Paper to be presented at: DRUID17
NYU Stern School of Business, New York, June 12-14, 2017

Platform Architecture, Multihoming and Complement Quality: Evidence from the U.S. Video Game Industry

Hakan Ozalp
LMU Munich
ISTO
h.oezaip@lmu.de

Carmelo Cennamo
Bocconi University
Department of Management and Technology
carmelo.cennamo@unibocconi.it

Tobias Kretschmer
LMU Munich
ISTO
t.kretschmer@lmu.de

Abstract
Multihoming – the decision to design a complement to operate on multiple platforms – is becoming increasingly common in many platform markets. Perceived wisdom suggests that multihoming is beneficial for complement providers as they expand their market reach, but it reduces differentiation among competing platforms as the same complements become available on different platforms. In a study of the US video game industry, we find that multihoming is not as simple as commonly assumed and that platforms, i.e. game consoles, differ in their attractiveness for multihoming complements, i.e. games. Specifically, we find that the complexity of a console reduces the quality performance of multihoming games so that the same game receives a lower quality score on a more complex platform than on a less complex one. However, games that are released on the complex platform with a delay or developed and marketed by a vertically integrated firm suffer a smaller drop in quality on complex platforms, while the use of standardized technological tools – middleware – does not help to attenuate the performance drop on complex platforms. This has important implications for managers considering expanding their reach through multihoming.

Jelcodes: L15, L25
INTRODUCTION
Technology systems based on platform technologies such as Apple’s iPhone, Microsoft’s Xbox or SAP Netweaver are becoming increasingly common in digital markets. One key benefit of platform architectures is that they can elicit innovation by independent developers of complementary products or module extensions (in short, complements) (Gawer 2014, Tiwana et al. 2010). Platforms also facilitate transactions between distinct groups of users, like providers of complements and consumers (Hagiu and Wright 2015, Parker and Van Alstyne 2005, Rochet and Tirole, 2006). One of the platform owner’s critical decisions is the design of the platform’s technological architecture, which determines the platform’s core functionalities (Baldwin and Woodard 2009, Gawer 2014) and ultimately its competitive performance (Gawer and Cusumano 2002, Zhu and Iansiti 2012). Moreover, it affects the incentives of independent firms to innovate and develop complements (Anderson et al. 2014, Tiwana et al. 2010, Claussen et al. 2015a).

The early literature on platforms has emphasized the role of indirect network effects in adoption decisions of independent complement providers and users (Hagiu and Wright 2015, Parker and Van Alstyne 2005, Rochet and Tirole 2006, Rysman 2009). The defining feature of a platform in this literature is that it facilitates transactions between complement providers on one side (sellers) and consumers on the other side (buyers), with each side of the platform benefiting more from the platform the higher the number of users on the other side. Thus, a complement provider wants to reach as many consumers as possible, on the same platform or even across multiple platforms. Indeed, several studies highlight a recent trend of complement providers to develop their products for multiple competing platforms, to “multihome” (Bresnahan et al. 2015, Corts and Lederman 2009, Landsman and Stremersch 2011),1 which represents an important shift

---

1 Interestingly, multihoming is often asymmetric as “…two-sided markets often seem to evolve toward a situation where members of one side use a single platform and the other side uses multiple platforms.” (Rysman, 2009: 130)
in the environmental dynamic of platforms (Tiwana et al. 2010). Most studies on multihoming make the point that the cost of multihoming (developing the product for one additional platform) declined over the past years, not least due to the emergence of “middleware” application tools that make the writing of software code less specific to a given platform technology (Corts and Lederman 2009).² With the cost of multihoming supposedly minimal compared to the high fixed development costs of the product, and with greater expected benefits from targeting users on multiple platforms instead of just one, multihoming has become the preferred strategy for the majority of complement providers in many markets. Consequently, platform specific factors such as the design and core technology are expected to play a negligible role for complementary product innovation as most complements are developed for multiple platforms anyway. Has technology in platform systems been modularized “to oblivion” so that technological quality and architecture are not decisive anymore in attracting high-quality innovation in complements? To tackle this general question, we focus on whether and how platform architecture impacts the innovation performance of multihoming complements. In essence, we ask: Do complements multihome better onto some platforms than others? If so, how does platform architecture affect this?

To get a comprehensive picture of the evolution of platform markets and systems, we address three often-overlooked issues in the existing literature. First, an exclusive focus on the market properties and network effects’ role in the evolution of platform systems neglects the technological dynamics common to many platform markets (Anderson et al. 2014; Gawer 2014), and with it, some of the “hidden” costs of complementary innovation and multihoming due to the specific integration with and workings of a platform core technology. Consider the 2011

blockbuster Skyrim, developed and published by Bethesda. Skyrim collected numerous “Game of the Year” awards, but its Playstation 3 (PS3) version was panned by critics with words such as: “It should be game of the year…it’s not. That experience is in there somewhere, but choked and held under by a bug that slows the frame rate...” (Hurley 2011). Consequently, Skyrim sold over 6 million units for Xbox 360 and only over 2.5 million units for PS3 in its first two years.³ Why do we observe this difference in (commercial and critical) performance across platforms for the same gaming product? One potential reason for this, and the issue we explore in detail, is PS3’s architecture: it was its generation’s most powerful console with its eight-core processor, but also the most complex one, which required a huge effort to program effective games for it (Takahashi 2013).

Second, the costs of multihoming are generally considered as exogenously given by the technological environment (Corts and Lederman 2009, Tiwana et al. 2010). To the extent that a specific platform architecture does not affect all multihoming complements and platforms in the same way, the true costs of multihoming are partly determined by platform owners via their technological architecture and platform governance (Claussen et al. 2013, Claussen et al. 2015a). This has implications for platform evolution and competitive performance (Cennamo and Santaló 2013, Wareham et al. 2014).

Third, understanding how technology-specific elements of a platform affect the innovation performance of complement providers is important to dissect the effect of structural elements of a platform (i.e., the platform architecture) from the effect of organizational elements (i.e., platform governance) on the evolution of the platform and its ecosystem of complement providers (Tiwana et al. 2010).

Consequently, we explore the role of platform-specificity and its impact on the quality of complementary products on a platform. There are multiple ways to capture the platform-specificity of the technology design. For example, Zhu and Iansiti (2012) capture the quality dimension of a platform and shows that even in the face of lower network size for the entrant (Xbox), a platform can successfully enter the market and challenge the incumbent platform thanks to platform-specific relative quality, so that technological quality can substitute to some extent for a large installed base of games for the previous generation (Kretschmer and Claussen 2016). Another way to capture platform-specificity is to acknowledge their modular architectures and to compare platform complexity (Baldwin and Woodard 2009, Tiwana et al. 2010, Gawer 2014). Platform owners may invest more in the core technology to increase platform performance; yet, this often comes with higher technological complexity and development costs for complement providers (Anderson et al. 2014). This then drives platform-specific investments for the better integration of complements (Schilling 2000, Claussen et al. 2015b), higher asset specificity of complement development knowledge (Baldwin 2008, Williamson 1985), and higher architectural learning needs (Sanchez and Mahoney 1996) for supporting the more complex platform.

Cognizant of this tradeoff, we explore how platform complexity affects the innovation performance of multihoming complements by comparing complement quality for more and less complex platforms. We start from the premise that if platform architecture matters for the performance of complementary innovation, the quality of multihoming complements across platforms with different levels of technological complexity will be different. We test this through an in-depth analysis of the U.S. videogame industry. Video game consoles are a platform market with strong cross-side network effects and ongoing technological progress (Clements and Ohashi 2005, Kretschmer and Claussen 2016). We focus on the sample of multihoming games and estimate the quality of complements through game fixed effects regressions to isolate within-game, cross-
platform quality differences, which is what we want to capture conceptually. We find that vis-à-vis other consoles, more complex consoles generally receive lower quality for the same games. However, we find that this link between complement quality and platform complexity does not apply uniformly to all multihoming complements. Specifically, we find that multihoming games produced by vertically integrated firms (publisher and developer belonging to the same firm) experience lower drops in their product quality compared to firms that separate the development from the marketing activity of the game. We also find that for staggered releases (games multihoming to other platforms at different times) the drop in quality for games on the complex console becomes less pronounced with longer lags between releases. Finally, and somewhat surprisingly, we find that multihoming games using third-party development tools (“middleware”) to reduce the part of software development that is console-specific (i.e., information hiding, Parnas 1972) actually perform worse on complex consoles, although this effect is not always statistically significant.

We contribute to the information systems and technology management literatures in several ways. First, by linking a platform’s technological complexity to the quality of the complementary innovations the platform attracts over time, we incorporate the role of platform architecture in the evolution of platform-based ecosystems to bring the IT artifact to the core of theory development in line with recent calls (Tiwana et al. 2010, Orlikowski and Iacono 2001). Second, our results offer insights on the potential asymmetric role of multihoming complements on platform evolution and competition. Conventional wisdom suggests that multihoming complements should mitigate cross-platform heterogeneity because a consumer can access and use the same complement on different platforms. We show that due to differences across platforms in technology complexity, the quality of a multihoming complement is perceived differently by users on different platforms. Accordingly, the impact of a multihoming complement on platform value and evolution can differ
across platforms. Although platforms may appear similar in terms of the (multihoming) complements they provide to their users, they differ in the way these complements work with the platform, and thus, in the way users experience these complements. Finally, by exploring which organizational structures and product strategies by complement providers are better suited to manage multihoming on heterogeneous platforms we offer a complement provider perspective, while existing research on multihoming tends to focus on the platform perspective (Corts and Lederman 2009, Mantena et al. 2010).

PRIOR LITERATURE
The emerging literature on the economics of multihoming largely draws on the broader literature on two-sided (or platform) markets (Parker and Van Alstyne 2005, Rochet and Tirole 2006), which in turn emerged from the literature on (indirect) network effects (Rysman 2009) to explain the dynamics of adoption and competition for platform technologies and their complementary products. Indirect network effects imply that the platform value to users on one side increases with the number of users and their offerings on the other side, so that the higher the number of complements, the more users are expected to adopt the platform, and thus, the greater the incentives for complement providers to support the platform. Accordingly, complement providers tend to choose platforms with a large existing (or expected) user base (Clements and Ohashi 2005, Kretschmer and Claussen 2016), while users tend to choose platforms with many (current or expected) high-quality complements (Binken and Stremersch 2009, Cennamo 2016).

The consequence of these dynamics is that a platform that gains an early edge is expected to become more dominant and eventually take over the entire market (Schneider 2014). Interestingly however, some platform markets like the video game or mobile phone industries do not reflect this winner-takes-all logic. For example, in the early days of the video game industry NES dominated the market with a 90% market share, while later generations had more balanced
market shares (Corts and Lederman 2009; Landsman and Stremersch 2011). Similarly, in many regional markets, mobile operating systems have almost equal market shares between two competitors, iOS and Android (Bresnahan et al. 2015).

Some studies advance that multihoming complements, i.e. complements available on multiple platforms, contribute chiefly to these “deviations” from the expected market structure outcome (Corts and Lederman 2009, Bresnahan et al. 2015). With many of a platform’s complements multihoming to competing platforms, platform-specific differentiation effects are held to even out to a great extent, with no single platform holding an advantage in terms of complements: multiple competing platform systems can coexist in the market.4 The increasing fixed costs of developing software complements such as games or apps relative to the cost of “porting” the game/app from one platform to another, has contributed to this trend towards multihoming (Corts and Lederman 2009). Moreover, many software-based complements benefit from standardized development tools that reduce multihoming costs further, while multihoming can grant benefits from targeting more users over multiple platforms compared to the more limited installed base of a single platform (Corts and Lederman 2009).

Multihoming choices are assumed to be driven primarily by the installed base and market share of the focal platform rather than the technological specificities of the single platform. For example, Bresnahan et al. (2015) consider the cost of multihoming as negligible for complement providers with sufficient quality. However, and precisely because multihoming complements reduce differentiation across platforms, platform owners may invest heavily in the core technology to advance its performance and gain an edge on competition (Zhu and Iansiti 2012).

---

4 However, Lee (2013) argues that absent exclusivity (whether produced by the platform owner or a complementor), competition could have “tipped” in favor of the Playstation 2 during 2000-2005 due to its initial market share advantage.
Contrary to “simple” platforms, technology-intensive platforms do not just serve as market infrastructures facilitating transactions among distinct groups of users (Baldwin 2008); they also serve as technological architectures that complement providers use to develop upon their innovations (Baldwin and Woodard 2009, Gawer 2014). This literature emphasizes the importance of platform architecture for overall system performance because it “…enables other features to be added or existing features to be removed in tailoring derivative products to special market niches” (Wheelwright and Clark 1992, p. 96). Moreover, it sets the rules complement providers need to work with (Baldwin and Woodward, 2009). Investing in greater platform performance can thus convey benefits to users by improving not just the platform core functionalities (Zhu and Iansiti 2012), but also affecting the performance of complements, and thus the overall evolution of the system (Anderson et al. 2014; Cennamo 2016; Claussen et al. 2015a). While the quality of a platform technology can still encourage user adoption and thus affect platform competition (Zhu and Iansiti 2012), a more advanced platform technology can also impair platform performance by increasing the technological complexity (and thus the required learning investment) and development costs of complement providers; this will constrain the overall number of available complements (Anderson et al. 2014). Higher technological complexity can also undermine the performance of complements and affect negatively the overall performance of the platform system over time (Claussen et al. 2015a). These mixed findings leave open the issue of whether and how platform specificity matters for the innovation performance of complement providers, and ultimately for platform competition.

**HYPOTHESES**

**Platform Complexity and Multihoming Complement Quality**

---

5 Product architecture is defined as “the scheme by which the function of a product is allocated to physical components (Ulrich, 1995, p. 419). Baldwin and Clark (2000) describe this in terms of the modules that are part of the system and their functions, and interfaces that define how modules interact.
Modular technologies have standardized component interfaces, which allows complement providers to partly develop their products for the platform technology without the need to fully know the specific workings of the platform (Sanchez and Mahoney 1996, Langlois 2002). Thus, modularity allows complement providers to eschew some of the platform-specific investments: In principle, all they need to learn is how to modify their complement to match the output with the standardized component interface. In turn, modular platforms provide an embedded coordination mechanism between the platform and complement providers for the generation of complementary innovation, which lowers the costs of coordination (Sanchez and Mahoney 1996) by reducing the need for designing dyadic cooperation agreements (Tiwana et al. 2010). This generally generates greater value creation opportunities by joining a multitude of external organizations into “loosely coupled” (Weick 1976) collaboration relationships, typically called platform-based ecosystems (Gawer 2014, Tiwana et al. 2010, Tiwana 2015).

However, modularity does not necessarily reduce technological and interorganizational complexity; there is still a need for the platform owner to facilitate better software-platform integration through output control mechanisms (Tiwana 2015) or other platform governance mechanisms (Cennamo and Santaló 2013, Claussen et al. 2014, Wareham et al. 2014). Greater integration needs are expected for a more complex platform (Anderson et al. 2014, Tiwana 2015), and the costs of developing products for it are likely higher because of the greater investments associated with learning how to best optimize the product for the specific workings of the complex platform architecture.

Platform owners therefore face a tradeoff between modular platform architecture, which minimizes interdependencies among modules through interfaces (Simon 1962, Baldwin and Clark 2000) and greater co-specialization (by complement providers) into the platform system, which requires higher integration between complements and platform architecture (Tiwana et al. 2010).
Complement providers willing to make greater (learning and economic) investments specific to the platform system may eventually achieve greater innovation performance for their products, which benefits the whole system (Claussen et al. 2015b). This creates benefits beyond what would be possible in a more generic, modular system (Schilling 2000). However, in such a tightly integrated system, each component is highly specific to the system, and using non-specific components entails a loss of performance (Simon 1962, Schilling 2000).

Multihoming complements clearly face this tradeoff as they must compromise between platform-specific investments that optimize the product for each platform but duplicate investments, and cross-platform optimization that leverage economies of scale but come at the cost of suboptimal integration with the focal platform. This tradeoff can be particularly strong for complex platforms (Anderson et al. 2014, Claussen et al. 2015b) which require greater platform-specific investments to optimize for and integrate the product with the platform and increase the need for coordination between platform owner and a multihoming complement provider (Tiwana et al. 2010). We thus expect multihoming complements to face greater product integration issues on complex platforms, which would result in lower innovation performance as captured by the quality of their complementary products on that platform.

HYPOTHESIS 1 (H1). The quality of multihoming complements is lower on more complex platforms.

Moderators of the Platform Complexity – Multihoming Complement Quality Relationship

Our hypothesis above refers to the average expected quality of multihoming complements. However, the characteristics of firms and complements themselves can weaken or strengthen this relationship as some firms may be better suited to handle the challenges from developing for a complex platform, or some complements may respond better to it.

Simultaneous vs. Staggered Releases
So far we assumed that the effect of platform complexity on multihoming complements affects all complements equally; yet, they may differ in the way they are developed for different platforms. Complement providers may either simultaneously develop and release their products across platforms, which allows for economies of scale by leveraging their complementary assets, such as marketing or distribution costs across different platforms. Alternatively, they may develop and release the product sequentially on different platforms, using the first platform release to test the market, and later leverage its success by releasing the complement on other platforms (Klompmaker et al. 1976). This has the advantage of focusing development and marketing efforts on the initial platform release, therefore achieving higher quality by optimizing the product to the platform-specific workings. In releasing the product to another platform, the complement provider can modify and tailor the product to adapt it to later platform releases. The more time passes between the original release and the launch to the complex platform, the more the developer can learn about the specific workings of the complex platform and how to optimize the complement for it. However, given the design of the product tailored to the platform architecture it was originally released on, renewed platform-specific investments on the ported platform are required to achieve the same level of quality performance as for the original release. This can be hard and costly to achieve, particularly for complex platforms. While multihoming complements released sequentially may still face lower quality when subsequently ported to the complex platform, we expect that the longer the timespan between release dates on the original and the complex platform, the smaller the magnitude of the drop in complement quality on the complex platform.

6 EA executive Rich Hilleman indicated that EA “now typically spends two or three times as much on marketing and advertising as it does on developing a game.”, EA’s chief creative officer describes game industry’s re-engineering, (http://venturebeat.com/2009/08/26/eas-chief-creative-officer-describes-game-industrys-re-engineering/), accessed 05 September 2016.

7 See for example the release strategy for Grand Theft Auto (GTA), which had been developed for the Playstation in its early editions and ported to the XBOX, while current versions are released simultaneously on all platforms.
A simultaneous product release strategy follows a less platform specific product development approach: the product is designed to be compatible up front with every platform it is released on. Because perfect modularity across distinct platforms is hard to achieve, the key challenge for firms with this strategy is keeping the compatibility costs of product development at a minimum while designing the product to work on multiple platforms (Lengyel 2000). Since the complement provider is focused on developing a modular product with minimal platform specific investments, the complement may perform worse on complex platforms following the logic of our first hypothesis. We thus expect the negative link between complement quality and platform complexity to be stronger for simultaneously released complements than for staggered ones, and to be weaker for staggered releases the longer the timespan between releases.

HYPOTHESIS 2 (H2). The negative relationship between platform complexity and the quality of multihoming complements weakens with the time since the first release of the complement.

Vertical Integration by Complement Providers
Following the “mirroring hypothesis” (Sanchez and Mahoney 1996), the structure of a modular product typically reflects (or should reflect) the organization of its production. Accordingly, for more complex platforms requiring greater integration with its complements, vertically integrating the production of complements should facilitate better complement performance. We study this logic at the level of complement providers who can vertically integrate development and publishing of the complement. In the video game industry, the actual development of the complement and its funding and marketing are often undertaken by different specialized firms, the developer and the publisher.

The need for vertical integration extends to complement providers for a complex platform. Greater platform complexity entails an increase in asset specificity (Williamson 1985) of the complement development process for the platform. This requires greater coordination between
platform owner and complement provider. Greater coordination is also required between developers and publishers of the complement. Under a contracted relationship, both the developer and the publisher cannot fully anticipate the possible development issues that may unfold during the process and will organize (and contract) production and marketing of the complement according to the average time and effort required. Once the launch date of the product is set by the publisher, the developer must get the game ready for this date. This might require sacrificing some of the extra (learning) investments required by more complex platforms and compromising on platform-complement integration for more complex platforms, resulting in lower quality complements. In contrast, a vertically integrated firm controlling internally both development and marketing activities can better coordinate the two and undertake additional specific investments needed to better integrate the complement with the complex platform.

HYPOTHESIS 3 (H3). The negative relationship between platform complexity and the quality of multihoming complements is weaker for complements developed by a vertically integrated provider.

Development Tools in the Production of Multihoming Complements
Our logic assumes that technological complexity cannot be easily reduced by complement providers unless they make specific learning investments into it. In fact, complexity within a platform’s subsystems can be shielded with a simple interface that hides the details of the interactions within the subsystems (Gawer 2014). An example is the graphical user interface (GUI) of the Windows, where every click by the user translates multiple and complex commands to the computer. Similarly, in the development of multihoming complements, development tools are available to reduce the complexities of each platform. For example, in the video game industry, middleware tools hide the inner workings of each platform and provide developers with a specific development framework that can translate the output to any platform-specific code. Therefore, we
expect that complements using such development tools will be better able to integrate with complex platforms as part of the platform-specific integration task will be handled by these tools.

HYPOTHESIS 4 (H4). The negative relationship between platform complexity and the quality of multihoming complements is lower for complements using development tools.

Figure 1 summarizes our hypotheses:

-------------------------------------
INSERT FIGURE 1 ABOUT HERE
-------------------------------------

METHODS
Setting: The US Video Game Industry
We study multihoming games in the U.S. Video Game Industry from 1999 to 2010. The industry is a quintessential example of a platform technology, with strong indirect network effects and the key importance of complementary products (Clements and Ohashi 2005, Cennamo and Santalo 2013, Kretschmer and Claussen 2016). Moreover, video game consoles are essentially IT artifacts with varying designs (Anderson et al. 2014), which lets us explore the role platform architecture plays in complements’ performance (Claussen et al. 2015a). Also, the industry shows rich heterogeneity in games and their developers, which gives us an opportunity to explore whether the hypothesized negative effect of complex platform architectures on complement performance applies uniformly to all multihoming games, or affects game providers differently depending on how they organize with respect to game development.

Video game console platforms are (typically incompatible) video game systems that compete with each other. These platforms are grouped into technological generations (Anderson et al. 2014, Corts and Lederman 2009). Each generation represents a group of consoles that are roughly similar in their hardware specifications, whereas different generations often differ significantly in their computing power and functionality (Cennamo 2016, Kretschmer and Claussen 2016). There have been eight generations of platforms in the video game industry, spanning 1972
to the present. We focus on the years 1999 to 2010 inclusive, which covers the launch and entire evolution of all sixth and seventh-generation consoles. Table 1 presents the platforms included in our sample, grouped into generations with some key characteristics.

As can be seen from Table 1, Generation 7 consoles represent a big improvement in their technical specifications compared to Generation 6 consoles. New hardware allowed more content and improved graphics, which also greatly increased the fixed cost of game development (Anderson et al. 2014). A blockbuster game in 1995 had a $1.5 million budget, whereas a blockbuster game in 2010 cost on average $60 million to develop (Kotaku 2014), and can run up to >$200 million for GTA V (released in 2013), the latest installment in a series of blockbuster games. Moreover, with the increased use of licensed content, the rise in content development costs that are not console-specific (e.g., voice acting, music…), and a decrease in multihoming costs due to multi-console programming and development tools have made multihoming more attractive (Corts and Lederman 2009). Consequently, the share and number of multihoming games in the industry has increased drastically. Figure 2 presents the increase in average development team size (the most significant component of a game’s fixed development costs) and the increase in multihoming games in the industry.

Our empirical setting provides a natural way of identifying our effect by looking at multihoming games. By using game fixed-effects, we can avoid any omitted variable bias arising from between-game differences and isolate game-level variance in quality from variance in quality performance across consoles due to game-console integration. Put differently, we can take the exact
same game (e.g., publisher, developer, genre, and so on), and look at how the same game performed differently across separate platforms. Yet, a potential econometric challenge with focusing on multihoming games is sample selection: games that multihome may (and do) differ in other aspects compared to games exclusively released on single console. In our estimation strategy section below, we discuss this further and offer our considerations on why this selection, if in place, would not affect our results.

Data
We build our dataset from multiple sources. Our primary data source is the MobyGames website, the world’s largest online video game archive on the Internet (Corts and Lederman 2009, Mollick 2012, Claussen et al. 2015a, Kretschmer and Claussen 2016). MobyGames provides detailed information for a game on its platforms of release, dates of each platform release, publisher and developer, characteristics (e.g., genre), and use of development tools in production (e.g., middleware and game engines). Although MobyGames provides publisher and developer information for each game, it cannot fully account for parent–subsidiary relationships. As this information is crucial for our purposes, we complemented the information from MobyGames with information manually collected from GiantBomb, official firm websites, and Factiva.

Our second data source is the GameRankings website, “a site dedicated to aggregating review scores from both online and offline sources, to give users an overall picture of a game's score.”8 The site has review information for over 14,000 games, with over 300,000 individual game reviews. To ensure the accuracy of review scores, GameRankings has strict requirements on which reviews (and their scores) can be added on the website.9 The website standardizes review scores

---

9 “The requirements for adding a new site are: Sites must have at least 300 archived reviews for a multi-system/multi-genre sites, or 100 reviews for single-system or genre sites; Sites must publish a minimum of 15 reviews a month; Sites must be visually appealing and looks professional; Sites must review a variety of titles; Sites must have a dedicated domain name with professional hosting; Site reviews must be well written; Sites must conduct themselves in a professional manner.”, available at http://www.gamerankings.com/help.html, accessed 23 August 2016.
from different outlets and produces one aggregate score for each game. We use scores on the GameRankings website for our analysis.

Our matched dataset includes 4,721 title-platform releases, which includes all games released in the US from 1999 to 2010 for sixth and seventh generation video game consoles. This number excludes games that are not sold in retailers (in our period of observation, download-only game titles represented small, independent game development efforts vastly different from game titles sold at retailers). For 526 title-platforms, we do not have information on their review scores, our dependent variable. From the remaining 4,195 title-platforms, 391 of them are published by the console owners themselves ("first-party games"). These first-party games are used for the calculation of a control variable but removed from our final sample for two main reasons. First, these games rarely multihome unless onto the next-generation console of the same owner (e.g., MLB 07: The Show, released on Playstation 2 and Playstation 3). Second, the console owner holds intimate knowledge regarding the architecture of the console, and also because it is the same firm designing the platform architecture and developing games specific for it, first-party games have high levels of quality independently of platform complexity.\textsuperscript{10} We further removed the Sega Dreamcast from our sample as mostly of the games in our sample were produced by Sega, owner of the platform; only 16 third-party games released for Dreamcast were multihoming. Moreover, all of these multihoming games were initially released exclusively for Dreamcast as it was launched one year before the Playstation 2 and two years before Gamecube and Xbox.\textsuperscript{11} These games always performed much better on Dreamcast compared to their later versions ported to other consoles. After removing these games, we are left with 3,635 title-platform observations, representing 2,343

\begin{footnotesize}
\textsuperscript{10} In fact, first-party games produced by Sony are able to take full advantage of its complex architecture and these games perform much better compared to multihoming titles released by third-party publishers on the platform.

\textsuperscript{11} However, we also tested our results that adding Dreamcast games to our models do not change our results.
\end{footnotesize}
sixth-generation console title-platform observations and 1,292 seventh-generation console title-platform observations.

Since each generation represents an ecology in which games and hardware are compared to each other (de Vaan 2014), we separate our sample for sixth and seventh generation consoles. We further removed singlehoming games. Again, we use these games to create a platform-level control variable, but we remove them as we require within-game variation in quality across platforms. This leaves us with a sample of 1,483 sixth-generation console title-platform observations and 806 seventh-generation console title-platform observations, a total sample of 2,289 title-platform observations. We first run our regressions for each console generation separately, and then run a regression with the combined sample.\(^{12}\)

**Platform Architecture and Complexity**

Platform owners carefully design their hardware as it affects a whole ecosystem of complement providers who need to work with it for the next six to seven years. There are three main components of platform design: CPU (including any co-processors), Graphics Processor (again, including co-processors), and RAM. Although other dimensions of performance matter (e.g., bus speed, cache memory, optical media, and so on), these three are by far the most important ones (and largely determine other architectural elements as well). When releasing new consoles, platform owners aim at high hardware power as this is the way to push cutting edge graphics games and the way to lure both users and developers alike (Kretschmer and Claussen 2016, Claussen et al. 2015, Zhu and Iansiti 2012). There is emerging theory (Anderson et al. 2014) and cases from the industry indicating that the advanced, but more complex, hardware creates difficulties in the production of complements. This explains for example why in the fifth-generation consoles, while Nintendo 64

\(^{12}\) For the combined sample, we also removed 52 title-platforms that multihomed across generations, leaving us with 2,237 title-platforms for the combined sample regressions.
and to a lesser extent Sega Saturn offered more hardware power, the Playstation was able to source much higher number of games with its easy to develop for platform architecture (Kent 2001), as shown in Figure 3. Similarly, in seventh-generation consoles, the PS3 offered great hardware power, but it lagged behind Wii and Xbox 360 in terms of the number of games developed for it, as shown in Figure 4. These cases suggest a common issue across generations: platforms with the most advanced platform architecture were also the most complex ones and difficult to develop games for (Reimer 2005), thus obtaining less games overall.

Two related issues are emphasized for “complex” platforms in the industry. First, the number of processors in a console was highlighted as a main source of complexity (Kent 2001, Pettus 2013, Parish 2014). The reason of this complexity is borne out of the need to manage an increasing number of interdependencies to utilize the architecture in an optimal way. Second, the complexity is also attributed to the need to use platform-specific language to optimize (i.e., to integrate) the game code for the platform (Pettus 2013, Roth 2013), which is also known as the “assembly language”. “a somewhat awkward non-expressive computer language that uses a mnemonic for each instruction of the instruction set, and is difficult to program in” (Afuah 2000, p. 402). This language is specific to the architecture of platforms, and represents a highly tedious programming task compared to well-known “higher-level” languages used commonly in game programming such as C and C++. Programmers needed to deal with low-level languages in two

---

13 These numbers might also be driven by market shares, however this seems unrealistic given the disproportionally low number of game releases for Nintendo 64 (given its market share) and disproportionally high number of game releases for Xbox 360.

14 “Saturn had eight processors... to get that platform to do what it was designed to do was a very complex and painful learning process for developers, including the best and sharpest minds that Sega had to bring to bear on it, in both Japan and the U.S.”, Interview: Joe Miller (Sega of America Senior VP of Product Development), http://www.sega-16.com/2013/02/interview-joe-miller/, accessed 07 September 2016.

15 “Trying to program for two CPUs has its problems. … The two CPUs start at the same time but there's a delay when one has to wait for the other to catch up... I think that only one out of 100 programmers is good enough to get that kind of speed out of the Saturn.” (Yuji Naka, lead programmer and creator of the “Sonic the Hedgehog”; Pettus, 2013, p.193).

16 This programming language consists “mostly of symbolic equivalents of a particular computer’s machine language. Computers produced by different manufacturers have different machine languages and require different assemblers and assembly languages. [P]rogramming in assembly languages requires extensive knowledge of computer architecture” (http://www.britannica.com/EBchecked/topic/39243/assembly-language, accessed 24 February 2015).
cases. First, when the platform contained a specialized or new processor for which no interpreter (“compiler”) existed for programming with higher-level languages. This represents a case of “thin crossing point” (Baldwin 2008) where standardized interfaces are not available or cannot easily be used to reduce platform-specific complexity by programmers. Second, programmers also used low-level languages when the platform had a high number of processors that needed to be coordinated as they give the most control over the hardware in allocating programming tasks. Table 2 reports information on the level of complexity of all consoles released in the U.S. in the fifth, sixth and seventh generation.

---

**INSERT TABLE 2 ABOUT HERE**

---

An example is helpful to illustrate the sources of variation in console complexity. Consider the technical architectures of the Xbox 360 and PS3 in Figures 5 and 6 respectively. Both consoles seem similar at the highest level of architecture (a CPU, a graphics unit – GPU, RAM, and an interface to connect the system to the output and the optical drive). However, the PS3 is much more complex. It uses a CPU with one main generic core (PPE) and seven specialized cores (SPE), requiring a careful allocation of tasks from the main core to these specialized cores on the part of the programmer. These specialized cores also have their own programming language. On the other hand, the Xbox 360 uses a CPU design with three generic cores (PPE), which are easier to handle both because of the lower number of cores and the minimal need of using low-level languages. The Appendix\(^\text{17}\) provides additional details of the technical architecture of each platform in our sample. Note that the Playstation 2 and Playstation 3 are the most complex consoles for the sixth and seventh generation, respectively.

**Dependent Variable**

\(^{17}\) Appendix is unfortunately not available in this submission due to word limit for this submission.
We study the comparative quality of games released on more complex vis-à-vis less complex platforms. We measure the quality of games by using the average aggregate GameRankings score with the variable Title-Platform Quality. The quality score varies between 0 and 100, with the average quality score being close to 70. As the quality score increases towards 100, the returns for that success (e.g., by sales) increase exponentially as the video game industry is “hit” driven: the top 5% of games capture around 50% of all sales (Binken and Stremersch, 2009). Games released on different platforms often receive different reviews, reflecting distinct game graphics or smoothness of gameplay between different platforms. Therefore, the quality of a specific version of a game on a particular platform gives information on how well a game has been implemented on a platform. Below, we discuss further how we use this feature of review scores to estimate the platform-specific effects.

**Independent Variables**

Complex Platform. We examined the effect of the complex platform architecture on the quality of complements (games) by using a dummy for those platforms. As explained above and in the Appendix, Playstation 2 and Playstation 3 are the complex platforms in the sixth and seventh generation, respectively. In our combined analysis of sixth and seventh generation consoles, the Complex Platform variable equals one if a game is released on Playstation 2 or Playstation 3, zero otherwise.

ln(Delay in Days). This variable measures the logged number of days of delay for the focal platform release since the first release of the game for another platform. It is zero for simultaneous releases and the first platform release of a game.

Inhouse Development. We capture vertical integration by creating a dummy that takes value 1 if a game’s developer and publisher belongs to the same parent company. A publisher in this industry may opt to develop a game by using one of its internal developers (“studios”, in industry jargon).
or external contracted developers (“independent developer”). In fact, a key aspect of publishers’ strategy is to balance its portfolio of vertically integrated games (“inhouse”) and contracted games. Licensed Middleware. We identified games using development tools for cross-platform development with this dummy, which takes value 1 if the game uses a licensed graphics engine (including well-known graphics engines categorized as “middleware” in our dataset), game engine, or 3d engine. We excluded minor middlewares (e.g., programs that create trees automatically – “Speed Tree”), as well as game engines and other middleware owned by the publisher or developer of the game (e.g., Unreal Engine used in the game Unreal Tournament by Epic Games, who owns and licenses the Unreal Engine).

Control Variables
We have two control variables to control for possible spurious relationships for the effect of a complex platform architecture on the quality of the game. The variables Average Quality of 1st Party Exclusives and Average Quality of 3rd Party Exclusives measure the average quality score of the exclusive games released by the platform owner in a year and 3rd party publishers in a year, respectively. These variables control for two effects: first, game reviewers may compare games released on a platform with each other, so that the quality of exclusive games on the platform sets the benchmark with which multihoming games are compared. Moreover, the benchmark set by exclusive games may also affect the incentives and ultimately the investments by producers of multihoming games to optimize their product for the focal platform. This effect could go in either direction as multihoming games may either “step up” to match the quality of exclusive games or “give up” and occupy a lower quality segment; it is thus important to control for it.

Estimation Approach

18 Wii has been an interesting case that most commentators blaming reviewers of Wii games, for which they accused with downgrading the score of any release on the platform. “Wii reviewers are the problem – Braben”, http://www.eurogamer.net/articles/wii-reviewers-are-the-problem-braben, accessed 27 August 2016.
To understand how platform-specificity affects the quality of complements, and, subsequently, how our other independent variables, (\(\ln(\text{Delay in Days})\), inhouse development, and middleware use), moderate this relationship, we take as unit of analysis the game and model the quality of game \(i\) on each of the platforms \(j\) it was released as follows:

\[
\text{Quality}_{ij} = a_i + \ln(\text{Delay in Days})_{ij} + \text{Average Quality of 1st Party Exclusives}_j + \text{Average Quality of 3rd Party Exclusives}_j + \text{Complex Platform}_j \times \ln(\text{Delay in Days})_{ij} + \text{Complex Platform}_j \\
+ \text{Inhouse Development}_i + \text{Complex Platform}_j \times \text{Licensed Middleware}_i + \varepsilon_{ij}
\]

Our dependent variable, game quality, is time invariant; however, we do observe that a game is released on multiple platforms at different points in time. To compare them, we pool the data into cross-sections and run a linear regression analysis. Our regression contains game fixed effects \((a_i)\), which captures all factors that do not change between releases of the same game across different platforms. In doing so, we identify differences across consoles for a given game.\(^{19}\) More specifically, our Complex Platform dummy will capture the differences in the game quality on the complex console compared to the mean quality of the game across all other consoles it was released on.

We focus only on multihoming games to identify the effects we hypothesized, excluding from the sample games that are released on only a single console. This, however, might bias our estimations if there are systematic differences across these two groups not captured by our independent and control variables affecting the quality difference of a game across consoles. We believe that our results are not affected by such a potential bias. First, it might be that indeed the quality of exclusive games on a focal console is systematically superior to that of multihoming games if developers of exclusive games can better optimize the game for the platform and because exclusive games might be rewarded with better programming access and support. This would not

\(^{19}\) Note that the coefficients on the “direct” terms of the variables Inhouse Development and Licensed Middleware cannot be independently estimated because we are controlling for game fixed effects; therefore, these terms are not present in our regression model. For a similar example, please see Boudreau, Lacetera, and Lakhani (2011).
bias our results as we focus on quality differentials across platforms of the same game rather than absolute levels of quality across games in a given platform. Second, there might be differences in developers’ skills and resources affecting the likelihood of selecting into one group (multihoming) or the other (exclusive). On one hand, if big developers with ample development and marketing resources are most likely to multihome their games, this selection would actually work against our predictions. These developers are well equipped to overcome the technological complexity associated with development of games on complex platforms, and thus should be expected to experience less, or no differences in complement quality across platforms. Finding a robust effect for a complex platform is evidence supporting our main hypothesis. Indeed, in our sample, most of the multihoming games come from large publishers. In our sample, multihoming games themselves are larger projects on average (team size 147 vs. 71), and published by firms that are older (12 years old vs. 9 years old) and larger (mean of 28 games (title-platform) published per year vs. 17 games (title-platform) published per year).

Third, selection into multihoming may be driven by increasing competition from high-quality first-party (games produced by the platform provider) and exclusive games. Here, developers may decide to multihome as a way to reduce competitive pressure from the games on the focal platform. Again, if porting costs were irrelevant compared to development and marketing of the initial product (Corts and Lederman 2009), there would be no reason to expect quality differences for these games across platforms. Also, note the sharp decrease in the strategic exclusivity deals with third parties over time in the industry. As Figure 2 shows, exclusive third-party title-platform releases have decreased from 45% in 1995 to about 30% in 2010.

**RESULTS**

Summary statistics and pairwise correlations for our three regression models are in Tables 3 and 4. Inhouse developed games are strongly correlated with game quality. Complex Platform
shows strong correlation with the Average Quality of 3rd Party Exclusives variable. This correlation is especially strong for the seventh generation (0.91). Indeed, the Complex Platform variable for the seventh generation sample has a VIF value of 11.11, which is above 10. Having two separate regressions for each generation will allow us to show that the results are not driven by multicollinearity. Moreover, we test the robustness of our results by excluding the Average Quality of 3rd Party Exclusives variable and find that our results are unaffected. In 5.1, we test our hypotheses for sixth generation consoles, in 5.2, for the seventh generation. Finally, in 5.3 we report our robustness tests and some analysis to further explore our main results.

Sixth Generation Consoles

Results are reported in Columns 5-1 to 5-5 in Table 5. The first model includes control variables along with the other variables we use later as moderators. It is worth noticing that the main effect of time delay in the release (ln(Delay in Days)) is negative and significant (p < 0.01), suggesting that games that are released with a delay get a lower quality score on average compared to their initial release. We cannot identify the main effect for the other two moderators, Inhouse Development and Licensed Middleware, because they are included in the game fixed effects.

Hypothesis 1 states that multihoming complements have lower quality on complex platforms compared to the other platforms they are released to. Columns 5-2 to 5-5 show that Complex Platform indeed has a negative and statistically significant coefficient, providing support for our first hypothesis for consoles of sixth generation. Hypotheses 2, 3, and 4 investigate moderators of the relationship between platform complexity and multihoming complement quality.
Hypothesis 2 predicts that the drop in quality on complex platforms decreases in the existence and the extent of a delay between initial release and subsequent release on a complex platform. We test this by interacting Complex Platform and ln(Delay in Days). Columns 5-3 to 5-5 report supporting evidence for the Hypothesis; the interaction is positive and significant so that the longer the time between the first release of the game and the release to the complex platform, the smaller the difference in game quality between Complex Platform and other platforms. Note also that ln(Delay in Days) has a negative main effect on a game’s quality on the focal platform relative to the quality on the platform it was first released. We interpret the interaction in Column 5-3 graphically by plotting the mean game quality for complex and non-complex platforms across no delay, mean level of delay (0.88) and delay at one standard deviation above mean and two standard deviations above mean. As Figure 7 shows, in the case of simultaneous release of multihoming games, complex platforms are at a significant disadvantage, whereas this difference disappears with increasing delay. Figure 8 shows the marginal effect of release delay on the quality of multihoming games for the groups of complex platform (i.e., Complex Platform=1) and non-complex ones. We also find support for Hypothesis 3 that vertically integrated firms have lower drops in quality on a complex platform, as indicated by the positive and significant coefficient of the interaction between Complex Platform and Inhouse Development in Columns 5-4 and 5-5. According to Column 5-4 in Table 5, the mean quality of an inhouse developed game is 70.91, whereas a game developed and published by separate firms is 70.12. We further run game fixed-effects regressions to estimate the economic significance of this difference.\textsuperscript{20} We find that the difference between quality scores (0.79) amounts to US $443,000 difference in the first year sales of the game (US $8,123,174 for 70.91 quality score compared to US $7,680,403 for 70.12 quality score). Note also that this sales

\textsuperscript{20}We used quality scores as the independent variable, and the market share of the console in each year as a control variable. Dependent variable is cumulative 12 month sales of the title-platform from its release date.
difference is for the average title and sales differences will grow disproportionally for games at higher quality levels (i.e., disproportional returns to high quality). We do not find support for our Hypothesis 4 that licensed middleware reduces the quality drops on complex platform. In fact, the interaction between Complex Platform and Licensed Middleware in Column 5-5 shows a negative and marginally significant coefficient.

**Seventh Generation Consoles**

Columns 5-6 to 5-10 in Table 5 report results for seventh-generation consoles. The main effect of the time delay in the release (ln(Delay in Days)) is again negative and significant (p < 0.01), showing a consistent, across-generation detrimental effect of delayed releases on the quality of multihoming, staggered games. Also, the control for the average quality of third-party exclusives (Average Quality of 3rd Party Exclusives) is significant and positive (p < 0.01), suggesting that platforms with higher average quality third-party exclusives also had increased quality multihoming games.

The Complex Platform variable is negative and significant across Columns 5-7 to 5-10, further supporting Hypothesis 1. The interaction between the Complex Platform and ln(Delay in Days), is again significant and positive in Columns 5-8 to 5-10, which again supports Hypothesis 2. We plotted our results from Model 5-8 to better appreciate the total impact. Figures 9 and 10 show similar patterns to those of sixth generation consoles, in line with our hypothesis. The interaction between Complex Platform and Inhouse Development (5-9 and 5-10) is positive, but insignificant. Columns 5-10 test the effect of the Complex Platform and Licensed Middleware interaction; in line with our findings in 5.1, the interaction is negative, but insignificant. Accordingly, we do not find support for Hypothesis 4.

**Robustness**
We ran multiple robustness checks to ensure the validity of the results and report them in Table 6. We first combine our samples of sixth and seventh generation consoles. In this regression, the Complex Platform dummy represents games released on Playstation 2 or Playstation 3 (the complex platforms in their respective generations). Model 6-1 reports results supporting our Hypotheses 1, 2, and 3. Model 6-2 excludes the Average Quality of 3rd Party Exclusives control to check if multicollinearity drives our results and find our results unchanged. Model 6-3 runs the baseline model of Model 6-1 with standard errors clustered at the title level. Again, results remain largely unchanged, although the significance of the Complex Platform x ln(Delay in Days) interaction reduces to the 5% level, while the significance of the Complex Platform and Inhouse Development interaction increases to the 1% level. We also break down our results for each console for the sixth and seventh generation consoles. In Model 6-4 we used dummy variables for Gamecube and Xbox and compare their coefficients to the complex platform (the excluded dummy), Playstation 2. Model 6-4 shows that both Gamecube and Xbox dummies are positive and significant, with a joint significance at $p < 0.01$ level. The interactions of these two platform dummies with ln(Delay in Days) are also jointly significant at $p < 0.05$, corroborating support for Hypothesis 2. We further find jointly significant result for the interactions of these platform dummies with Inhouse Development ($p < 0.05$), which supports Hypothesis 3; but we do not find the coefficients for the interactions with Licensed Middleware ($p=0.14$) to be significant. Model 6-5 repeats the same analysis for seventh-generation consoles, comparing the Wii and Xbox 360 dummies with the excluded dummy for the complex platform, Playstation 3. Here, we find jointly significant results for the Hypothesis 1 ($p < 0.01$), and Hypothesis 2 ($p < 0.05$), but no support for Hypotheses 3 and 4.

DISCUSSION AND CONCLUSION

We study the impact of platform architecture on multihoming complement quality and consider the complexity of the underlying platform core technology as a key platform specific factor in determining the quality of multihoming complements (after controlling for game-specific factors).

---

\[21\text{ After merging, we excluded 41 observations that were multihomed cross-generation as our analysis focuses on comparing platform complexity across platforms of the same technological generation.}\]

\[22\text{ There are additional robustness checks not reported here and in Table 6 due to space constraints, such as adding Dreamcast to our sample. Results remain unchanged.}\]
We look at multihoming video games in two generations of video game consoles from 1999 to 2010 and find support for our main prediction that games receive lower quality scores on complex platform architectures. Put simply, it is more difficult to port a game onto a complex platform and accordingly, quality performance of the complement on that platform is lower. This confirms that platform characteristics, and in particular their technological complexity, matter for the decisions and outcomes of complement providers to develop and port their complements to specific platforms. Multihoming therefore depends on which platform a complement is multihomed to. Moreover, as complex platforms are typically the most advanced of their generation, we add another dimension to the discussion on why superior technological performance does not always yield a competitive edge (Anderson et al. 2014, Kretschmer and Claussen 2016, Claussen et al. 2014).

Our study of the video game consoles also sheds light on the different product and firm strategies that are more or less suited to manage complements for a complex platform. We find that simultaneous game releases perform worse on complex platforms than staggered releases, as under time pressures the design of these complements tends to be more modular and hence less integrated for a complex platform. We also found that firms vertically integrating the development and publishing of the game experienced lower drops in quality when porting their products to a complex platform. Finally, we did not find robust evidence that the use of licensed middleware facilitated multihoming on complex platforms.

Overall, our findings show the value of considering platform architecture in explaining the evolution of platform markets and systems. Hence, we answer recent calls to bring the IT artifact to the core of theory development (Tiwana et al. 2010, Orlikowski and Iacono 2001). We offer a first answer to the question of “how platform architecture influences the evolutionary dynamics of ecosystems and modules [complements] in platform settings” (Tiwana et al. 2010) by focusing on a key dimension of platform architecture: complexity. Interestingly, a seemingly beneficial design choice by the platform owner (higher performance that brings about complexity) results in lower quality multihoming complements. This has multiple implications. First, complex platforms require higher co-specialization by third-party developers to achieve high-quality of software performance. Using cross-platform development technology such as middleware tools does not help in avoiding these platform-specific investments; exclusive complements are better suited. The owner of a complex platform might need to encourage more exclusive products (or develop them
itself) to take advantage of its architecture, and affect the evolution of the platform rather than sticking to lower-quality multihoming complements. Our findings also contribute to the understanding of the coevolution of the platform and its environment (Tiwana et al. 2010). In the video games industry, the (direct) costs of multihoming have decreased over the years, reflecting a change in its environment. This made the high-performance, high-complexity design choice by Sony a profitable one in the era of the Playstation 2 when Sony was dominant and had lots of exclusive titles on its console, but a costly one for the Playstation 3 when it had to compete head-to-head with Xbox 360 and multihoming titles became the norm in the industry. Sony’s current console, the Playstation 4, has now been designed to be much closer to a simple PC architecture to leverage multi-homing games. Indeed, the current head architect of the platform deliberately made this choice based on his experience with the Playstation 3 (Takahashi 2013). In fact, all three players in the industry (Microsoft – Xbox One, Sony – Playstation 4, and Nintendo – Wii U and upcoming NX) have chosen to use simple architectures for their latest consoles. Microsoft and Sony, which share many multihoming titles with PCs, even chose components commonly found in PCs.\(^{23}\) Our results also let us make some predictions about platform evolution. For example, we expect that complex platforms attract more games from vertically integrated developers of complementary products as they have the organizational means to overcome the corresponding integration problems. Conversely, if the industry starts to rely more on simultaneous releases, complex architectures may turn out to be a burden for the platform owner, especially in an environment where exclusivity is the exception.

We also gather insights into the economic characteristics of platforms. First, we note that there are some “hidden”, platform-specific costs of complementary innovation and multihoming due to the need to integrate a complement with a given platform architecture. Hence, part of the costs of multihoming are not exogenously given and homogenous across platforms. Second, we show that multihoming complements need not equalize indirect network effects across platforms, suggesting an asymmetric effect on platform evolution and competition. Rather than finding that multihoming complements reduce differentiation between platforms as consumers can access the same complement on different platforms (Corts and Lederman 2009, Rysman 2009), our results suggest that the quality of complements differs across platforms depending on their complexity.

\(^{23}\) It is also interesting to observe Nintendo continued to stick with a PowerPC CPU (instead of Microsoft and Sony that switched to standard PC x86 CPU) due to its exclusivity driven game library.
This opens up another strategic dimension for platform owners, the platform’s amenity to attract high-quality multihomed games, which in turn calls for integrating the design rules for complement providers (Baldwin and Woodard 2009, Gawer 2014) with the economic market incentives to multihome (Corts and Lederman 2009, Bresnahan et al. 2015).

Our results also have a number of managerial implications. First, the difference in quality performance for the same game on distinct platforms reveals a hidden economic cost that managers might not consider when multihoming to different platforms. Research shows that the quality of a game is an important predictor of game sales (Zhu and Zhang 2010). Accordingly, when multihoming to a complex platform, managers may want to stagger the release and take more time to optimize and integrate the complement with the specific workings of the complex technology. Second, explicitly considering platform architectures can help complement providers consider various strategies in multihoming their complements on complex platforms. Specifically, vertically integrated development may be more successful when multihoming to a complex platform. This creates another potential hidden cost of multihoming: If multihoming calls for integration, this reduces platform owners’ flexibility to react to technological and market changes (Claussen et al. 2015b). Similarly, it seems prudent to focus complement development first on the complex platform, and then modify (often simplify) this code to other platforms over time.

Our results of how platform architecture influences platform complementary innovation opens up new research questions at the firm- and platform-level. First, if platform complexity generates development and performance issues for complement providers, why do firms choose to design a complex architecture in the first place? Some evidence exists that more advanced platforms can offer more value to users and outperform competitors (Zhu and Iansiti 2012). However, as these improvements in performance come at the cost of greater technological complexity, these platforms may also attract fewer complements (Anderson et al. 2014) or, as shown in our analysis, complements of lower quality. If platform owners cannot resolve the technological dilemma of having a technologically advanced platform that is not complex, the real question becomes one at the organizational level: How do platform owners manage this complexity with their complement providers? We have limited knowledge on the inter-organizational schemes and mechanisms used by platform providers to guarantee high-quality integration with complements, and to govern the evolution of the ecosystem of complement providers to reduce unintended variance (in quality) and increase intended variance (variety of high-quality
complements) (Tiwana et al. 2010, Wareham et al. 2014). This relates to one common strategy by platform owners to maintain control over the type and quality of complementary innovations produced: they produce complements themselves. However, this strategy has both advantages and disadvantages (Gawer and Henderson 2007). If platform owners vertically integrate more, co-opetition with complement providers will be stronger (Cennamo 2016), especially for those releasing multihoming games. This may create additional tensions with complement providers. Overall therefore, we believe that research on platform technological infrastructures and the ecosystem of platform core, first-party and third-party complements has to include both strategic and technical parameters to fully reflect the underlying dynamics of platform evolution and innovation.
References


http://www.gamasutra.com/blogs/SimonRoth/20130522/192814/Coding_quotTo_The_Metalquot_is_a_dan gerous_ideal.php.
Table 1. Sample Consoles in US Video Game Industry (1999-2010)

<table>
<thead>
<tr>
<th>Console</th>
<th>Generation</th>
<th>U.S. Launch Date</th>
<th>Platform</th>
<th>Parent</th>
<th>CPU</th>
<th>Total System and Graphics RAM (Mb.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dreamcast</td>
<td>6</td>
<td>Sept. 1999</td>
<td>Sega</td>
<td>Sega</td>
<td>Hitachi SH-4 RISC @ 200 MHz</td>
<td>24</td>
</tr>
<tr>
<td>Playstation 2</td>
<td>6</td>
<td>Oct. 2000</td>
<td>Sony</td>
<td>Sony</td>
<td>Custom Made “Emotion Engine” RISC @ 294 MHz</td>
<td>36</td>
</tr>
<tr>
<td>Xbox</td>
<td>6</td>
<td>Nov. 2001</td>
<td>Microsoft</td>
<td>Microsoft</td>
<td>Intel Pentium III x86 @733 MHz</td>
<td>64</td>
</tr>
<tr>
<td>Gamecube</td>
<td>6</td>
<td>Nov. 2001</td>
<td>Nintendo</td>
<td>Nintendo</td>
<td>IBM PowerPC “Gekko” RISC @ 485 MHz</td>
<td>43</td>
</tr>
<tr>
<td>Xbox 360</td>
<td>7</td>
<td>Nov. 2005</td>
<td>Microsoft</td>
<td>Microsoft</td>
<td>IBM PowerPC “Xenon” RISC @ 3200 MHz with 3 Main Cores (3PPE)</td>
<td>512</td>
</tr>
<tr>
<td>Playstation 3</td>
<td>7</td>
<td>Nov. 2006</td>
<td>Sony</td>
<td>Sony</td>
<td>IBM PowerPC “Cell Processor” RISC @ 3200 MHz with 1 Main and 7 Specialized Cores (PPE + 7SPE)</td>
<td>512</td>
</tr>
<tr>
<td>Wii</td>
<td>7</td>
<td>Nov. 2006</td>
<td>Nintendo</td>
<td>Nintendo</td>
<td>IBM PowerPC “Broadway” RISC @729 MHz</td>
<td>88</td>
</tr>
</tbody>
</table>

Table 2. Architectures of Consoles in the US Video Game Industry (1995-2011; Study Sample Covers 1999-2011)

<table>
<thead>
<tr>
<th>Console</th>
<th>Generation</th>
<th>U.S. Launch Date</th>
<th>Platform Parent</th>
<th>Total Number of Processors</th>
<th>Requirement of Low Level Language Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>3DO</td>
<td>5</td>
<td>Oct. 1993</td>
<td>3DO (Licensor)</td>
<td>3</td>
<td>No</td>
</tr>
<tr>
<td>Jaguar</td>
<td>5</td>
<td>Nov. 1993</td>
<td>Atari</td>
<td>5</td>
<td>Yes</td>
</tr>
<tr>
<td>Saturn</td>
<td>5</td>
<td>May 1995</td>
<td>Sega</td>
<td>8</td>
<td>Yes</td>
</tr>
<tr>
<td>Playstation</td>
<td>5</td>
<td>Sep. 1995</td>
<td>Sony</td>
<td>2</td>
<td>No</td>
</tr>
<tr>
<td>Nintendo 64</td>
<td>5</td>
<td>Jun. 1996</td>
<td>Nintendo</td>
<td>3</td>
<td>Yes</td>
</tr>
<tr>
<td>Dreamcast</td>
<td>6</td>
<td>Sept. 1999</td>
<td>Sega</td>
<td>3</td>
<td>No</td>
</tr>
<tr>
<td>Playstation 2</td>
<td>6</td>
<td>Oct. 2000</td>
<td>Sony</td>
<td>5</td>
<td>Yes</td>
</tr>
<tr>
<td>Xbox</td>
<td>6</td>
<td>Nov. 2001</td>
<td>Microsoft</td>
<td>2</td>
<td>No</td>
</tr>
<tr>
<td>Gamecube</td>
<td>6</td>
<td>Nov. 2001</td>
<td>Nintendo</td>
<td>2</td>
<td>No</td>
</tr>
<tr>
<td>Xbox 360</td>
<td>7</td>
<td>Nov. 2005</td>
<td>Microsoft</td>
<td>4</td>
<td>No</td>
</tr>
<tr>
<td>Playstation 3</td>
<td>7</td>
<td>Nov. 2006</td>
<td>Sony</td>
<td>9</td>
<td>Yes</td>
</tr>
<tr>
<td>Wii</td>
<td>7</td>
<td>Nov. 2006</td>
<td>Nintendo</td>
<td>2</td>
<td>No</td>
</tr>
</tbody>
</table>
### Table 3. Descriptive Statistics and Correlation Matrix for the Sixth-Generation Console Sample

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>S.D.</th>
<th>Min</th>
<th>Max</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Title-Platform Quality</td>
<td>71.11</td>
<td>12.15</td>
<td>24.06</td>
<td>95.85</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Complex Platforma</td>
<td>0.42</td>
<td>0.49</td>
<td>0</td>
<td>1</td>
<td>-0.03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. In(Delay in Days)</td>
<td>0.88</td>
<td>1.76</td>
<td>0</td>
<td>6.91</td>
<td>0.01</td>
<td>-0.23**</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Average Quality of 1st Party Exclusives</td>
<td>77.58</td>
<td>3.49</td>
<td>55.29</td>
<td>85.11</td>
<td>0</td>
<td>-0.35**</td>
<td>-0.04</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. Average Quality of 3rd Party Exclusives</td>
<td>69.28</td>
<td>2.89</td>
<td>61.58</td>
<td>78.96</td>
<td>-0.05</td>
<td>0.16**</td>
<td>-0.13**</td>
<td>0.26**</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6. Inhouse Developmenta</td>
<td>0.51</td>
<td>0.5</td>
<td>0</td>
<td>1</td>
<td>0.38**</td>
<td>-0.01</td>
<td>-0.02</td>
<td>0.01</td>
<td>-0.01</td>
<td></td>
</tr>
<tr>
<td>7. Licensed Middlewarea</td>
<td>0.13</td>
<td>0.34</td>
<td>0</td>
<td>1</td>
<td>0.04</td>
<td>-0.01</td>
<td>0.04</td>
<td>0.02</td>
<td>0.02</td>
<td>0.08**</td>
</tr>
</tbody>
</table>

Note. N = 1413. aDummy variable. *p < 0.05; **p < 0.01.

### Table 4. Descriptive Statistics and Correlation Matrix for the Seventh-Generation Console Sample

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>S.D.</th>
<th>Min</th>
<th>Max</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Title-Platform Quality</td>
<td>71.39</td>
<td>13.25</td>
<td>16</td>
<td>97.04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Complex Platforma</td>
<td>0.42</td>
<td>0.49</td>
<td>0</td>
<td>1</td>
<td>0.04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. In(Delay in Days)</td>
<td>0.49</td>
<td>1.41</td>
<td>0</td>
<td>7.03</td>
<td>-0.01</td>
<td>0.1**</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Average Quality of 1st Party Exclusives</td>
<td>77.56</td>
<td>3.39</td>
<td>66.97</td>
<td>83.64</td>
<td>0.01</td>
<td>0.62**</td>
<td>-0.01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. Average Quality of 3rd Party Exclusives</td>
<td>68.78</td>
<td>4.75</td>
<td>58.36</td>
<td>75.32</td>
<td>0.07</td>
<td>0.91**</td>
<td>0.05</td>
<td>0.54**</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6. Inhouse Developmenta</td>
<td>0.61</td>
<td>0.49</td>
<td>0</td>
<td>1</td>
<td>0.37**</td>
<td>0</td>
<td>0.04</td>
<td>-0.03</td>
<td>0.01</td>
<td></td>
</tr>
<tr>
<td>7. Licensed Middlewarea</td>
<td>0.16</td>
<td>0.37</td>
<td>0</td>
<td>1</td>
<td>-0.09**</td>
<td>0.02</td>
<td>0.02</td>
<td>0</td>
<td>0.02</td>
<td>-0.05</td>
</tr>
</tbody>
</table>

Note. N = 806. aDummy variable. *p < 0.05; **p < 0.01.
### Table 5. Quality Differences across Complex vs. Non-Complex Platforms for the Sixth-Generation (Columns 5-1 to 5-5) and Seventh-Generation (Columns 5-6 to 5-10) Consoles. Fixed Effect OLS Regression Model with Games as Focus. DV = Review Score of the Game on the Focal Platform.

<table>
<thead>
<tr>
<th></th>
<th>Column 5-1</th>
<th>Column 5-2</th>
<th>Column 5-3</th>
<th>Column 5-4</th>
<th>Column 5-5</th>
<th>Column 5-6</th>
<th>Column 5-7</th>
<th>Column 5-8</th>
<th>Column 5-9</th>
<th>Column 5-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>ln(Delay in Days)</td>
<td>-0.631***</td>
<td>-0.679***</td>
<td>-0.741***</td>
<td>-0.741***</td>
<td>-0.747***</td>
<td>-0.651***</td>
<td>-0.614***</td>
<td>-0.991***</td>
<td>-0.993***</td>
<td>-0.989***</td>
</tr>
<tr>
<td></td>
<td>(0.061)</td>
<td>(0.063)</td>
<td>(0.070)</td>
<td>(0.070)</td>
<td>(0.116)</td>
<td>(0.117)</td>
<td>(0.171)</td>
<td>(0.172)</td>
<td>(0.172)</td>
<td></td>
</tr>
<tr>
<td>Average Quality of 1st Party Exclusives</td>
<td>0.067**</td>
<td>0.031</td>
<td>0.028</td>
<td>0.029</td>
<td>0.027</td>
<td>-0.068</td>
<td>0.006</td>
<td>0.059</td>
<td>0.066</td>
<td>0.066</td>
</tr>
<tr>
<td></td>
<td>(0.025)</td>
<td>(0.029)</td>
<td>(0.029)</td>
<td>(0.029)</td>
<td>(0.047)</td>
<td>(0.057)</td>
<td>(0.059)</td>
<td>(0.060)</td>
<td>(0.060)</td>
<td></td>
</tr>
<tr>
<td>Average Quality of 3rd Party Exclusives</td>
<td>-0.052</td>
<td>-0.024</td>
<td>-0.026</td>
<td>-0.026</td>
<td>-0.026</td>
<td>0.037</td>
<td>0.189**</td>
<td>0.203**</td>
<td>0.204**</td>
<td>0.206**</td>
</tr>
<tr>
<td></td>
<td>(0.035)</td>
<td>(0.037)</td>
<td>(0.037)</td>
<td>(0.037)</td>
<td>(0.036)</td>
<td>(0.072)</td>
<td>(0.072)</td>
<td>(0.072)</td>
<td>(0.072)</td>
<td></td>
</tr>
<tr>
<td>Complex Platform</td>
<td>-0.529**</td>
<td>-0.709**</td>
<td>-1.111***</td>
<td>-1.029***</td>
<td>-1.803*</td>
<td>-2.442**</td>
<td>-2.706**</td>
<td>-2.642**</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.196)</td>
<td>(0.216)</td>
<td>(0.272)</td>
<td>(0.275)</td>
<td>(0.770)</td>
<td>(0.792)</td>
<td>(0.873)</td>
<td>(0.878)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complex Platform x ln(Delay in Days)</td>
<td>0.293*</td>
<td>0.304*</td>
<td>0.314*</td>
<td>0.675**</td>
<td>0.680**</td>
<td>0.685**</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.146)</td>
<td>(0.146)</td>
<td>(0.145)</td>
<td>(0.227)</td>
<td>(0.227)</td>
<td>(0.228)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complex Platform x Inhouse Development</td>
<td>0.780*</td>
<td>0.829*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.323)</td>
<td>(0.323)</td>
<td></td>
<td>(0.496)</td>
<td>(0.497)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complex Platform x Licensed Middleware</td>
<td>-0.925^</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.478)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Constant</td>
<td>70.064***</td>
<td>71.166***</td>
<td>71.611***</td>
<td>71.620***</td>
<td>71.791***</td>
<td>74.401***</td>
<td>58.962***</td>
<td>54.154***</td>
<td>53.566***</td>
<td>53.406***</td>
</tr>
<tr>
<td></td>
<td>(2.876)</td>
<td>(2.894)</td>
<td>(2.897)</td>
<td>(2.889)</td>
<td>(3.034)</td>
<td>(7.246)</td>
<td>(7.363)</td>
<td>(7.411)</td>
<td>(7.419)</td>
<td></td>
</tr>
<tr>
<td>Observations</td>
<td>1,413</td>
<td>1,413</td>
<td>1,413</td>
<td>1,413</td>
<td>1,413</td>
<td>806</td>
<td>806</td>
<td>806</td>
<td>806</td>
<td>806</td>
</tr>
<tr>
<td>R-squared (within)</td>
<td>0.122</td>
<td>0.130</td>
<td>0.134</td>
<td>0.140</td>
<td>0.144</td>
<td>0.073</td>
<td>0.084</td>
<td>0.102</td>
<td>0.103</td>
<td>0.104</td>
</tr>
<tr>
<td>Number of Titles</td>
<td>612</td>
<td>612</td>
<td>612</td>
<td>612</td>
<td>612</td>
<td>361</td>
<td>361</td>
<td>361</td>
<td>361</td>
<td>361</td>
</tr>
<tr>
<td>Game FE</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Generation</td>
<td>Sixth</td>
<td>Sixth</td>
<td>Sixth</td>
<td>Sixth</td>
<td>Sixth</td>
<td>Seventh</td>
<td>Seventh</td>
<td>Seventh</td>
<td>Seventh</td>
<td>Seventh</td>
</tr>
</tbody>
</table>

Notes. Standard errors in parentheses. *** p<0.001, ** p<0.01, * p<0.05, ^ p<0.10
## Table 6. Robustness Checks

<table>
<thead>
<tr>
<th></th>
<th>6-1 Combined Sample</th>
<th>6-2 No 3rd Party Excl. Control</th>
<th>6-3 Clustered Std. Errors</th>
<th>6-4 Consoles (6th Gen.)</th>
<th>6-5 Consoles (7th Gen.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ln(Delay in Days)</td>
<td>-0.740***</td>
<td>-0.735***</td>
<td>-0.740***</td>
<td>-0.444***</td>
<td>-0.257^</td>
</tr>
<tr>
<td></td>
<td>(0.065)</td>
<td>(0.065)</td>
<td>(0.096)</td>
<td>(0.131)</td>
<td>(0.151)</td>
</tr>
<tr>
<td>Average Quality of 1st Party Exclusives</td>
<td>0.031</td>
<td>0.039^</td>
<td>0.031</td>
<td>0.045</td>
<td>0.146*</td>
</tr>
<tr>
<td></td>
<td>(0.024)</td>
<td>(0.021)</td>
<td>(0.022)</td>
<td>(0.030)</td>
<td>(0.062)</td>
</tr>
<tr>
<td>Average Quality of 3rd Party Exclusives</td>
<td>0.021</td>
<td>0.021</td>
<td>-0.031</td>
<td>0.124^</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.028)</td>
<td>(0.021)</td>
<td>(0.034)</td>
<td>(0.037)</td>
<td>(0.073)</td>
</tr>
<tr>
<td>Complex Platform</td>
<td>-0.993***</td>
<td>-0.922***</td>
<td>-0.993***</td>
<td>(0.034)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.238)</td>
<td>(0.218)</td>
<td>(0.233)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complex Platform x ln(Delay in Days)</td>
<td>0.339**</td>
<td>0.337**</td>
<td>0.339*</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.117)</td>
<td>(0.117)</td>
<td>(0.159)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complex Platform x Inhouse Development</td>
<td>0.652*</td>
<td>0.668*</td>
<td>0.652**</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.270)</td>
<td>(0.269)</td>
<td>(0.251)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complex Platform x Licensed Middleware</td>
<td>-0.592</td>
<td>-0.582</td>
<td>-0.592</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.383)</td>
<td>(0.383)</td>
<td>(0.381)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gamecube (6th)/Wii (7th)</td>
<td></td>
<td></td>
<td></td>
<td>1.344***</td>
<td>0.894</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(0.363)</td>
<td>(1.086)</td>
</tr>
<tr>
<td>Xbox (6th)/Xbox 360 (7th)</td>
<td></td>
<td></td>
<td></td>
<td>0.757*</td>
<td>2.574**</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(0.312)</td>
<td>(0.872)</td>
</tr>
<tr>
<td>Gamecube/Wii x ln(Delay in Days)</td>
<td>-0.432**</td>
<td>-0.749**</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(0.158)</td>
<td>(0.251)</td>
</tr>
<tr>
<td>Xbox/Xbox 360 x ln(Delay in Days)</td>
<td>-0.185</td>
<td>-0.226</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(0.162)</td>
<td>(0.308)</td>
</tr>
<tr>
<td>Gamecube/Wii x Inhouse Development</td>
<td>-1.108**</td>
<td>-0.102</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(0.426)</td>
<td>(0.866)</td>
</tr>
<tr>
<td>Xbox/Xbox 360 x Inhouse Development</td>
<td>-0.688^</td>
<td>-0.403</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(0.367)</td>
<td>(0.503)</td>
</tr>
<tr>
<td>Gamecube/Wii x Licensed Middleware</td>
<td>0.730</td>
<td>-1.366</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(0.623)</td>
<td>(1.414)</td>
</tr>
<tr>
<td>Xbox/Xbox 360 x Licensed Middleware</td>
<td>1.033^</td>
<td>0.404</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(0.536)</td>
<td>(0.643)</td>
</tr>
<tr>
<td>Constant</td>
<td>68.096***</td>
<td>68.863***</td>
<td>68.096***</td>
<td>69.966***</td>
<td>50.644***</td>
</tr>
<tr>
<td></td>
<td>(1.933)</td>
<td>(1.640)</td>
<td>(1.823)</td>
<td>(2.934)</td>
<td>(8.078)</td>
</tr>
<tr>
<td>Observations</td>
<td>2,178</td>
<td>2,178</td>
<td>2,178</td>
<td>1,413</td>
<td>806</td>
</tr>
<tr>
<td>R-squared (within)</td>
<td>0.111</td>
<td>0.111</td>
<td>0.111</td>
<td>0.150</td>
<td>0.148</td>
</tr>
<tr>
<td>Number of Titles</td>
<td>951</td>
<td>951</td>
<td>951</td>
<td>612</td>
<td>361</td>
</tr>
<tr>
<td>Game FE</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reference Category</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Playstation 2</td>
<td>Playstation 3</td>
</tr>
</tbody>
</table>

Notes. Standard errors in parentheses. *** p<0.001, ** p<0.01, * p<0.05, ^ p<0.10
Figure 1. The Direct and Moderated Impact of Platform Complexity on Multihoming Complement Quality

Figure 2. Average Project Size of Development Teams and the Share of Multihoming Title-Platforms in the U.S. Console Games Industry (1990-2010)

Figure 3. Yearly Number of Third Party Game Releases for Fifth-Generation Consoles

Figure 4. Yearly Number of Third Party Game Releases for Seventh-Generation Consoles

Figure 5. Xbox 360 Architectural Diagram

Figure 6. Playstation 3 Architectural Diagram
Figure 7. Complex Platform x ln(Delay in Days): Predicted Means with Confidence Intervals for Sixth-Generation Consoles

Figure 8. Complex Platform x ln(Delay in Days): Average Marginal Effect of Complex Platform=1 with Confidence Intervals for Sixth-Generation Consoles

Figure 9. Complex Platform x ln(Delay in Days): Predicted Means with Confidence Intervals for Seventh-Generation Consoles

Figure 10. Complex Platform x ln(Delay in Days): Average Marginal Effect of Complex Platform=1 with Confidence Intervals for Seventh-Generation Consoles