

Why Do New Platform Technologies Fail? The Paradox of Technological Superiority

Abstract

Technological superiority provides more functional value for users, and is a critical dimension that challengers can leverage to dethrone incumbent technological platforms. We argue that while technological superiority might help accelerating adoption on one side of the market by driving early user adoption, it also causes greater innovation challenges for complementors who face time compression diseconomies due to the learning of the new technological environment. This limits their contribution of complementary products, which creates unfavorable expectations and slows down platform adoption. Eventually, these platforms fail. We develop this logic conceptually through a qualitative analysis of the US video game console industry. A number of studies have documented a puzzling result that it is often the inferior technology to become the dominant standard. In the context of platform markets, our study highlights that what is perceived as ‘superior’ from a technological perspective is in fact ‘inferior’ from an ecosystem perspective that accounts also for complementors.

“We recognize that our technical architecture has initially made Sega Saturn more difficult to develop for than other next generation formats, including the Playstation. But that is also why we know that Sega Saturn is a superior gaming platform”.

– Tom Kalinske, CEO of Sega of America, August 1995

INTRODUCTION

Many high-tech markets are nowadays organized around platform technologies, which provide value to final customers not only through their standalone functionalities, but also by allowing third-party firms to build upon and develop complementary products that extend the reach and value of the core platform technology to users (see Gawer 2014 for a review). In essence, platform technologies give rise to standards at the industry level, which form a system of use (David & Greenstein, 1990; Katz & Shapiro, 1994), whereby final customers benefit from using the platform in conjunction with its stable of complements (Choi, 1994).

The literature on technological standards battles (see Suarez, 2004 for a review), which are analyzed under labels of dominant designs (e.g., Anderson and Tushman, 1990), technological trajectories (e.g., Dosi, 1982), standards (e.g., Katz and Shapiro, 1985), or platforms (Gallagher and Park, 2002), has highlighted different factors that may lead a technology to win (or being locked out from the market), with particular emphasis on technological features such as compatibility with the dominant standard and technological superiority (e.g., Rosenbloom and Cusumano, 1987; Schilling, 2002, 2003; Sheremata, 2004; Suarez, 2004). In particular, technological superiority – “how well a given technology performs vis-à-vis competing alternatives” (Suarez 2004: 276) – has been considered a critical success factor for new technologies aspiring to dethrone the dominant technological standard (Sheremata 2004; Suarez & Kirtley 2012; Schilling, 2003; Christensen, Suarez, and Utterback, 1998). By providing more benefits to final customers that compensate the costs they bear to switch from the

incumbent to the new technology, technological superiority may generate an “innovation shock” (Argyres et al. 2015) in demand and induce adoption of the new technology.

Consistent with this “user perspective”, the economics literature on platform competition has highlighted the importance of a platform’s user installed base as a primary factor for a platform technology to eventually emerge as the dominant platform in the market – which has been referred to as the *winner-takes-all* hypothesis (Armstrong, 2006; Lee et al., 2006). The focus in this literature is generally on the importance of network externalities – adoption of the platform by users creates positive feedback effects for developers of complementary products, inducing other users to adopt the platform (Eisenmann et al. 2006; Evans 2003; Rochet & Tirole 2006). In this regard, technological superiority of a platform would provide direct benefits for users to adopt the new technology, and may generate early lead in the market and thus increase the likelihood of success.

Both literatures thus extensively focus on the users’ benefits as major driver of a technology success, fuelled by “bandwagon effects” (Katz & Shapiro, 1992), “community support” (Wade 1995) or “winner take all” (Eisenmann et al. 2006). An implicit assumption in the two-sided platform literature is that, when a new platform arrives to the market, complementors may be ready to build complementary products for it as long as users adopt the new technology. The problem, in other words, is a market problem: implementing market penetration strategies to create positive network externalities. Designing a technological platform such that its functionalities offer enough value for users to start adopting it would help jump start this adoption process. Yet, evidence exists in the standards literature that often is the “inferior” technology that wins, usually because of wider support from complementors (Anderson et al. 2014; Rosenbloom & Cusumano, 1987; Tushman and Anderson, 1986; Wade, 1995). Why do

complementors choose to support an “inferior” technology remains though a puzzle, challenging the major theoretical views in the literature. Scant work exists on how these “communities” or networks of complementors emerge and evolve around the core technology. A small set of studies has recently shown that when complementors face innovation challenges, the core technology may face adoption growth constraints (Adner & Kapoor 2010; Cennamo & Santaló, 2013). In other words, superior technologies may still fail because of limited availability of complements (Schilling 2002). Missing though is an understanding of why this happens.

In this study we argue that technological superiority entails “shadow costs” for complementors, particularly in terms of learning and commitment of resources under high uncertainty, which limit value creation from the supply-side of the platform market. Since we are interested in the emergence and growth of the whole platform ecosystem, we focus on the early market stages of challengers’ new platform technology vis-à-vis incumbents’ where “windows of opportunity” can be leveraged by technological leapfroggers (Christensen et al., 1998; Schilling, 2003). We argue below that the higher the technological gap between the new generation and the incumbent platforms, the higher the challenges for complementors to embrace the new platform technology and build complements for it. This would eventually put a brake on the subsequent adoption of the platform, which, despite early momentum, may fail to reach the mass market. Instead of looking at the aspects that enable the platform to gain momentum we rather focus our attention on what limits the capacity to sustain and leverage such momentum and turn the platform into a mass-market product. The research question we try to answer is thus: *why do new-generation platform technologies fail despite early momentum?*

We start by revisiting the concept of technological superiority. So far, this concept has been treated mainly in terms of benefits to users. A noticeable exception, Anderson, Parker and Tan (2014) discuss, and formally model, the potential tradeoffs platform owners face between investing in the technology to provide more functional benefits to users and investing in tools to benefit complementors by facilitating their production of complements. Building on this insight, and other studies from the information systems literature that point to increased programming complexity and learning costs that may be associated with superior technologies (e.g., Baldwin, MacCormack, and Rusnak, 2014; Venkatesh, 2000; Xu et al., 2010), we expand the concept of technological superiority to also include benefits (or constraints) that the new technology offers to complementors.

We then explore this revisited construct in the context of the video game industry, which shows the introduction of different platform technology generations, high variance in platform leadership and several failures of platforms pioneering the new generation of the platform technology. Many of these platforms had some user base and complements available, showing that they gained momentum; nevertheless, they fail. Building on our reading of these failures, we develop a conceptual framework identifying three factors – time compression diseconomies, resource commitment, and lock-in problems – affecting complementors’ learning of the new technology programming environment, and thus co-specialization and value creation, which consistently contributed to platform failure of the different players in the industry. This was particularly the case when the functional gap with the previous generation technology was higher. In fact, one of the major finding from our study is that breaking with the current technology trajectory by moving up the frontier of the technology may be major obstacle for the emergence of the ecosystem. Technological superiority, while it might enable innovation shocks

on the user demand side, might represent a “shadow barrier” for complementors’ support on the complement supply side.

THEORETICAL BACKGROUND

Technological Superiority

Dominant design (Utterback and Abernathy, 1975; Tushman and Anderson, 1986) emerges through complex relationships between social and economic factors. One of the most fundamental firm-level factors considered in technological battles for dominance is the superiority of a technology (e.g., Schilling, 1998, 2003; Suarez, 2004). Although technological superiority alone is not a sure mean to win the battle for technological dominance, it has been conceptualized as an important firm-level advantage in the battles for dominance. For example, Suarez (2004: p.276) argues that: “*Other things being equal* [emphasis added], the better a technology performs with respect to competing technologies, the higher the likelihood that it will become dominant.” Schilling (2003: p.17) claims that: “...if the new entrant is unable (or unwilling) to make its technology compatible with the existing standard... functionality advantage must offer *so much value* [emphasis added] to the customer that it exceeds the combination of functionality installed base, and complementary goods value offered by the incumbent technology standard.”

A specific interest in studies on technological dominance has been in industries where network externalities are prevalent (Katz & Shapiro, 1986). Such positive consumption externalities mean that the value derived by a user increases with the number of other users. Factors that determine success and failure in technological battles may often have different implications when network externalities are involved in (Schilling, 2002). It has been argued that in an industry where network effects are present, early entry is more rewarding in case of

technological superiority as it will help to draw users early on to the platform. Sheremata (2004) advance that investing in radical innovation that would enhance the functionality gap of the new technology compared to incumbents as the best option for new entrants to dethrone incumbent standards. Schilling (1998: p.278) theorizes that, "...when a technology yields a dramatic improvement over previous generations or different technologies that serve similar functions, that technology will more rapidly gain customer acceptance". Studies on network externalities highlight the importance of gaining early user installed base, which can be interpreted as "a reflection of its [i.e., technology] intrinsic value - or, alternatively, as an indication of the "brand" value of being associated with a superior product" (Brynjolfsson & Kemerer, 1996: 1629).

Most studies have conceptualized superiority of the technology mainly in terms of technical features that affects the output of a technology. Since focus is on the output, this essentially implies that technological superiority could be a key driver for user adoption of the technology. This shall be the case when technology has value on its own. Yet, when it requires additional components or complements to generate full benefits for the final customer, technical features are not sufficient as they represent only one part of the whole system (Adner & Kapoor, 2010; Anderson et al. 2014; Xu et al. 2010). Nonetheless, authors have advanced that even in these contexts, since new technologies bear a disadvantage in terms of user installed base or complements, they will have very limited chances of succeeding unless they provide users with greater functionalities (i.e., technology superiority) that compensate for the lack of installed base or complements (e.g., Schilling, 2003; Sheremata 2004).

Technological Superiority Revisited

Platform technology innovations do not take place in a vacuum; their success depends on other actions of ecosystem members (Adner and Kapoor, 2010), which include users but also and importantly complementors. According to such an ecosystem view, value creation from platform innovation is severely reduced if complementors experience innovation challenges in supporting the platform (Anderson et al. 2014; Cennamo & Santaló 2013). Technology superiority of the platform may thus have limited or no value to consumers if complementary innovation is lacking because of complementors' innovation challenges.

In these contexts, where technology is only one component within a system of use, albeit a central one (Gawer 2014), when assessing the superiority of a technology *system* one should not only consider how the key attributes of the core technology benefit users (compared to incumbent technologies) but also how these attributes benefit (or hamper) complementors' ability to respond timely to such game changer innovation and build compelling complementary products. Accordingly, we conceptualize technological superiority not only in terms of user (dis)functionalities but also in terms of *complementor (dis)functionalities*. We are grounding our ideas on two established literatures: technology acceptance model in information systems literature (e.g., Venkatesh, 2000; Xu et al. 2010), and complexity originated in systems theory literature (e.g., Simon, 1962; Baldwin, MacCormack, and Rusnak, 2014).

Technology acceptance model, originally developed by Venkatesh (2000), and refined in later studies (Venkatesh & Davis, 2000; Venkatesh & Bala, 2008), explains how users will adopt and use a technology. Two main factors on the user side are identified as critical for adoption of a new technology: (i) perceived usefulness, and (ii) perceived ease of use. Perceived usefulness is the degree to which an individual believes using the technology help that individual to reach his/her goal (Venkatesh and Davis, 2000). The greater the expected gains, the greater the intent

to use the new technology. Perceived ease of use is the degree to which using the new technology is free of effort (Venkatesh and Davis, 1996).

Recently, this model has been also used to understand when users will switch to a new platform technology (Xu, Venkatesh, Tam, and Hong, 2010): usefulness, enjoyment of use, and perceived superiority of services obtained from the platform have been highlighted as important factors. In general, this literature confirms previous conceptualizations of user-related technological superiority and its role in user adoption.

On the other hand, literature on system theory explores how connections and hierarchies forming the system itself create complexity (Baldwin, MacCormack, and Rusnak, 2014). Complexity of architectures are generally measured by identifying linkages existing between different elements in the system (Simon, 1962; Alexander, 1964) and it has been thought to increase as number of interconnected components and subsystems increase (Dibiaggio, 2007). As complexity, therefore linkages between parts of the system, increases, it becomes more costly and harder to understand and verify the workings of a given system, due to the nested hierarchy of decisions required for performing the task. Indeed, in software engineering, complexity has been defined as: “the degree to which a system or component has a design or implementation that is difficult to understand and verify” (IEEE Standards Board, 1990) and it is measured by lines of code (LoC) or other measures based on relationships between components in a system (Lagerstrom, Baldwin, MacCormack, and Dreyfus, 2014).

This literature on complexity provides us with the important insight that a system’s architecture may involve higher costs for those that need to use the core technology of the system, or platform, to create complementary goods for users. For instance, in a recent theoretical model, Anderson et al. (2014) advance that investing in technical functionalities to

enhance performance of a new platform technology could be a selling point for users, but would also make the system more complex and require developers to make large investments to produce complements for these new platform technologies.

Building on these studies, we conceptualize superiority of a technology system in terms of the benefits it offers to both users and complementors. In particular, we consider technology's usefulness, enjoyment, superiority of services and hardware capabilities/potential as user-related benefits, and a technology's number of lines of codes, interconnected components, and development cost per unit performance as indicators of system complexity, thus technology's (dis)functionalities for complementors. Table 1 provides a list summary of various definitions of user and complementor (dis)functionalities.

Insert Table 1 about here

RESEARCH METHODOLOGY

Due to limited theory about platform emergence and evolution (Jacobides, Cennamo, and Gawer 2015) and why platform innovations fail, we conducted an inductive multiple case study (Eisenhardt, 1989; Yin, 1994). Such an inductive study is useful when existing theories fall short of answering the existing question and especially in cases question is related to a process or evolution over time (Langley, 1999; Hannah, 2014).

We chose to study video game consoles, such as Sony's PlayStation or Sega's Genesis, which serves as platforms on which game titles are developed by complementors (on the sellers' side) and these are consumed by (played by) users (on the buyers' side). First, they are a quintessential example of platform markets that shows very strong support for cross-sides network effects and pricing strategies (e.g., Clements and Ohashi 2005). Second, they are

technical devices, so they allow us to have a variance on their technological superiority and how it affects network emergence in earlier stages of competition. Moreover, having quantifiable performance differences between consoles allows us to objectively assess how each console is positioned in terms of user and developer (dis)functionalities. Third, there have been multiple generations of platform releases and several changes in market leadership that allows us to separate challengers that “win” from challengers that “lose”. Fourth, availability of performance measures in terms of installed base and game releases allows us to easily track each console’s evolution and their eventual failure or dominance.

Our analysis covers all major consoles released from 1988 until 2012 – this excludes current generation of consoles recently initiated with the release of Wii U by Nintendo. During this time period, a total of 20 consoles (including 2 console add-ons released by Sega) have been released. We have treated each console as a separate platform case, and tracked its evolution, performance as well as technical characteristics. Video game consoles have been historically divided into different generations based on the word instruction length (in bits), CPU speed and amount of RAM (de Vaan, 2014; Forster, 2005). Table 2 lists these consoles, according to their technological generation.

Insert Table 2 about here

Data Sources

We relied on several data sources to understand the evolution of each platform in our sample, and their eventual performance: (1) books that have documented the history of the video game industry as well as individual console producers (Kent, 2001; Pettus, 2013; Harris, 2014); (2) *retrospective* section of the various issues of *Retrogamer Magazine* where each platform in our sample is explored in-depth through interviews done with key managers and game

developers that have worked for each platform; (3)¹ publicly available information in video game industry websites such as arstechnica.com, 1UP.com, Gamespot.com, Gamasutra.com, IGN.com as well as other publicly accessible information such as official websites (especially for technical specifications), major press articles (e.g., *The Economist*), focused fan sites that document specific platforms including interviews with key managers and developers (eg., www.Sega-16.com) and Usenet groups archived within Google repositories; (4) Game Documentation and Review Project Mobygames.com for tracking game releases and its characteristics for each platform; and (5) platform sales data from NPD research group as well as official information shared by platform providers. Last but not least, we have also used (6) historical data provided by previous research articles on video game consoles (Schilling, 2002; Corts & Lederman, 2009; Clements & Ohashi, 2009). This allows us to compare what has been reported on historical events and *ex-post* stories of these managers, which allow us to evade both *ex-post* rationalization and deducing wrongly from history of events.

In general we relied more on video game related sources as major press tend to focus on most known aspects of the industry (e.g., pricing, winners, latest hits etc.). In particular, interviews with key executives and developers, mostly within the retrospective section of Retrogamer and Video Game websites allowed us to understand perspectives of both market issues as well as development related issues. We also accessed any other publicly available yearly reports and 10-K reports to triangulate our findings. All these in-depth sources of archival data allowed us to build a comprehensive and accurate history of each platform in our sample.

Data Analysis

¹ In case a source on the internet was not accessible, we have reached the website by using Internet Wayback Machine (<http://archive.org/web/>)

We have started data analysis by combining our various data to build the historical case of each platform in our sample. In combining these data, we directed our attention and understanding on the emergence and evolution of each platform system and the factors that caused them to fail (or succeed). We focused in particular on important events and choices related to user and developer sides of each platform, and competitive dynamics between old and new-generation platforms, as well as between platforms of same generation.

We identified three main constructs to categorize problems of failing platforms: (i) time compression diseconomies, (ii) resource allocation problems, and (iii) lock-in problems. Time compression diseconomies will be present due to impossibility of compressing learning in a short time period even with an increased allocation of resources (Dierickx and Cool, 1989). Resource allocation problems will be present due to less current value of the new platform for the complementor, inducing the complementor to allocate second-tier resources for the platform. Lock-in problems are simply the case that, although the market uncertainty is solved to some extent with users adopting, the complementor will be unsure if it pays off to invest and learn for this new platform, and in case it doesn't work, there will be an increased risk of lock-in.

Measures

We needed two kinds of measures to test our theory. On one hand, we required to show the performance of various platforms in order to see if they have failed or succeeded. For this we relied on the main measure of success of a platform: installed user base at a given point of time (relative to competing systems).

On the other hand, we needed to also map the positioning of consoles in terms of user and developer (dis)functionalities. In order to do so, we have first collected data about the technical specifications of each platform. As studies on technology adoption suggest that benefits for users

mainly come from perceived usefulness and power performance, we have used main technical components of a platform, consisting of its CPU, Graphics Card, RAM, Sound Card and its media format (cartridge, CD-ROM, DVD-ROM etc.) to capture benefits for users. Capturing platform (dis)functionalities for developers requires identifying factors that reveal in our context how difficult for developers to program games for the whole platform system. With this in mind, we have captured complexity of a given console with 3 main specific characteristics: (1) Total Number of Non-Sound Processors, (2) Requirement of using parallel processing to fully utilize the hardware, and (3) the default development environment for the platform. Difficulties of programming due to number of non-sound processors and using parallel processing are similar; it increases the need to coordinate the flow of processes undertaken by either various chips in the case of number of processors, or central processing units in the case of parallel processing. This in turn, increases complexity, as we highlighted above that complexity increases as the linkages and connections between different nodes of a system increases. Development environment, on the other hand, affect the difficulty of development either by requiring the programmer to code everything from scratch, or allowing an easier interface where developers could use ready-made tools instead of coding at length (an analogous comparison could be made by considering web page design through HTML coding vs. using a WYSIWYG "What You See Is What You Get" web page creator program where you click and drop elements of a page). For each generation, we map the technical specifications of each console along these two dimensions – user functionalities and developer (dis)functionalities.

TECHNOLOGICAL LEAPFROGGING IN THE US VIDEOGAME SECTOR

Fourth Generation Consoles (1989 – 1993)

The fourth generation has been characterized by improved graphics, sound, and computing capabilities (as can be seen by comparing 4th generation consoles with 3rd generation in Table 4), as well as by the start of well known “bit wars” (Harris, 2014). At the beginning of this generation, Nintendo was dominating the market with its third generation console, NES (Nintendo Entertainment System), having virtually all developers working for its platform. In the fourth generation, there were 3 consoles and 2 console add-ons releases. Early challengers were NEC and Sega, respectively with their Turbografx-16 and Genesis consoles. Subsequent releases of new consoles include Nintendo’s SNES (Super Nintendo Entertainment System) and 2 add-on releases by Sega to improve the technical features and market position of Genesis: Sega-CD and 32X.

Turbografx-16 and Genesis were released around the same time in US, August 1989, both trying to challenge the incumbent NES by technologically leapfrogging it. Though Turbografx-16 was considered a 16-bit console due to its graphic processors, its main processor was essentially 8-bit, as NES, but had four times the clock speed and main memory of the NES. Most important technical edge of Turbografx-16 was its graphics capabilities. Its 2 special graphics processors were able to show up to 482 colors on screen at a time, compared to 25 colors of NES. On the other hand, Genesis was the most advanced console in overall until Nintendo responded by releasing SNES two years later. As can be seen from Tables 2 and 3, NES was based on an 8-bit 1.8 MHz speed processor and had 0.002 MB Ram, while Genesis had a 16-bit 7.6 MHz processor and had 0.064 MB Rams. Genesis was able to show 64 colors, drastically improving upon NES’ graphic capabilities. Also, Genesis had the same processor 8-bit processor of its older generation Sega Master System, and could use it for full backwards compatibility.

SNES arrived as a late response by Nintendo in August 1991, and it offered major improvements with respect to all other consoles. It had a 16-bit 3.58 MHz processor, only outmatched by Genesis, but it was able to show 256 colors out of 32,768 color palette, giving the highest color palette available for games –the actual number of colors on screen was only outmatched by Turbografx-16. It had double main memory of its close rival, Genesis, and it offered the most advanced sound capabilities in a console by that time (sound processor was developed together with Sony, which eventually leveraged this acquired knowledge of the industry to develop its own console, PlayStation, later on).

Sega released Sega-CD and 32X add-ons (October 1992 and November 1994 respectively) to enhance capabilities of Genesis console. Sega-CD was an early attempt to use CD as a media to store games, instead of cartridges, but it added minimal technical functionalities to the core console technology (and therefore we do not consider it for our analysis). Released at the end of the fourth generation, 32X had dual 32-bit processors, over 32,000 colors and four times the memory of SNES.

Main consoles in this generation are characterized by similar development requirements through assembler heavy programming. An assembly language is a specific type of low-level computer programming language, specific to each console, whose knowledge is fundamental for the game developers to build a game for the console². Although these consoles represented an important jump with respect to 8-bit consoles they replaced, such as NES, there were 2 reasons such jump didn't ensue into important developer disfunctionalities, especially for early challengers. First, NES was technically outdated by the time early challengers came at 1989, and

² 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).

developers had already a good amount of experience with programming for 16-bit processors. Both Atari ST and Amiga 500 personal computers, which were based on the same 16-bit processor used in Sega's Genesis, were released around 1985-1986 and they were highly popular for their video games respectively in US and Europe. This allowed many developers to have experience with 16-bit computer-based game development by the time challengers arrived to the console market with similar developing technology. Second, early challengers Turbograft-16 and Genesis, both had to sustain their game development activities internally when they were released since NES had most of the developers tied in exclusivity contracts that forbid them to switch to competing consoles for two years (Evans, Hagi, and Schamalansee, 2006). By the time developers switched to these challenger consoles, this jump was quite incremental for their skills. For all these reasons, developers needed to make equal learning investments for developing games to any console in this generation, meaning that they incurred the same cost and difficulty for development whichever new console of fourth generation they choose. Since developer (dis)functionalities were equal between these consoles, merits of these systems were based exclusively on their user functionalities, which meant their technical power.

Looking at add-ons, Sega 32-X is an interesting occurrence. The next generation console by Sega, Saturn, was released in Japan the day 32X, the add-on for the current console, was released in US. Saturn was going to be released in US just one year after Japan, giving 32X only one year to go. This proved detrimental for the prospects of the add-on for two reasons. First, when a new programming technology is released, developers need to learn a new set of capabilities, which can be cumbersome as evident in the following quote: *"we're still on the 12 month cycle. Every time there's a new hardware generation, we have huge struggles making it work"* (Anderson et al., 2014: 154). Second, because of these difficulties and the limited residual

life of the console, many developers ended up not using the additional hardware capabilities provided by the add-on. Sega of America's vice president of technology at the time, Marty Franz recalls in an interview:

"I think the real issue was timing [emphasis added]; the games in the queue were effectively jammed into a box as fast as possible, which meant massive cutting of corners in every conceivable way. Even from the outset, designs of those games were deliberately conservative because of the time crunch. By the time they shipped they were even more conservative; they did nothing to show off what the hardware was capable of" (McFerran, 2012; p.243).

32X offered tremendous capabilities but, because of timing issues, developers could not invest into new learning and experimentation, rather pressed to have games completed as fast as possible. This problem can be also detected in the same interview, when discussing why 32Xs' sound system has never been utilized:

"Developing a new audio engine was probably deemed not worth the investment by developers. The 32X was destined to be a short lifespan product. The developers put the dollars to the screen and saved money by not enhancing the audio" (McFerran, 2012: 243).

Performance Implications

At the end of the 1992, Sega sold 6.9 millions of Genesis, while Nintendo sold 4.2 millions (Shankar and Bayus, 2003) of SNES, and NEC's Turbografx-16 ended up selling only around 2 million³. As an add-on, Sega 32X was sold only a little above 650,000 for over 8 million Genesis user base. Towards the end of the generation, SNES trumped Genesis in sales. A similar pattern in performance can be also seen in terms of quantity and quality of game releases. Genesis and SNES offered a high and similar number of critically acclaimed hits, 59 and 57, respectively, though game sales were higher for SNES (46 titles with over 1 million sales). Turbografx-16 on the other hand, had only 7 critically acclaimed hits (that received low sales due

³ <http://www.usgamer.net/articles/turbografx-16-at-25-remembering-the-little-pc-engine-that-could>

to low installed user base of the console). In terms of the quantity of game releases, SNES had close to 650 third-party games, while Genesis followed it with 513. On the other hand, Turbografx-16 had only 32 third-party game releases. Contrary to the main console, Genesis, 32X had a dismal market performance. In total, there were 35 game releases, and 19 of these were just minimally modified versions of games made for Genesis. Table 3 lists information on developer disfunctionalities and game release performance.

Insert Table 3 about here

Why did SNES and Genesis succeed? Simply put, they both offered technological superiority in the classical sense the literature has referred to –in terms of user functionality. Since consoles in this generation had minimal and similar developer disfunctionalities, technological superiority were mainly based on the user functionality dimension. In 1989, Genesis was the superior console, and led the market until early 1993, while SNES was the superior console after its release in 1991, and gained market leadership through 1994.

Why did Turbografx-16 and 32X failed? Compared to Genesis, which released around the same time, Turbografx-16 was a technically inferior console offering more limited user benefits. On the other hand, 32X proved to be “inferior” because of developer disfunctionalities. On a technological prowess basis, 32X was clearly superior to any fourth generation console, being in fact closer to fifth generation consoles. Yet, its’ problems with developers left this add-on with most of its power untapped. Essentially, game developers found it difficult to timely learn about how to effectively use this new programming environment and eschewed console-specific commitment of resources.

In sum, two main points shall be highlighted from the battle for the fourth generation console. One, absent developer (dis)functionalities, indeed technological superiority is critical factor in challenging incumbent platforms, and succeeding in the market, as it proved to be a powerful trigger of early user adoption. Two, Sega's 32X experience gives us the first glimpse of how developer (dis)functionalities could trigger problems that could nullify any value driven by user functionalities with technological superiority.

Fifth Generation Consoles (1993 – 1999)

This generation has influenced the Video Game Industry in a critical way. First, CD-ROM was a major improvement compared to ROM cartridges used by the consoles until that point. It offered more than 600 MB's of data storage while an average cartridge was offering up to 16MB's of data. Moreover, CD-ROM was much cheaper to produce, therefore reducing costs for developers as they generally had to pay to the platform owner such production costs (e.g., Nintendo was asking for paid minimum production limit to developers). As CD-ROM drives become affordable for the mass-market, it opened up new possibilities for developers to create new experiences. Second, development of hardware with 3D graphics capability was a major change in game designs, as 3D gaming has always been aspirational for platform owners and developers. Third, this generation represented the most technologically turbulent period with the highest number of entrants and competing consoles in the industry (de Vaan, 2014), each having their own quite different technology designs.

Consoles in this generation could be separated in two groups: pioneering challengers, 3DO and Atari, and following consoles by Sega, Sony, and Nintendo. 3DO's 3DO Interactive Multiplayer (3DO) and Atari's Jaguar were released in October and November 1993, respectively, starting the Fifth Generation Console era.

3DO was more powerful than any other console in the market at the time of its release. It had a 32-bit RISC processor (a first for a console) with 12.5 MHz speed made by ARM, the main processor was also supported with a custom designed math co-processor to take the burden of repetitive calculations from the main processor. For graphics, it used 2 accelerated graphics co-processors, capable of advanced 3D capabilities for its own time. 3DO was also the first console that had CD-ROM only as a media; instead of cartridges that have been common previously (CD was only offered as an add-on in the previous consoles).

Atari's Jaguar was a 64-bit console (although contested) even when the first 32-bit machine was just released a month earlier (3DO) and it aimed to capture market through its hardware capabilities combined with low price for gamers. Jaguar featured a dual main processor structure with each working at 26.6 MHz and these two processors supported by a 16-bit general purpose processor, object processor, and a "blitter" coprocessor⁴. In total, there were five processors within Jaguar.

Sega Saturn was developed based on Sega's back then cutting edge arcade video game technology and upcoming Playstation's advanced 3D technical capabilities. Similar to Jaguar, Saturn also adopted a dual main processor structure, having two 32-bit 28.6 MHz processors. It also had 2 video processors, each with separate capabilities. Sega Saturn is generally considered to have a complex architecture, as it had eight processors making up the console.⁵

Sony's Playstation was a simple but powerful design that focused on 3D gaming. It had a single 32-bit processor with 38.8 MHz speed, and it had two specialized co-processors with the

⁴ Blitter is a specific type of coprocessor that helps with fast movement of graphical data through memory, and it was first invented for Amiga personal computers. <http://www.google.com/patents/US4874164>.

⁵ "Saturn was a complex beast that used twin CPUs and another couple of video processing chips" <http://www.theguardian.com/technology/2015/may/14/sega-saturn-how-one-decision-destroyed-playstations-greatest-rival>, accessed 25 May 2015.

CPU, the one named “Geometry Transformation Engine” giving an edge to Playstation in 3D graphics when combined with Playstation’s main graphics card.

Nintendo 64 was the latest entrant to the Fifth Generation, and had the most technologically advanced console. It was the first console to have all 64-bit processor, running at 93.75 MHz. It also had the “Reality Coprocessor” which featured processors for its graphics capabilities. Perhaps the only part where Nintendo 64 was lagging behind competition was in its choice of using cartridges instead of CDs as a media. This was driven by two main reasons: on one hand cartridges allowed faster load time during playing games, while on the other hand Nintendo wanted to stop rampant piracy through CDs in that era.

3DO, the pioneer of the new console generation, struggled to obtain games from third-party developers by the time of its launch and the next follow-up months. Switching from 16-bit consoles to 32-bit consoles was indeed difficult for developers. Trip Hawkins, founder of 3DO, recently recalls:

“... but in fairness game developers were dealing with many new things and issues on this type of hardware... it turned out that even though many of the game developers were sure ... that they going to make [games] for 1993 holiday season ... didn’t really get to market until summer of 1994 ... You could make the argument that that was the fatal blow” (Matthews, 2013; p.27).

As a result of these delays there was only 1 title (Crash ‘n Burn) when 3DO was released. Turning to Atari’s case, Jaguar, which released just a month later than 3DO, not only it experienced similar difficulties as 3DO’s, but it also created additional difficulties for developers with its complex and dual processor architecture. Full utilization of this console required to code the games to process everything in parallel for these processors at the same time (hence the name, “parallel processing”) which meant programmers need to code heavily in assembly

language. Many developers considered this tedious, and an unnecessary hassle by the late 1993. Wayne Smithson, an old developer of DMA design (currently named as Rockstar - developer of best seller Grand Theft Auto series), explains:

"It was initially difficult to get anything decent out of it, it had more than one processor if I remember correctly ... [A]lso, it was the first time I had encountered parallel processing which took some getting used to, interleaving instructions to get the best performance out of the code was an art form in itself..." (Retrogamer, 2013).

This has been also admitted by Atari in their 10-K report of 1995:

"Atari attributes the poor performance of Jaguar to a number of factors including (i) extensive delays in development of software for the Jaguar which resulted in reduced orders due to consumer concern as to when titles for the platform would be released and how many titles would ultimately be available..." (Atari Corporation, 1995).

Yet, Atari Jaguar was an advanced console for its time. A recent release of a classic video game by a hobbyist coder for Jaguar has shown that the console could be considered quite powerful in its own time, given the right amount of investment to learn its specifics:

"It was like jumping into an alternate reality in the past where someone coded Another World on this computer," recalls Chahi [original developer of Another World]. "I was amazed by the quality of this version. Seb coded it in assembly language using the advantage of the Jaguar hardware. It is one of the best versions, clearly. The code is so well optimized that if the frame rate is not limited, it can run maybe at least five times faster than the original with all the enhanced graphics." (Crawley, 2013).

Yet, developers at that time experienced difficulties being this a novel programming environment and ended up using mainly the slowest, yet the most familiar of the console's processors, which resulted into production of titles that look similar in their quality and playing action to those released for old generation consoles SNES or Genesis (Retrogamer, 2013).

Sega Saturn had its own share of problems due to its complex architecture as well. Having 8 total processors, it offered high technical power, but mustering this proved particularly difficult. Sega's star game developer, Yuji Naka, asserts that:

"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." (Pettus, 2013; p.193).

Notwithstanding these issues, Sega hoped that investments over time to learn Saturn's architecture could give them an edge against Playstation. Tom Kalinske, back then President of Sega of America in 1995, has openly stated to Sega community on the Usenet group in the early days of Saturn that:

*"We recognize that our technical architecture has initially made Sega Saturn more difficult to develop for than other next generation formats, including the Playstation. But that is also why we know that Sega Saturn is a superior gaming platform ... We absolutely believe there will continue to be dramatic differences in software as our developers learn to unleash the power of Sega Saturn."*⁶

Nintendo 64, although being easier to develop for than Jaguar and Saturn, still required some assembly level programming for its co-processor in order to get the most performance from its graphics capabilities. Genyo Takeda, back then Nintendo's hardware development chief, summarizes the issues with Nintendo 64:

"When we made Nintendo 64, we thought it was logical that if you want to make advanced games, it becomes technically more difficult. We were wrong," he admits. "We now understand it's the cruising speed that matters, not the momentary flash of peak power." (Newsweek, 2000, p.53).

On the other hand, there was a consensus on how easy it was to develop games for Playstation. For example, it has been highlighted that:

"Saturn may have been equal - even superior - to the PlayStation at the assembly language level, but Sony had effectively changed the field in its favor. Video game programmers were happily adjusting to coding in C, and didn't want to go back, leaving fewer studios willing to wrestle with the untapped potential of Saturn's dual-processor architecture." (Retrogamer, 2012).

⁶ Accessed from Google Usenet group archives, rec.games.video.sega, <http://groups.google.com/forum/#!topic/rec.games.video.sega/VuUmORr-bw8>, accessed 24 February 2014.

This has been combined with other elements to ease game development for Playstation. Importantly, Sony used an operating platform that allowed developers to unleash the potential of the console easily: The software platform for the PlayStation was proprietary... designed exclusively for the PlayStation and optimized to make the most of the console's hardware capabilities (Evans et al., 2006: 130). Moreover, Sony was the first to provide developers with development tools to enable them to develop games for the PlayStation from the get-go. This proved an important factor in Sony's success, making it easier to write games to the PlayStation than to the competing systems (Evans et al., 2006: 130-131). We have mapped user and developer (dis)functionalities for all consoles of this generation in Figure 1.

Insert Figure 1 about here

Performance Implications

Performance of these consoles varied highly, but two big players emerged by the end of 1999, the more dominant Playstation, and the follower Nintendo 64. Early entrants were the first to fail. Atari Jaguar sold only around 250,000 units, while 3DO was able to sell around 700,000 units⁷ during their lifetime. Sega Saturn, contender of the market when it was released head-to-head with Playstation, sold only close to 2 million units. Playstation sold over 26 million units by the end of 1999 (Sony Corporation, 2014), and Nintendo 64 sold close to 15 million units (Nintendo, 2014). As for game-related performance figures, Atari Jaguar's, 3DO's, Sega Saturn's, and PlayStation's game release numbers in their first 12 months reflect the easiness or development difficulties of each console: Jaguar had 17 titles, 3DO had 71 titles, Saturn had 53 titles, and PlayStation had 133 titles. Nintendo 64 had an unusual low number of titles, 24, but these were of extremely high quality. Indeed, in terms of quality, Nintendo had 41 critically

⁷ Since 3DO was produced under license by many manufacturers, there is no exact number for 3DO.

acclaimed hits, compared to Saturn's 21; Atari had only 2, and 3DO had 7 hit games; PlayStation, instead, had over 90 critically acclaimed games released for the console. Table 5 summarizes information on developer disfunctionalities and game release performance.

Insert Tables 4 & 5 about here

Why did Playstation and to a lesser extent, Nintendo 64 succeeded? Playstation was not the most capable hardware at the time, even before the release of Nintendo 64; Saturn was thought to be more complex, but possibly a better console (Pettus, 2013). Success of Playstation has been attributed to its easiness of development and support tools for developers, as much as its balanced hardware for early 3D gaming as exemplified by this quote:

"PlayStation had a single processing chip with a 3D geometry engine in its CPU. This processor, along with the excellent development tools Sony made available, made PlayStation extremely easy to program." (Kent, 2010; p.518).

Nintendo 64 presented some developer disfunctionalities. Yet, because of its user functionality and Nintendo's in-house game development unit that produced hit titles such as Mario 64 or Pilotwings 64, the console received some noticeable sales.

Why did 3DO, Jaguar, and Saturn fail? Although 3DO was fairly easy to develop for, it offered low user functionality, and was relatively expensive for users. Bearing also this market uncertainty, developers delayed their game production for the console –this made 3DO lose momentum, and eventually fail. Jaguar failed because it had a too complex structure for developers, which caused delays and low quality games. Similar issues can be also seen in Sega's Saturn, which resulted into lack of third party support.

This generation represents one of the most turbulent periods in the console industry. Interestingly, this generation has been characterized by architecturally complex hardware except

for one console that was generally accepted to be fairly easy to work with, Sony's Playstation. It became clear, *ex post*, that those consoles that had more raw power, yet more complex architectures were generally "losers". On the other hand, Playstation was aimed at bringing 3D gaming with a good but not the most powerful hardware, but rather pursuing high developer support through its developer functionalities. In overall, we see that those consoles that were positioned in the high user functionality & high developer disfunctionality quadrant in Figure 1 were much more likely to fail compared to those consoles having high user functionality and low developer disfunctionality.

Sixth Generation Consoles (1999 – 2005)

In this generation there were 3 main changes: First, all consoles in this generation moved on optical discs (custom-DVDs, custom-CDs, DVDs), leaving cartridges obsolete. Second, 3D gaming matured, making 2D games belonging to a small niche. Third, multiplayer gaming through internet became more established. There were 4 consoles in total in this generation. Early mover was Dreamcast, through which Sega aimed to become a strong player in the market again. Dreamcast was followed by Sony's PlayStation 2, Microsoft's Xbox, and Nintendo's Gamecube.

Dreamcast was a considerable leap from the previous generation consoles in terms of hardware at the time of its release. It offered a 200 MHz processor with 16MB of main memory, as well as a NEC PowerVR II graphics processor that was cutting edge at the time of its release. Dreamcast had a custom-CD (GD-ROM) drive, as Sega decided not to use DVD since they would be prohibitively expensive in 1999 for a console. This would later prove to be a major issue for Dreamcast due to very fast adoption of DVD, and rampant piracy due to security vulnerability. Moreover, Dreamcast was the first console to have a built-in modem for Internet

and multiplayer gaming – which was not even the case for some following consoles such as Playstation 2.

Playstation 2 had 293 MHz custom processor named “Emotion Engine” and combined this with custom graphics processor named “Graphics Synthesizer”. These processors are supported by a math co-processor (FPU), but more importantly by two vector unit co-processors (VPU0 & VPU1) which were the heart of PS2’s hardware power. In overall, Playstation 2 adopted a considerably more complex hardware than Playstation. Moreover, Playstation 2 was the first console to have a DVD. This was a deliberate strategy by Sony to push its own DVD standard, and also to sell Playstation 2 at prices close to standard DVD players – subsidizing part of the cost of Playstation 2.

Xbox was developed by Microsoft, which learned about consoles through its partnership with Sega to help creating a development environment using Windows CE for Dreamcast. Microsoft perceived Sony as a big threat for its business, and decided to enter the “battle for living rooms” by using its existing experience in the PC operating system and related developers’ ecosystem. Microsoft adopted a different approach than other console producers, opting to use the PC hardware as core technology for the Xbox to court developers familiar with this technology – those that were developing already games for Windows PCs. It used a 32-bit Intel Pentium III processor at 733 MHz, combined with a NVIDIA graphics processor. Besides Playstation 2, Xbox was the other console that used DVDs as media. Xbox is the strongest hardware of the generation.

Gamecube was released a few days after Xbox in the U.S., and it was aimed to be an easy-to-develop, low-cost and graphically capable console. It was also the departure of Nintendo

from cartridges, though it did not adopt DVDs, but rather custom mini DVDs that have approximately a third of the size of DVDs (1.5GB vs. 4.7GB).

Dreamcast was designed in every way that Saturn wasn't: Sega took care this time to make the console simple and easy to develop for developers. Also, Sega had invested heavily in Dreamcast development tools (Evans et al., 2006; p.133). Yet, one important point that harmed Dreamcast was the lack of a DVD player. As DVDs became more popular – thanks to Playstation 2 – Dreamcast's lack of DVD player made mainstream consumer to ignore the console. Also, due to security vulnerability, Sega custom-CDs became effectively useless, making the console able to work with standard CD-ROMS, without any security lock on the console side (Pettus, 2013). Thus, rampant piracy harmed Sega on two fronts. First, Sega lost lots of royalties that would come from game sales – which it required to survive as Dreamcast was sold below its cost. Second, game developers, having seen piracy, effectively discounted the installed base of Dreamcast.⁸

Playstation 2, on the other hand, was designed to be the cutting-edge customized gaming machine, combining DVD player functionality for users in addition to console gaming. However, the complex architecture of Playstation 2 required considerable amount of time and resource investment to utilize the hardware in its fullest. The architecture has been explained by the co-founder of prominent developer *Naughty Dog*'s Jason Rubin:

"There are three main chips that you use on the PS2 for computing potential. There's the CPU chip, which is a pretty powerful CPU. There's VU0 [Vector Unit 0] and VU1 [Vector Unit 1]... [T]he CPU of the PlayStation 2 is 100 to 150MHZ slower than the Gamecube. So the base CPU is a slower piece of hardware. However, if you only use that, that would be the equivalent of driving a 12-cylinder car and using only six of its cylinders. It's not the way you do it correctly." (IGN, 2000). This has been explained by IGN website as: "To adequately tap PlayStation 2's Vector Units is to devote your development resources entirely to the console -- to literally allocate all of your time, money and energy

⁸ <http://kotaku.com/piracy-killed-the-dreamcast-piracy-saved-the-dreamcast-1629828642>, accessed 12 February 2015. Also, note that although it has been argued that piracy led to more sales for Dreamcast, having sold the console below its cost, and had its installed base discounted by developers, did more harm than any benefit.

into getting the most out of the system. It's not easy, and it has led many developers formerly working on PlayStation 2 software to jump and run to the competition.” (IGN, 2000).

Many others in the industry commented this openly. Gozo Kitao, the general manager of Konami, stated: "If you focus on making full use of all the specs, it will be very expensive and time-consuming to produce a game" (Evans et al., 2006; p. 132. However, there were two important user benefits of PS2, beyond its hardware power. First, it was backwards compatible. Given the enormous game library of Playstation 1, which consisted of over thousand games, Playstation 2 had an available library of games at the time of launch. Second, it was a DVD player with a lower cost than standalone DVD players.

Xbox on the other hand, was the most superior console of the generation when it arrived to the market. Also, it had the advantage of being very easy and familiar to develop for many PC game developers. "[Microsoft] built the Xbox software platform around DirectX, a collection of Windows software services that were specially designed to help PC game developers deal with the diversity of user hardware, particularly the sound and graphics cards that were so important to games." (Evans et al., 2006; p...). It has been also noted that Microsoft gave active support to developers, which "it courted... to an extent unprecedented in the video game industry" (Evans et al., 2006; p...). An additional merit of Xbox to users is its default broadband connectivity, which proved to be an important source of advantage as internet and multiplayer gaming became more established. Later versions of PS2 sold from 2004 onwards ("slimline models") added this feature by default to compete with Xbox.

Gamecube, emphasized its easiness to develop for, to the extent that Nintendo engaged some developers in the designing process of the console.

"The N2000 [development name of Gamecube] is designed from the get-go to attract third-party developers by offering more power at a cheaper price. Nintendo's design doc for the console specifies that cost is of utmost importance, followed by space" (IGN, 1999).

"To ensure that GameCube was more developer friendly, Nintendo brought developers on board to help influence the hardware's design. One of those developers was Martin Hollis, the director of Goldeneye 007 and Perfect Dark for the Nintendo 64" (Rogers, 2014).

Performance Implications

In this generation, there was one clear dominant player, Playstation 2. Xbox and Gamecube ended up sharing the rest of the market, while Dreamcast had a good start initially, but exited the market prematurely once the PS2 arrived. Financially troubled, Sega decided to leave the hardware market and focus on software only – Dreamcast had approximately 3.5 million sales in US by the time Sega left the hardware market, far behind the 5 million sales they targeted as requirement to keep stay in the market. Dreamcast was the worst performing console in this generation. PS2 in total reached approximately 47 million sales in US, starting from only 1.5 million sales in Christmas 2000, and quickly ramping up to 10 million cumulative sales by the end of Christmas 2001. Xbox had a quite successful early launch, selling 1.5 million units in 3 months (Orland, 2013). Still, it only ended up selling 15.7 million units in US and Canada combined by the end of 2005 (Microsoft Corporation, 2006). Gamecube sold 2.2 million in US and Canada combined until third quarter of 2002, and had total sales of approximately 13 million for the same region (Nintendo Corporation, 2014). These performance figures were a bit different in terms of game quantity and quality, perhaps also because of the unusual victory of PS2 and premature exit of Dreamcast. For example, due to development difficulties of PS2, it had only 44 titles by its first Christmas (2000), while Dreamcast had 51 titles in its first Christmas (1999). Dreamcast also had unusual number of hits for a failing console: it had 38

critically acclaimed hits just in 16 months of life. However in the long-term, game performance data is congruent with the sales of consoles: Playstation 2 had over thousand third-party game releases, and around 150 and 160 hits in terms of critical acclaim and sales respectively. Gamecube had only around 450 third party game releases and 57 and 27 hits in critical acclaim and sales. Xbox had 747 third party game releases and 111 and 20 hits in critical acclaim and sales. Table 6 lists information on developer disfunctionalities and game release performance.

Insert Table 6 about here

Why did Playstation 2 succeed (to the extent of dominating the market)? Playstation 2 was the most technically complex console in this generation. However, it was able to “buy time” that was required for developers to get familiar with the console. It was due to combination of unique factors that allowed Sony to manage high developer disfunctionality, while offering high user functionality to build momentum quickly. There were two major factors that allowed this risky strategy to pay off. First, Sony already had the incumbent platform, as Playstation already held half of the console market when Playstation 2 was released. Sony’s existing Playstation user and developer base gave it a great advantage. However, this alone would not guarantee success, as proved by Nintendo’s demise in the earlier generations, and as Sony itself would experience with Playstation 3 in the Seventh Generation. Besides backwards compatibility and DVD functionality, there was another equally important factor in Playstation 2’s success. After Dreamcast left the market early in 2001, Playstation 2 was the only sixth generation console in the market without any competition until the Christmas of 2001: the only major competitor, PlayStation, was also owned by Sony. In that period, PS2 was already able to reach high number of user base, that would support what *Naughty Dog*’s Jason Rubin said earlier in 2000:

"My point is, if the PlayStation 2 is going to sell as many hardware units as the PlayStation 1 sold, then I don't care if I have to pierce my nails with pins to get it to work, I'm going to do it because that's where the money is. And that's the attitude we go into every game with" (IGN, 2000).

Why did Xbox and Gamecube (to a lesser extent) have some success, and why did Dreamcast fail? Xbox and Gamecube were able to hold on the market even in the face of PS2's dominance due to their offering of high levels of user functionality, with low levels of developer disfunctionalities. This was even sharper in the case of Xbox, which offered both higher level of user functionality through its technical power, and was easiest to develop for in this generation. Indeed, it performed better than Gamecube, yet it was still dominated by PS2, as PS2 already had several millions of user base by the time of release of Xbox (Christmas of 2001). Failure of Dreamcast was due to a combination of unique factors, which permeated this generation.⁹ Dreamcast was able to gather many developers early on due to its easiness of development, and stronger hardware compared to older generation consoles early on. However, Sega no longer had financial resources to support a head-to-head competition when PS2 loomed ahead, and although it tried to reach an installed base large enough to gain momentum, it was not able to do so for two main reasons: (1) Dreamcast lacked DVD, which drove majority of the early PS2 adoption, and it eventually made the system vulnerable to piracy, cutting significantly the royalties required for Sega to compete, and game sales for developers; (2) Dreamcast entered too early the generation, which caused it to be underpowered. In sum, this generation shows idiosyncratic events whereby PlayStation 2, a console presenting lots of challenges for developers, was able to dominate the market.

⁹ <http://www.cnet.com/news/sega-dreamcast-gamings-most-magnificent-failure-video/>, accessed 16 February 2015
<http://www.polygon.com/2013/8/7/4599588/why-did-the-dreamcast-fail-segas-marketing-veteran-looks-back>, accessed 16 February 2015
<http://venturebeat.com/community/2009/09/11/qaa-former-sega-president-on-dreamcasts-failure-pranks-against-sony-his-ouster/>, accessed 16 February 2015

In sum, this generation was an exceptional period in the industry, where Dreamcast left the market early, and allowed PS2 to dominate it due to its already strong PS1 incumbency and no competition for almost a year. In such a period, PS2's developer disfunctionalities delayed game development, but, being the only platform in the sector, eventually developers commit to invest in learning how to work with it. This left Xbox and Gamecube with residual demand.

Seventh Generation Consoles (2005 – 2012)

In this recent generation of consoles, most important difference was that there were no new entrants to the market. Microsoft wanted to challenge Sony, so it moved fast with its release of Xbox 360, which was the first entrant to this generation. Actually, Xbox 360's processor, produced jointly with IBM, was based on the research alliance Sony had with IBM for Playstation 3. Though these two machines had their processor from the same research line in IBM, they adopted quite different technological architectures. Xbox 360 was able to come to market faster due to its less complicated hardware, while Playstation 3's CPU research took some more time, and it arrived almost a year later than Xbox 360. Nintendo was the last entrant with its console, Wii, following Playstation 3 with a month. Two technically advanced consoles of this generation, Xbox 360 and Playstation 3, adopted full support for "high definition" resolutions, while all three consoles embraced full internet connectivity for multiplayer gaming, social community, and digital game store.

Xbox 360 followed Xbox to a great extent, in combining both user functionality and easing the work of developers as much as possible. However, this time Microsoft used "Xenon" processors, which are a three-core variant of IBM's PowerPC line, instead of PC processors of Intel. This is a simplified version of the same processor used in PS3 later (that variant had seven-cores). Xbox 360 used DVD player, as in Xbox. In terms of hardware power, Xbox 360 was

superior than Nintendo Wii, but behind PS3. In terms of developer disfunctionalities, Xbox 360 was more complex, but also more capable than Wii, but easier to develop for than PS3.

PS3 was a high-end console that was based on custom chips focused on delivering high performance, and it also introduced the cutting edge optical media for the first time to the market: Blu-ray. However, PS3 resulted the most complex console in the market as it had an advanced processor that requires special coding techniques to get the most out of the hardware.

Nintendo followed perhaps the most interesting strategy. Wii was not designed to compete with the other two consoles in terms of technical power. It had the simplest architecture out of the three consoles, relying on a single core PowerPC based processor, based on an iteration of Gamecube design. It also didn't feature HD graphics, unlike the two other consoles of this generation. Also, Wii used custom Wii Optical Discs, and it was not able to play DVDs officially (though later homebrew hacks enabled it to do so). As put forward by famous developer of Mario games, Shigeru Miyamoto:

“The consensus was that power isn't everything for a console. Too many powerful consoles can't coexist. It's like having only ferocious dinosaurs. They might fight and hasten their own extinction” (Kenji, 2007).

Xbox 360 has been praised to be as relatively easy to develop given its powerful hardware. In the early days of Xbox 360, Japanese developers expressed that the platform has a developer-friendly development environment and array of tools that also gives them flexibility in reusing programs they have previously created (Gamespot, 2005). Especially later when PS3 was out, the comparison became clearer:

“In a generation with few third-party exclusives to separate them, 360 still places ahead of PS3. ... it's about the choices Microsoft made back in the original Xbox's lifetime. The PC-like architecture meant those early EA Sports titles ran at 60fps compared to only 30 on PS3, Xbox Live meant every dedicated player had an existing friends list, and Halo meant Microsoft had the killer next-generation

exclusive. And when developers demo games on PC now they do it with a 360 pad – another industry benchmark, and a critical one” (Edge, 2013).

Playstation 3 was the strongest console in the generation, having a similar but more advanced and complex version of Xbox 360’s processor, and having a Blu-ray reader that allows both game play and high definition movie playing functions. It also featured a more capable main memory. Although PS3 was cutting-edge in terms of hardware, it was not so well received by developers. Gabe Newell, founder of Valve which owns the biggest digital PC game retailer, Steam, said that:

"The PS3 is a total disaster on so many levels, I think It's really clear that Sony lost track of what customers and what developers wanted... Just say, this was a horrible disaster and we're sorry and we're going to stop selling this and stop trying to convince people to develop for it" (Bishop, 2007).

Sony has also admitted it. Perhaps ringing too similar to what Sega of America’s CEO Tom Kalinske maintained about Saturn back in 1995, Kaz Hirai, back then president of Sony Computer Entertainment told that:

"We don't provide the 'easy to program for' console that [developers] want, because 'easy to program for' means that anybody will be able to take advantage of pretty much what the hardware can do, so then the question is what do you do for the rest of the nine-and-a-half years? So it's a kind of - I wouldn't say a double-edged sword - but it's hard to program for, and a lot of people see the negatives of it, but if you flip that around, it means the hardware has a lot more to offer" (Purchase, 2009).

Wii was the unexpected console. Initially many developers thought an underpowered console would stand no chance in the market. However, this view has quickly changed.¹⁰ Developers have praised easiness of development for Wii , and also the fact that Wii games were far less resource consuming, as they were more similar to previous generation games.

¹⁰ "The happy story is the Wii," he [Gabe Newell] said. "I'm betting that by Christmas of next year, the Wii has a larger installed base than the 360. Other people think I'm crazy." (Bishop, 2007)

“One of the things we like about the Wii is that development costs are nowhere near what they are on the PS3 and Xbox 360. (...) The Wii wasn't a whole new programming environment. So we had a lot of tools and tech that work in that environment” (Sinclair, 2006; Anderson et al., 2014, p.153).

However, towards the end of the generation, developers had considered Wii underpowered, and many turned their attention to Xbox 360 and PS3, as the statement made by Call of Duty series' developer, Infinity Ward's community manager Robert Bowling, made clear:

“If we felt like we could deliver the cinematic experience we were going for on other platforms, then we would gladly move to that platform.. [R]ight now, we don't think the Wii can deliver the exact experience that we're doing. We like to be very equal across all platforms, and if it's not equal then we won't do it” (McFerran, 2009).

We have mapped user and developer (dis)functionalities for all consoles of this generation in Figure 2.

Insert Figure 2 about here

Performance Implications

Latest entrant Wii proved to be a runaway success, dominating the market by mid-2009. In April 2009, Nintendo Wii had 46.58% market share, as compared to 35.49% market share of Xbox 360, and 17.93% market share of Playstation 3 in the U.S..¹¹ Later on, as Nintendo Wii adoption slowed down, Xbox 360 gained market share. At the end of 2011, when Nintendo released Wii U (and focused its efforts to this new console) to upgrade the older hardware and started the new console generation, market share of the three consoles in the US were as follows: Wii 42%, Xbox 360 36%, and PS3 22%. Game release performance shows strongly how developer

¹¹ Source: www.vgchartz.com

disfunctionalities harmed PS3 compared to other consoles. Wii had the most games available for it by its first Christmas, 48, while Xbox 360 had 31, and PS3 just 17. Although Wii didn't have much critically acclaimed games (31 compared to 89 of PS3 and 111 of Xbox 360), it had highest number of hits in terms of best-sellers (103 compared to 27 of PS3 and 56 of Xbox 360). It is also clear that Xbox 360 was superior in both quality and quantity of games vis-à-vis PS3 – PS3's technical superiority didn't pay off. It could be said that Nintendo Wii was the leader in this generation, and Xbox 360 became a strong contender towards the end of 2011, while PS3 remained a follower with little hope for leadership, losing its strong position after many years. Table 7 lists information on developer disfunctionalities and game release performance.

Insert Table 7 about here

Why did Wii, and Xbox 360 succeed? Though Nintendo Wii's success has been attributed to its innovative motion-sensitive control and accompanying games more favored by a growing casual gaming segment in the industry, "... the remote cannot fully explain Nintendo Wii's dominance, because for a long time most of the games developed for the Wii used the traditional joystick technology" (Anderson, Parker, and Tan, 2014: 153). Wii's success can be attributed to the relative ease of game development by developers on two fronts: First, it was essentially a modified previous generation console that was perhaps only double the power of previous generation consoles, which allowed developers to directly apply their learning accrued in the previous generation (Anderson et al., 2014). Second, due to having less hardware demanding games, it allowed developers to have much cheaper game development costs in otherwise an industry having immensely rising game development costs.

“So those costs--and again, I hate these broad generalizations--but they could be as little as a third of the high-end next-gen titles... Maybe the range is a quarter to a half” (Sinclair, 2006).

Xbox 360 was successful because it offered relatively low developer disfunctionalities while offering user benefits, especially compared to its close competitor PS3. For example, Activision’s CEO, Bobby Kotick said:

“It’s expensive to develop for the console [PS3], and the Wii and the Xbox are just selling better. Games generate a better return on invested capital on the Xbox than on the PlayStation” (Walton, 2009).

Actually development for PS3 initially proved so difficult that even with its superior hardware, PS3 games that were also developed for Xbox 360 mostly underperformed compared to their Xbox 360 counterparts (especially at the initial release of PS3 where problems are most significant).

“You’d think that the PS3 versions would be exactly the same or slightly superior to the Xbox 360 versions, since many of these games appeared on the 360 months ago, but it seems like developers didn’t use the extra time to polish up the graphics for the PS3. We found that the Xbox 360 actually had better graphics in the majority of the games we compared” (Shah and Yu, 2006).

Why did PS3 fail? PS3 failed, because developer disfunctionalities at the early stage of its release caused it to lose important momentum. In developing PS4 (not included in this study), Mark Cerny, the head architect of the system attributed the problems Sony experienced prior to the release of PS3 to the complexity of the processors’ and programming environment, which made it difficult for developers to perform "the most basic tasks" (Scammell, 2013). Development tools support for PS3 was also limited.

“The development environment was in a very primitive state. The first party teams were having a difficult time of it but the third party teams, without the luxury of being able to focus just on PS3, and without the benefit of our (Sony’s) head start, were having an even more troubling time. The teams that I’d worked with, first party, needed basically a whole year to create usable graphic engines. The sky high expectations for game titles could only be met with clever use of the SPUs, but both the unique nature of

the Cell and the primitive state of the development environment meant that game creation on PlayStation 3 was more time consuming than any previous product” (Hurley, 2013).

In sum, results from this generation were quite similar to the 5th generation. Back then, Sony won the market by combining low levels of developer disfunctionalities with high levels of user functionalities. Yet, this time Playstation 3 was the “new Saturn”, and Sony eventually lost its lead in the market after two generations due to developers’ learning problems and relatively less willingness to commit resources.

Summary of study’s findings

The analysis of the different platform technology generations in the video game industry presented above reveals interplay between technological (dis)functionalities to users and complementors that critically affects the formation and growth of an ecosystem around the new-generation platform technology. As described above, the fourth generation of gaming consoles showed a similar level of complementor-related (dis)functionalities; the platforms that ended up gaining dominance were indeed those with superior technology, i.e., those offering greater technological functionalities to users. Consistent with traditional conceptualization of technological superiority, these platforms were offering users greater value and were been adopted. Since developers faced similar technological environments (and thus innovation challenges), they just selected the platforms that were gaining (or expected to gain) increasing user base. However, the following generations showed high variance in terms of technological (dis)functionalities, both on the user and complementor side. Platforms that offered limited functionalities on both sides, such as Philips CD-i, were clearly of limited value to both users and complementors. These platforms hardly gained any ground. However, also platforms offering

great levels of functionalities to users like Sega Saturn and Playstation 3 eventually failed to gain market dominance vis-à-vis competing platforms with similar or lower levels of user-related (dis)functionalities that provided greater benefits or lower dis-functionalities to complementors – e.g., Nintendo Wii, PlayStation.

Essentially, this suggests that the window of opportunity for the leapfrogging platform could be shorter than expected. New generation consoles, even if of superior technology, offer not much value to users unless there are games developed for it to play. It has been a common problem that games developed for new machines simply didn't take advantages of the new superior technical features, as developers didn't invest much to harness the technological power of the new console. This issue is exacerbated by the installed base of old generation consoles, with developers preferring to release games for this installed base even after console user base is not increasing (Clements and Ohashi, 2005).

In light of these findings, technological superiority as traditionally conceptualized, instead of being an enabler, represented, in fact, an important potential obstacle to ecosystem formation and growth. This highlights a paradoxical property of technological superiority (as traditionally conceptualized) when a radical innovation is undertaken to leapfrog the incumbent. While it provides more benefits to users, it also creates a gap in the knowledge required for the complementors to support the platform. The greater the technological superiority, the greater the gap in this knowledge, and the required learning investments. This paradox essentially limits the value of the technology in system contexts, contrary to what has been argued in the literature for more traditional contexts.

This paradox is essentially determined by three main problems associated with technological superiority: (i) time compression diseconomies, (ii) resource allocation problems,

and (iii) lock-in problems. Time compression diseconomies will be present due to impossibility of compressing learning in a short time period even with an increased allocation of resources (Dierickx and Cool, 1989). Resource allocation problems will be present due to limited current value of the new platform for the complementor, inducing the complementor to allocate second-tier resources to the platform-specific complements' development. Lock-in problems will be present due to market uncertainty – complementors will be unsure if it pays off to invest and learn for this new platform as much of this investment would be lost in case the platform does not turn to be popular in the market.

DISCUSSION

We have started this article by pointing out the recognized determinant role played by new, superior technology in shaping competition toward a dominant technological platform, as acknowledged by several studies in the literature. It has been also recognized that this effect is more pronounced in networked markets because of the increasing returns to scale ensuing from network effects dynamics. Notwithstanding these benefits, superior technology may not always become the dominant platform. Scholars have attributed this to excess inertia (Katz & Shapiro 1994), and lack of a sufficient user base or complements (Schilling 2002; Suarez 2004). Also, even when users and complementors embrace early on the new, superior technology, this may fail to become the industry standard. However, why users, and complementors, would choose an “inferior” technology remains a puzzle, and, surprisingly, an under-theorized, and overlooked phenomenon. Only a handful of studies have explored the issue, pointing to industry specificities or peculiar, idiosyncratic events (e.g., Arthur 1989; Rosenbloom and Cusumano, 1987).

We thus raised such question as the core of our inquiry. Specifically: Why do new, technologically superior platforms fail, despite gaining early momentum? Our objective was to

gain insights on the underlying factors magnifying (constraining) the benefits (costs) of introducing a new, superior technology. Our analysis of US Video Game Industry revealed that, technological superiority, which is used by challengers in platform competition, has a paradoxical property. As much as it drives early user adoption by offering direct benefits to users (greater technological functionalities), it entails shadow costs for complementors in terms of learning, opportunity costs, and investments. This limited the emergence and growth of a thriving ecosystem around the new technology.

However, this does not mean that technological superiority should not be pursued. To the contrary, examples of successful platforms in our research context show that, when properly managing the inherent paradox that technology superiority entails, it creates opportunities for value creation. We have gone further in our analysis to gain insights on how do firms manage this paradox of technological superiority, an issue we briefly turn now.

Managing the Technological Superiority Paradox: Vertical Integration and Co-opetition

We have so far mentioned difficulties faced by complementors when deciding to join and support a new platform technology. How were console owners then able to attract developers early on in the new technology life cycle? Almost every platform did by producing itself parts of the needed games. Most platform owners in the videogame industry had an extensive in-house video game production. These complementary products released by the console owner itself were used to sustain the console through its life, but its primary purpose was to build the best games possible to showcase the power of the console early on in the new technology life cycle, and gain an initial lead in user base. Joe Miller, who was the Senior Vice President of Product Development of Sega in the 90s answered in an interview that: “First party software is meant to

accelerate the growth of an installed base so third parties can jump on the bandwagon and quickly develop great titles for the hardware and generate higher margin revenue” (Horowitz, 2013). This initial user base will then serve to attract developers, and induce them to invest in learning to develop games for new consoles. Recent research also confirms the value of vertical integration in the form of in house development for attracting users to platforms (Lee, 2013).

Vertical integration in the complementary products segment, however, also entails its own issues. Objectively, it would be an ideal situation where platform owner supplies complementary products to the platform to gain a head-start, and also shares the in-house knowledge of complementary product development for the new console with external complementors. However, incentive and co-opetition problems brought by vertical integration in complementary products is a critical issue for firms sponsoring platform technologies (Gawer and Henderson, 2007). In the video game industry, platform owners were also additionally tempted to produce strong complements as the business model is based on selling the console below or at marginal cost and getting royalties for each video game sold from third party developers.¹² In the case of in house development, the firm gets 100 percent of the profit, which tempts platform owners even more to compete with their complementors. It has been widely noted that, Nintendo and Sega were reluctant to provide their knowledge through the history of the industry, until Sony beat them with Playstation through the strategy of providing its development knowledge extensively to complementors (Evans, Hagiwara and Schmalensee, 2006). Along these lines, it has been argued that, “the software development kits (SDKs) for Sega CD were late in arriving from Japan, which was one of the major contributing factors towards the lack of a good software base” (Pettus, 2013, p.282). Providing a level field for both in house and

¹² It is important to note that challenging this business model is not useful either. For example, 3DO failed despite its strong external developer support with 3DO’s below industry royalties as console was too expensive.

external complementors is important in attracting complements (Gawer and Henderson, 2007), and failure to do so may do more harm than benefits.

Supporting Complementors: Thriving from Vertical Integration

Governance of multiplicity of factors is critical in reaching platform success (Boudreau & Hagiu, 2010), and video game industry is no different in this regard. Vertical integration into complements could be a source of advantage in fact in managing complementors since the platform can use insights gained from developing games for the new technology, and transfer this knowledge to external complementors. This would reduce their learning, hence switching costs, and increase their ability to produce better quality games. Our reading of the history shows that those consoles that were able to successfully make the technological leapfrogging were the ones that have combined vertical integration to build the user base, while providing a level play field between complementors, echoing findings of Gawer and Henderson (2007). While technological superiority of Sony's Playstation was highly debated, its success has been attributed to software development tools (Evans et al., 2006). These development tools include graphical libraries, sound libraries and API's that ease programming for the platform greatly. By reducing the challenges faced by complementors, Sony was able to create much more value to users, and sustain support from external developers. Interestingly, among major competitors, Sony had the least extent of in house video game development – which intrinsically supports our idea in the above discussion.

Finding the Right Balance in Technological Superiority

On the long run, technological superiority may become essential. Although we argued that challenging with very advanced technology will have liabilities, so do having not enough

superiority in the long run. In console industry, developers will generally favor consoles that have superior technical capabilities (graphical and processing power, memory and the like). For example, on the 128-bit console generation, Sega's Dreamcast was easier to develop games for compared to Sony's Playstation 2 (PS2). PS2's game development difficulties were attributed to "Emotion Engine" chip of the console, which was technological state of art in the generation until Xbox's late arrival. Early on, Sega hoped that it could survive as game development for PS2 delayed, while there was constant game release for Dreamcast. In fact, Dreamcast that released more than a year earlier, served as an "interim" system where developers learned how to code in the next generation technology, and developers hopped on to the later entrant PS2 when it became more promising (Pettus, 2013). This shows that technological superiority is a liability at the beginning of a new generation, while it becomes an advantage through the course of technological life cycle if it is able to sustain its momentum through network effects. Indeed, those many consoles that lead the user base by pioneering a new generation such as Dreamcast become stepping stones for later entrants like PS2, while those consoles that were superior right away in pioneering the new generation were simply not able to attract enough developers due to additional difficulties brought by technological superiority (such as Atari Jaguar).

CONCLUSION

In this paper, we have adopted an ecosystem view on superior platform technology, considering both users and complementors, and have highlighted an important paradox of technological superiority: it creates functional value for users that could lead to initial momentum, but it also puts more burden on complementors as the technological gap increases compared to existing platforms. This makes the new platform less easy and more expensive to switch to and support by complementors, therefore reducing the amount and quality of

complements provided to a platform. In particular, impediments in the form of time compression diseconomies, resource allocation problems and switching costs for complementors negatively affect the ability of a technological superior new platform to obtain complements at the rate and quality needed to support the initial market momentum. This in turn would generate unfavorable expectations, and eventually lead to the failure of the platform.

Platforms would need to take an active stance to manage the ecosystem and alleviate these problems. In our discussion, we have provided several solutions to manage these issues, such as vertical integration, supporting complementors, and having the right balance in technological superiority and complementor ease of support. Yet, as we have argued, platforms should also be aware of the second order tradeoffs that these solutions entail – such as co-opetition with other complementors due to vertical integration.

By focusing on platform failures instead of successes, and adopting an ecosystem perspective, we have been able to highlight a set of dilemmas firms sponsoring new platform technologies face when trying to create more value for users, and identified the possible ways to manage them. We hope other studies may build upon these insights to enrich the analysis and understanding of this superior technology paradox.

REFERENCES

- Adner, R., R. Kapoor. 2010. Value creation in innovation ecosystems: How the structure of technological interdependence affects firm performance in new technology generations. *Strategic Management Journal*. 31(3) 306–333.
- Afuah, A. (2013), Are network effects really all about size? The role of structure and conduct. *Strat. Mgmt. J.*, 34: 257–273.
- Alexander, L. 2009. Hirai Claims PS3 'Official' Leadership, Criticizes Rivals. http://www.gamasutra.com/php-bin/news_index.php?story=21933, accessed 20 February 2015.
- Anderson, E.G., G. Parker, B. Tan (2014). "Platform Performance Investment in the Presence of Network Externalities." *Information Systems Research*. Vol. 25, No. 1.
- Argyres, N., Bigelow, L. and Nickerson, J. A. (2015), Dominant designs, innovation shocks, and the follower's dilemma. *Strat. Mgmt. J.*, 36: 216–234.
- Armstrong M. 2006. Competition in two-sided markets. *RAND Journal of Economics* 37(3): 668–691.
- Atari Corporation, 1995 10-K Report, <http://www.sec.gov/Archives/edgar/data/802019/0000891618-96-000213.txt>, accessed February 24, 2014.
- Baldwin, Carliss Y., Alan MacCormack, and John Rusnak. 2014. "Hidden Structure: Using Network Methods to Map System Architecture." *Research Policy*, 43, 1381–1397.
- Bishop, S. 2007. Gabe Newell: PS3 "a waste of everybody's time". <https://web.archive.org/web/20141103232220/http://www.computerandvideogames.com/155309/valve-ps3-a-total-disaster-on-so-many-levels/>, accessed 20 February 2015.
- Boudreau KJ, Hagiu A. 2010. Platform rules: multi-sided platforms as regulators. In *Platforms, Markets and Innovation*, Gawer A (ed). Edward Elgar: Northampton, MA; 163–191.
- Boudreau KJ. 2012. Let a thousand flowers bloom? An early look at large numbers of software app developers and patterns of innovation. *Organization Science* 23(5): 1409–1427.
- Cennamo C. 2013. Network Effects Theory and Competitive Dynamics in Platform Markets: The Missing Link. Working paper, Bocconi University
- Cennamo, C. and Santalo, J. 2013. Platform competition: Strategic trade-offs in platform markets. *Strategic Management Journal*, 34: 1331–1350.
- Choi J.P., 1994. Network Externality, Compatibility Choice, and Planned Obsolescence. *The Journal of Industrial Economics*, Vol. 42, No. 2, pp. 167–182.
- Christensen, C. M., Suarez, F. F., & Utterback, J. M. 1998. Strategies for survival in fast changing industries. *Management Science*, 44: 207–220.
- Clements MT, Ohashi H. 2005. Indirect network effects and the product cycle: video games in the U.S., 1994–2002. *Journal of Industrial Economics* 53(4): 515–542.
- David PA. 1985. Clio and the economics of QWERTY. *American Economic Review* 75(2): 332–337.
- David, P. A. & Greenstein, S. (1990). The Economics of Compatibility Standards: An Introduction to Recent Research. *The Economics of Innovations and New Technology*, 1(1/2), 3–41.
- De Vaan, M. 2014. Interfirm networks in periods of technological turbulence and stability. *Research Policy*, 43, 1666–1680.
- Dhanaraj, C., & Parkhe, A. 2006. Orchestrating innovation networks. *Academy of Management Review*, 31(3): 659–669.
- Dibiaggio, L. 2007. "Design complexity, vertical disintegration and knowledge organization in the semiconductor industry," *Industrial and Corporate Change*, 16, 239–267.
- Dierickx, I. and K. Cool (1989), 'Asset stock accumulation and sustainability of competitive advantage,' *Management Science*, 35, 1504–1511.
- Dosi, G. 1982 "Technological paradigms and technological trajectories: A suggested interpretation of the determinants and directions of technical change." *Research Policy*, 11: 147–162.
- Edge. 2013. The ten best consoles: our countdown of the greatest gameboxes of the last 20 years. <https://web.archive.org/web/20141128181657/http://www.edge-online.com/features/the-ten-best-consoles-the-greatest-gameboxes-from-the-past-20-years/>, accessed 20 February 2015.
- Eisenhardt KM. 1989. Building theories from case study research. *Academy of Management Review* 14(4): 532–550.
- Eisenmann TR, Parker G, Alstyne MWV. 2006. Strategies for two-sided markets. *Harvard Business Review* 84(10): 92–101.

- Evans, D. S., A. Hagiu and R. Schmalensee, 2006, *Invisible Engines: How Software Platforms Drive Innovation and Transform Industries*, Cambridge, MA: The MIT Press.
- Evans DS. 2003. Some empirical aspects of multi-sided platform industries. *Review of Network Economics* 2: 191–209.
- Forster, W., 2005. *The Encyclopedia of Game Machines. Consoles, Handhelds, and Home Computers 1972–2005*. Game Plan, New York, NY.
- Gallagher, S., S. H. Park. 2002. Innovation and competition in standard-based industries: A historical analysis of the U.S. home video game market. *IEEE Trans. Engrg. Management* 49(1) 67–81.
- Gamespot. 2005. Japanese developers weigh in on Xbox 360. <http://www.gamespot.com/articles/japanese-developers-weigh-in-on-xbox-360/1100-6132218/>, accessed 20 February 2015.
- Garud, R. Jain, S. and Kumaraswamy, A. 2002 'Institutional entrepreneurship in the sponsorship of common technological standards: The case of Sun Microsystems and Java'. *Academy of Management Journal*, 45: 196-214.
- Gawer A, 2014. Bridging differing perspectives on technological platforms: Toward an integrative framework. *Research Policy*, Volume 43, Issue 7, Pages 1239-1249.
- Gawer A, Cusumano MA. 2008. How companies become platform leaders. *MIT Sloan Management Review* 49(2): 28–35
- Gawer A. & Henderson R.M. 2007. Platform Owner Entry and Innovation in Complementary Markets: Evidence from Intel. *Journal of Economics & Management Strategy* 16: 1-34.
- Gibson, E. 2007. GDC: What's Next for PS3? <http://www.gamesindustry.biz/articles/gdc-whats-next-for-ps3>, accessed 25 January 2014.
- Glaser BG, Strauss AL. 1967. *The Discovery of Grounded Theory*. Aldine: Chicago, IL
- Hagiu A. 2006. Pricing and commitment by two-sided platforms. *RAND Journal of Economics* 37: 720–737.
- Harris, B.J. 2014. *Console Wars: Sega, Nintendo, and the Battle that Defined a Generation*. HarperCollins Publishers. New York.
- Hawkin, K., 2013. *Retroinspective: Atari Jaguar 20 Years On*, *Retrogamer Magazine*, 119.
- Horowitz K., 2013. Interview: Joe Miller. <http://www.sega-16.com/2013/02/interview-joe-miller>, accessed February 20, 2014.
- Iansiti, M., and Levien, R. 2004. *The Keystone Advantage: What the New Dynamics of Business Ecosystems Mean for Strategy, Innovation, and Sustainability*, Boston: Harvard Business School Press.
- IEEE Standards Board 1990. *IEEE Standard Glossary of Software Engineering Technology*. Technical report. The Institute of Electrical and Electronics Engineers.
- IGN, 1999. It's Alive! <http://www.ign.com/articles/1999/03/13/its-alive-3>, accessed 10 February 2015.
- IGN, 2000. Gamecube vs. Playstation 2. <http://www.ign.com/articles/2000/11/04/gamecube-versus-playstation-2>, accessed 16 February 2015.
- Katz ML, Shapiro C. 1985. Network externalities, competition, and compatibility. *American Economic Review* 75: 424–440
- Katz, M., & Shapiro, C. 1986. Technology adoption in the presence of network externalities. *Journal of Political Economy*, 94: 822-841.
- Katz, Michael and Carl Shapiro, (1994), "Systems Competition and Network Effects," *Journal of Economic Perspectives*, vol. 8, no. 2, pp. 93-115.
- Kenji, H. 2006. "The Big Ideas Behind Nintendo's Wii" https://web.archive.org/web/20070102233737/http://www.businessweek.com/technology/content/nov2006/tc20061116_750580.htm, accessed 10 January 2015.
- Kent, S. L. 2001. *The Ultimate History of Video Games*, New York, Three Rivers Press.
- Lagerstrom, Robert, Carliss Y. Baldwin, Alan MacCormack, and David Dreyfus. 2014. "Visualizing and Measuring Software Portfolio Architectures: A Flexibility Analysis." *Harvard Business School Working Paper*, No. 14-083.
- Langley, A. 1999. Strategies for theorizing from process data. *Academy of Management Review*, 24, 691-710.
- Lee E, Lee J, Lee J. 2006. Reconsideration of the winner-take-all hypothesis: complex networks and local bias. *Management Science* 52: 1838–1848
- Lee, Robin S. 2013. "Vertical Integration and Exclusivity in Platform and Two-Sided Markets." *American Economic Review*, 103(7): 2960-3000.

- Lieberman, M. B., and D. B. Montgomery 1988 "First-mover advantages." *Strategic Management Journal*, 9: 41–58.
- Lien T, 2013. Why Do We Need New Consoles? <http://www.polygon.com/2013/11/7/5070206/why-do-we-need-new-consoles>, accessed February 25, 2014.
- Matthews, W. 2013. Ahead of its Time: A 3DO Retrospective. *Retro Gamer*, 122: 18–29.
- McFerran, 2010. D., *Retrospection: 32X, Videogames Hardware Handbook Vol. 2*, Imagine Publishing, Bournemouth, UK.
- Newsweek, 2000. It's Hip To Be Square. 136 (10): 53.
- Utterback, J. M., W. J. Abernathy. 1975. A dynamic model of product and process innovation. *Omega* 3(6) 639–656.
- Pettus, S. 2013. *Service Games: The Rise and Fall of SEGA: Enhanced Edition*. CreateSpace Independent Publishing Platform.
- Rochet JC, Tirole J. 2006. Two-sided markets: a progress report. *RAND Journal of Economics* 37: 645–667.
- Rogers, E. 2014. A Dolphin's Tale: The Story of GameCube. <http://www.dromble.com/2014/01/07/dolphin-tale-story-of-gamecube/>, accessed 20 December 2015.
- Rosenbloom, R. and M. Cusumano. "Technological pioneering and competitive advantage: The birth of the VCR industry", *California Management Review*, 29(4), 1987, pp. 51-76.
- Scammel, D. 2013. PS4 architect explains PS3's 'weak' launch and 'primitive development environment'. http://www.videogamer.com/ps3/i8/news/ps4_architect_explains_ps3s_weak_launch_and_primitive_development_environment_2.html, accessed 20 February 2015.
- Schilling, MA. A. 1998 "Technological lockout: An integrative model of the economic and strategic factors driving technological success and failure." *Academy of Management Review*, 23: 267–284.
- Schilling MA. 2002. Technology success and failure in winner-take-all markets: the impact of learning orientation, timing, and network externalities. *Academy of Management Journal* 45: 387–398.
- Schilling MA. 2003. Technological leapfrogging: lessons from the U.S. video game console industry. *California Management Review* 45(3): 6–32.
- Shapiro C, Varian HR. 1999. The art of standards wars. *California Management Review* 41(2): 8–32.
- Shankar, Venkatesh, Bayus, Barry, 2003. Network effects and competition: an empirical analysis of the home video game industry. *Strategic Management Journal* 24, 375–384.
- Sheremata, W. A. 2004. Competing through innovation in network markets: Strategies for challengers. *Academy of Management Review*, 29: 359–377.
- Simon, H. A. 1962. The architecture of complexity. *American Philosophical Society* 106, 6, 467-482.
- Sinclair, B. 2006. Wii dev costs fraction of PS3's, 360's. *Gamespot*, <http://www.gamespot.com/articles/e3-06-thq-wii-dev-costs-fraction-of-ps3s-360s/1100-6149154/>, accessed 20 February 2015.
- Suarez F. 2004. Battles for technological dominance: an integrative framework. *Research Policy* 33: 271–286.
- Suarez F. and Kirtley J., "Dethroning an Existing Platform," *MIT Sloan Management Review* 53, no. 4 (summer 2012): 35-40.
- Tushman, M. L., and P. Anderson 1986 "Technological discontinuities and organizational environments." *Administrative Science Quarterly*, 31: 439–465.
- Venkatesh, V. 2000. "Determinants of Perceived Ease of Use: Integrating Perceived Behavioral Control, Computer Anxiety and Enjoyment into the Technology Acceptance Model," *Information Systems Research*, 11, 342-365.
- Venkatesh, V. and Davis, F.D. "A Theoretical Extension of the Technology Acceptance Model: Four Longitudinal Field Studies," *Management Science*, 46, 2000, 186-204.
- Venkatesh, V., and Bala, H. "Technology Acceptance Model 3 and a Research Agenda on Interventions," *Decision Sciences*, 39, 2008, 273-315.
- Walton, M. 2009. Activision threatens to drop PS3, PSP. <http://www.gamespot.com/articles/activision-threatens-to-drop-ps3-psp/1100-6212246/>, accessed 20 February 2015.
- Xin, X., Venkatesh, V., Tam, K.Y., and Hong, S. 2010. "Consumer Migration across Generations of Technology Service Platforms: Roles of Generation, Technology Hierarchy, and Complementarities," *Management Science*, 56, 1304-1323.
- Yin, R. K. 1994. *Case Study Research: Design and Methods*, 2nd ed. Sage Publications, Thousand Oaks, CA.

Table 1. Summary of definitions on User (Dis)functionality and Developer (Dis)functionality

Reference	User (Dis)functionality Factors	Explanation
Venkatesh, 2000	Technology Acceptance Model - Perceived Usefulness	The extent to which a person believes that using a technology will enhance her/his productivity
Xu et al., 2010	Technology Perceptions - Usefulness	"The degree to which an individual believes that using the technology (i.e., ICT platform) will help him or her to attain gains in personal productivity" (Venkatesh and Davis, 2000; Xu et al., 2010).
Xu et al., 2010	Technology Perceptions - Enjoyment	Enjoyment is defined as the extent to which the act of using the new ICT platform is perceived to be pleasurable in its own right, apart from any expected performance consequences (Venkatesh, 2000).
Xu et al., 2010	Technology Perceptions - Superiority of services	The extent to which the new platform is expected to support and provide services that are better than those available in an existing platform generation.
Anderson, Parker, and Tan (2014)	Platform Performance	A vertically differentiated dimension of quality (better graphics and processing capabilities) - Increases end users' utility
Reference	Developer (Dis)functionality Factors	Explanation
Simon (1962); Alexander (1964)	Complexity	Linkages existing between different elements in the system in a network representation
IEEE Standards Board (1990)	(Software) Complexity	The degree to which a system or component has a design or implementation that is difficult to understand and verify
Lagerstrom, Baldwin, MacCormack, and Dreyfus (2014)	(Software) Complexity	Lines of Codes
Lagerstrom, Baldwin, MacCormack, and Dreyfus (2014)	(Software) Complexity	Relationships between components in a system
Dibiaggio (2007)	Complexity	Number of interconnected components and subsystems
Anderson, Parker, and Tan (2014)	Development cost per unit performance	Game creation for higher-capable hardware is more complex - Increases complementors' costs

Table 2. Sample Consoles in US Video Game Industry (1985-2012)

Console	U.S. Launch Date	Platform Parent	CPU bits	MHz	RAM	Launch Price
Generation 3						
Nintendo Entertainment System (NES)	Oct. 1985	Nintendo	8-bit	1.8	0.002	\$199
Master System	Oct. 1986	Sega	8-bit	3.54	0.008	\$199
Generation 4						
Genesis	Aug. 1989	Sega	16-bit	7.6	0.064	\$190
Turbografx-16	Aug. 1989	NEC/Hudson Soft	8/16-bit	7.16	0.008	\$199
Super Nintendo Entertainment System (SNES)	Aug. 1991	Nintendo	16-bit	3.58	0.128	\$199
Sega CD (Add-on for Genesis)	Oct. 1992	Sega	16-bit	12.5	0.75	\$299
32X (Add-on for Genesis)	Nov. 1994	Sega	32-bit	23 X 2	0.512	\$159
Generation 5						
3DO	Oct. 1993	3DO	32-bit	12.5	2	\$699
Jaguar	Nov. 1993	Atari	32/64-bit	26.6 X 2	2	\$249
Saturn	May. 1995	Sega	32-bit	28 X 2	2	\$399
Playstation	Sept. 1995	Sony	32-bit	33.87	2	\$299
Nintendo 64	Sept. 1996	Nintendo	64-bit	93.75	4 (Unified)	\$199
Generation 6						
Dreamcast	Sept. 1999	Sega	128-bit	200	16	\$199
Playstation 2	Oct. 2000	Sony	128-bit	294.9	32	\$299
Xbox	Nov. 2001	Microsoft	Pentium II	733	64	\$299
Gamecube	Nov. 2001	Nintendo	128-bit	485	40	\$199
Generation 7						
Xbox 360	Nov. 2005	Microsoft	Power PC	3200 x 3	512 (Unified)	\$299
Playstation 3	Nov. 2006	Sony	Power PC	3200 x 7	256	\$499
Wii	Nov. 2006	Nintendo	Power PC	729	88	\$249

Table 3. Summary of Complementor Disfunctionalities, and Game Release Performance for Fourth Generation Consoles

	Time Compression Diseconomies	Resource Allocation Problems	Lock-in Problems	Important Market Issues	Game Release Performance
Genesis	No	No	No	Demand shock occurred through the release of Sonic the Hedgehog	Games available by the first Christmas (89'): 17
					Games available in the first 12 months: 70
					Exclusive games for the console: 355
					Ported games from old generation devices for the console: 38
	Example Quotes				First-party (second-party) games: 168 (103)
	Early devices up to 32-bit machines had similar development requirements with assembler heavy programming.		Third-party games: 513		
			Average quality of games (First & Second-Party): 74.67/100		
		Average quality of games (Third-Party): 71.49/100			
		# of hits: 59 (Critical Acclaim= 85+/100)			
Turbografx-16	No	No	No		# of hits: 17 (1 Million+ sales)
					Games available by the first Christmas (89'): 17
					Games available in the first 12 months: 47
					Exclusive games for the console: 56
					Ported games from old generation devices for the console: 2
	Early devices up to 32-bit machines had similar development requirements with assembler heavy programming.		First-party (second-party) games: 76 (58)		
			Third-party games: 32		
		Average quality of games (First & Second-Party): 81.11/100			
		Average quality of games (Third-Party): 78.5/100			
SNES	No	No	No		# of hits: 7 (Critical Acclaim= 85+/100)
					# of hits: N/A (1 Million+ sales)
					Games available by the first Christmas (91'): 35
					Games available in the first 12 months: 113
					Exclusive games for the console: 406
					Ported games from old generation devices for the console: 54
	Early devices up to 32-bit machines had similar development requirements with assembler heavy programming.		First-party (second-party) games: 49 (33)		
		Third-party games: 648			
		Average quality of games (First & Second-Party): 82.81/100			
		Average quality of games (Third-Party): 71.95/100			
32X	YES	YES	No	Add-on: requires the base console to function, therefore it splits the installed base of Genesis for developers. 32X was considered between generations.	# of hits: 57 (Critical Acclaim= 85+/100)
					# of hits: 46 (1 Million+ sales)
					Games available by the first Christmas (94'): 13
					Games available in the first 12 months: 35
					Exclusive games for the console: 16
					Ported games from old generation devices for the console: 19
	Example Quotes				First-party (second-party) games: 17 (8)
"I spent weeks working with id Software's John Carmack, who literally camped out at the Sega of America building in Redwood City trying to get Doom ported. That guy worked ... and he still had to cut a third of the levels to get it done in time."		Third-party games: 19			
		Average quality of games (First & Second-Party): 65.47/100			
		Average quality of games (Third-Party): 65.52/100			
		# of hits: 3 (Critical Acclaim= 85+/100)			
		# of hits: N/A (1 Million+ sales)			

Table 4. Total Game Releases in each year in US for 1991-2001

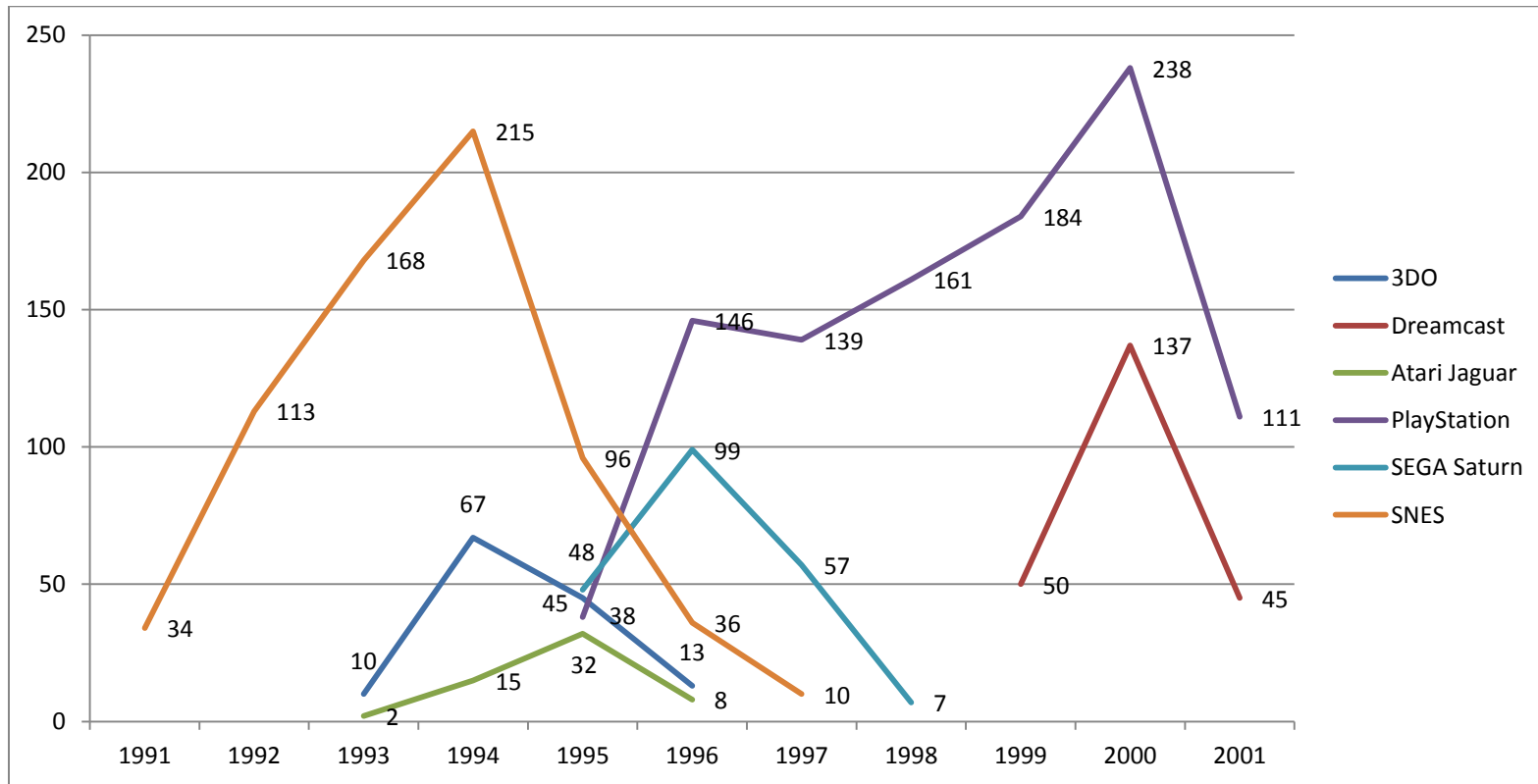


Table 5. Summary of Complementor Disfunctionalities, and Game Release Performance for Fifth Generation Consoles

	Time Compression Diseconomies	Resource Allocation Problems	Lock-in Problems	Important Market Issues	Game Release Performance
3DO	YES	No	YES	Due to business model of 3DO (subsidizing developers instead of users, and licensing the production) its console price was high, and price cuts come slower than expected.	Games available by the first Christmas (93'): 10
					Games available in the first 12 months: 71
					Exclusive games for the console: 73
					Ported games from old generation devices for the console: 17
					First-party (second-party) games: 11 (2)
Jaguar	YES	YES	YES	David Wightman: "The Jaguar had a European soul with a Texan hat... [A]merican coders by route of the Apple II and PC had learned to go through a bios and an operation system to code, they had a tough time dropping down to binary after the luxury of APIs and libraries which they had become accustomed to. That's a core reason why very few titles came from American corporations before launch, they struggled to find people who understood how to program hardware without a soft pillow to sit on." (Retrogamer, 2013, p.14)	Third-party games: 126
					Average quality of games (First & Second-Party): 77.85/100
					Average quality of games (Third-Party): 69.81/100
					# of hits: 6 (Critical Acclaim= 85+/100)
					# of hits: N/A (1 Million+ sales)
Saturn	YES	YES	YES	Early launch caused resentment among some retailers, up to the point that Kay Bee Toys refused to carry it. Developers were also harmed as it even increased learning related problems due to pressure to release games even earlier.	Games available by the first Christmas (93'): 2
					Games available in the first 12 months: 17
					Exclusive games for the console: 47
					Ported games from old generation devices for the console: 14
					First-party (second-party) games: 41 (33)
Playstation	No	No	No	"Sega's machine...required the best coders to really get the most out of it. Meanwhile, third-party studios were getting stuck in to Playstation game production, and a string of classic titles began to emerge." (Retrogamer, 2012)	Third-party games: 30
					Average quality of games (First & Second-Party): 59.91/100
					Average quality of games (Third-Party): 68.72/100
					# of hits: 2 (Critical Acclaim= 85+/100)
					# of hits: N/A (1 Million+ sales)
Nintendo 64	YES	No	No	Use of cartridges instead of CD-ROMs hampered N64 with many developers preferring large data storage provided by CD-ROM.	Games available by the first Christmas (93'): 48
					Games available in the first 12 months: 53
					Exclusive games for the console: 86
					Ported games from old generation devices for the console: 28
					First-party (second-party) games: 64 (28)
Nintendo 64	YES	No	No	Required to rewrite part of the graphics processor microcode to get most out of the machine (Retrogamer, 2012)	Third-party games: 149
					Average quality of games (First & Second-Party): 76.82/100
					Average quality of games (Third-Party): 71.69/100
					# of hits: 21 (Critical Acclaim= 85+/100)
					# of hits: 2 (1 Million+ sales)
Nintendo 64	YES	No	No	Use of cartridges instead of CD-ROMs hampered N64 with many developers preferring large data storage provided by CD-ROM.	Games available by the first Christmas (96'): 8
					Games available in the first 12 months: 24
					Exclusive games for the console: 163
					Ported games from old generation devices for the console: 3
					First-party (second-party) games: 51 (37)
Nintendo 64	YES	No	No	Required to rewrite part of the graphics processor microcode to get most out of the machine (Retrogamer, 2012)	Third-party games: 238
					Average quality of games (First & Second-Party): 79.9/100
					Average quality of games (Third-Party): 68.69/100
					# of hits: 41 (Critical Acclaim= 85+/100)
					# of hits: 42 (1 Million+ sales)

Table 6. Summary of Complementor Disfunctionalities, and Game Release Performance for Sixth Generation Consoles

	Time Compression Diseconomies	Resource Allocation Problems	Lock-in Problems	Important Market Issues	Game Release Performance
Dreamcast	No	No	No	Sega had to prematurely exit hardware business. Rampant piracy further hampered Dreamcast's prospects.	Games available by the first Christmas (99'): 51
					Games available in the first 12 months: 131
					Exclusive games for the console: 155
					Ported games from old generation devices for the console: 50
					First-party (second-party) games: 51 (16)
PS2	YES	YES	No	PS2 was cheaper than standalone DVD players in Japan when it was first released. Combined with its backwards compatibility with PS1 games, it gave an important window for PS2 developers to come up with their games and sustain sales early on.	Third-party games: 193
					Average quality of games (First & Second-Party): 78.78/100
					Average quality of games (Third-Party): 67.42/100
					# of hits: 38 (Critical Acclaim= 85+/100)
					# of hits: 7 (1 Million+ sales)
Xbox	No	No	No	PS2 was cheaper than standalone DVD players in Japan when it was first released. Combined with its backwards compatibility with PS1 games, it gave an important window for PS2 developers to come up with their games and sustain sales early on.	Games available by the first Christmas (2000'): 44
					Games available in the first 12 months: 155
					Exclusive games for the console: 622
					Ported games from old generation devices for the console: 51
					First-party (second-party) games: 114 (44)
Gamcube	No	No	No	PS2 was cheaper than standalone DVD players in Japan when it was first released. Combined with its backwards compatibility with PS1 games, it gave an important window for PS2 developers to come up with their games and sustain sales early on.	Third-party games: 1375
					Average quality of games (First & Second-Party): 73.96/100
					Average quality of games (Third-Party): 69.73/100
					# of hits: 142 (Critical Acclaim= 85+/100)
					# of hits: 161 (1 Million+ sales)
Xbox	No	No	No	PS2 was cheaper than standalone DVD players in Japan when it was first released. Combined with its backwards compatibility with PS1 games, it gave an important window for PS2 developers to come up with their games and sustain sales early on.	Games available by the first Christmas (2002'): 37
					Games available in the first 12 months: 158
					Exclusive games for the console: 208
					Ported games from old generation devices for the console: 21
					First-party (second-party) games: 50 (37)
Gamcube	No	No	No	PS2 was cheaper than standalone DVD players in Japan when it was first released. Combined with its backwards compatibility with PS1 games, it gave an important window for PS2 developers to come up with their games and sustain sales early on.	Third-party games: 747
					Average quality of games (First & Second-Party): 76.29/100
					Average quality of games (Third-Party): 70.2/100
					# of hits: 111 (Critical Acclaim= 85+/100)
					# of hits: 20 (1 Million+ sales)
Gamcube	No	No	No	PS2 was cheaper than standalone DVD players in Japan when it was first released. Combined with its backwards compatibility with PS1 games, it gave an important window for PS2 developers to come up with their games and sustain sales early on.	Games available by the first Christmas (2001'): 20
					Games available in the first 12 months: 128
					Exclusive games for the console: 121
					Ported games from old generation devices for the console: 19
					First-party (second-party) games: 48 (29)
Gamcube	No	No	No	PS2 was cheaper than standalone DVD players in Japan when it was first released. Combined with its backwards compatibility with PS1 games, it gave an important window for PS2 developers to come up with their games and sustain sales early on.	Third-party games: 456
					Average quality of games (First & Second-Party): 76.1/100
					Average quality of games (Third-Party): 70.23/100
					# of hits: 57 (Critical Acclaim= 85+/100)
					# of hits: 27 (1 Million+ sales)

Table 7. Summary of Complementor Disfunctionalities, and Game Release Performance for Seventh Generation Consoles

Xbox 360	Time Compression Diseconomies	Resource Allocation Problems	Lock-in Problems	Important Market Issues	Game Release Performance
	No	No	No		Games available by the first Christmas (2005'): 31
					Games available in the first 12 months: 121
					Exclusive games for the console: 584
					Ported games from old generation devices for the console: 227
	Example Quotes				First-party (second-party) games: 124 (83)
Xbox tools and support were always excellent, and the TCRs and supplementary FTCs [functional test cases - Ed.] were much easier to read, understand, implement, and test. They were also much more lenient about what was acceptable in a lot check, or making exceptions when it made sense to do so. I've heard Sony has greatly improved their tools support for development and testing, though I believe they're still lagging behind what the Xbox 360 had available at launch." (Usher, 2011)				Third-party games: 1187	
				Average quality of games (First & Second-Party): 73.66/100	
				Average quality of games (Third-Party): 71.10/100	
				# of hits: 111 (Critical Acclaim= 85+/100)	
					# of hits: 56 (1 Million+ sales)
PS3	Time Compression Diseconomies	Resource Allocation Problems	Lock-in Problems	Important Market Issues	Game Release Performance
	YES	YES	YES		Games available by the first Christmas (2006'): 17
					Games available in the first 12 months: 96
					Exclusive games for the console: 161
					Ported games from old generation devices for the console: 132
	Example Quotes				First-party (second-party) games: 100 (64)
	"This is how sad the industry is right now. If Sony thought of a way that their architecture designers could somehow add even more power for less money, but made programming a misery - actually made you just want to kill yourself - they would do it. (Gibson, 2007)				Third-party games: 675
					Average quality of games (First & Second-Party): 78.11/100
					Average quality of games (Third-Party): 72.84/100
				# of hits: 89 (Critical Acclaim= 85+/100)	
					# of hits: 27 (1 Million+ sales)
Wii	Time Compression Diseconomies	Resource Allocation Problems	Lock-in Problems	Important Market Issues	Game Release Performance
	No	No	No	It aimed a different demographic than other two consoles and had an innovative controller.	Games available by the first Christmas (2006'): 48
					Games available in the first 12 months: 217
					Exclusive games for the console: 353
					Ported games from old generation devices for the console: 232
	Example Quotes				First-party (second-party) games: 102 (41)
					Third-party games: 653
					Average quality of games (First & Second-Party): 75.7/100
					Average quality of games (Third-Party): 64.64/100
				# of hits: 31 (Critical Acclaim= 85+/100)	
					# of hits: 103 (1 Million+ sales)

Figure 1. Dimensions of Technology: Technological Superiority as User Benefits and Developer Costs for 5th Generation Consoles

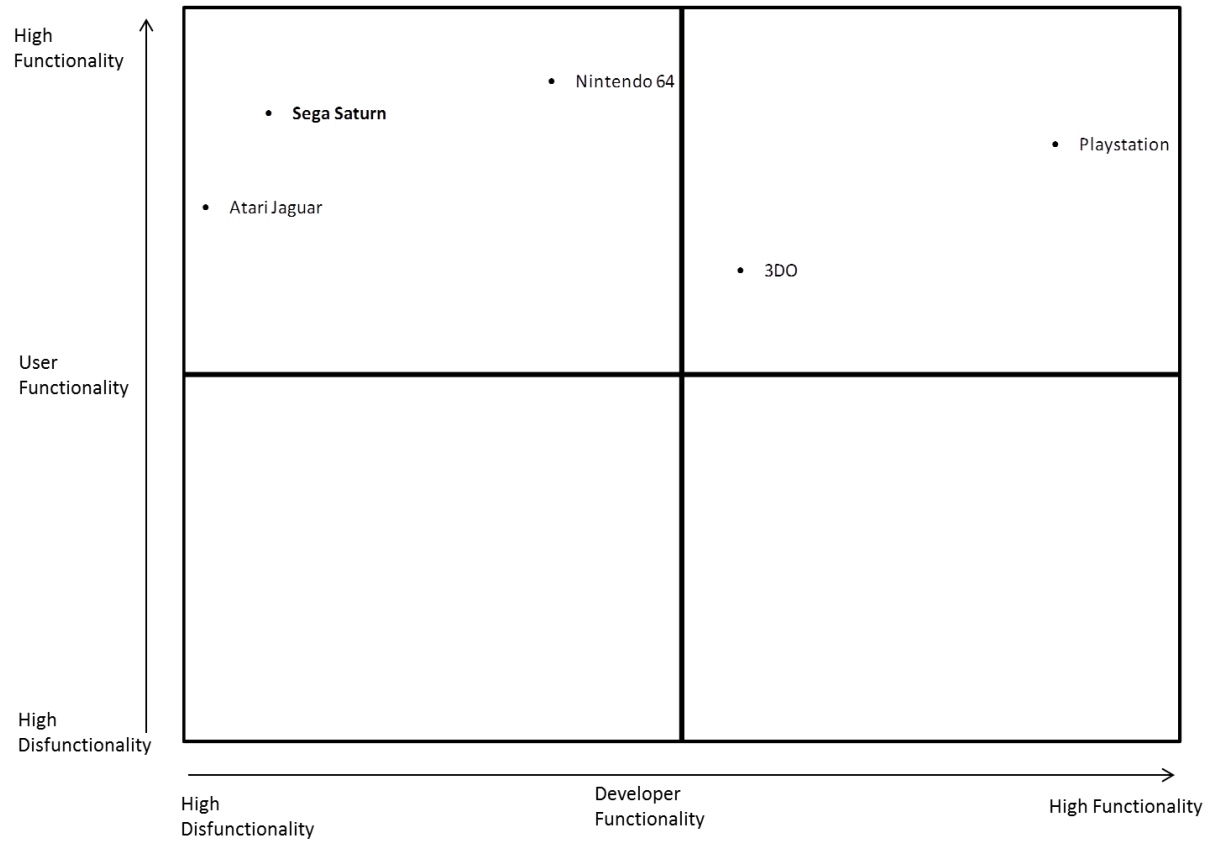


Figure 2. Dimensions of Technology: Technological Superiority as User Benefits and Developer Costs for 7th Generation Consoles

