptance</td>
<td>• Platform’s review process, End user interests</td>
</tr>
<tr>
<td>Basis of norms</td>
<td>• Membership in community of practice</td>
<td>• Self-interest, Dependency on the platform</td>
</tr>
<tr>
<td>Basis of mission</td>
<td>• Build sustainable business, Increase sales</td>
<td>• Build competitive position of apps, Build apps to be favored by the platform</td>
</tr>
<tr>
<td>Basis of attention</td>
<td>• Resolve entrepreneurial challenges, Implement infrastructure best suited for the business, Adapt to platform’s system progress and technology change</td>
<td>• Stand out in competition, Embrace the mass app users, Adapt to platform’s changing technological and administrative policies</td>
</tr>
<tr>
<td>Basis of strategy</td>
<td>• Organic growth</td>
<td>“Hit” growth</td>
</tr>
</tbody>
</table>
Table 2. Changing pattern of each stakeholder category over time. Phase 1: prior to announcement of iOS App Store; phase 2: after announcement of iOS App Store and prior to announcement of Mac App Store; phase 3: after announcement of Mac App Store

<table>
<thead>
<tr>
<th>Categories</th>
<th>Phase 1</th>
<th>Phase 2</th>
<th>Phase 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage of platform coding incidents</td>
<td>11.35%</td>
<td>11.31%</td>
<td>21.35%</td>
</tr>
<tr>
<td>Percentage of customer coding incidents</td>
<td>11.35%</td>
<td>13.06%</td>
<td>16.67%</td>
</tr>
<tr>
<td>Percentage of infrastructure coding incidents</td>
<td>13.68%</td>
<td>11.70%</td>
<td>6.25%</td>
</tr>
<tr>
<td>Percentage of market / strategy coding incidents</td>
<td>34.45%</td>
<td>37.04%</td>
<td>30.73%</td>
</tr>
<tr>
<td>Percentage of identity / routine coding incidents</td>
<td>29.16%</td>
<td>26.90%</td>
<td>25.00%</td>
</tr>
<tr>
<td>Total</td>
<td>99.99%</td>
<td>100.01%</td>
<td>100.00%</td>
</tr>
</tbody>
</table>

Table 3. Changing distribution of software and platform logics in each stakeholder category over time

<table>
<thead>
<tr>
<th>Categories</th>
<th>Phase 1</th>
<th>Phase 2</th>
<th>Phase 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Software logic</td>
<td>Platform logic</td>
<td>Count</td>
</tr>
<tr>
<td>Developers’ relationship with platform</td>
<td>88</td>
<td>100.00%</td>
<td>0</td>
</tr>
<tr>
<td>Developers’ relationship with customers</td>
<td>88</td>
<td>100.00%</td>
<td>0</td>
</tr>
<tr>
<td>Developers’ relationship with infrastructure providers</td>
<td>106</td>
<td>100.00%</td>
<td>0</td>
</tr>
<tr>
<td>Developers’ market environment and strategies</td>
<td>262</td>
<td>98.13%</td>
<td>5</td>
</tr>
<tr>
<td>Developers’ identity and routines</td>
<td>226</td>
<td>100.00%</td>
<td>0</td>
</tr>
<tr>
<td>Mean</td>
<td>99.63%</td>
<td>0.37%</td>
<td>89.95%</td>
</tr>
</tbody>
</table>
References


