Risk Management in Video Game Development Projects

Marc Schmalz
University of Washington
Information School
mschmalz@uw.edu

Aimee Finn
University of Washington
Information School
aemeyer@uw.edu

Hazel Taylor
University of Washington
Information School
hztaylor@uw.edu

Abstract

The video game software industry has a reputation for volatile, chaotic projects yet, in spite of dramatic growth in global revenues, surprisingly little academic work has examined these projects. This study reports a preliminary investigation into this under-researched area. We interviewed eight video game producers from a range of companies, using a critical incident method to explore risk management practices and risk perceptions. Our results revealed that in lieu of formal risk management practices, these managers relied on prototyping, pre-production decision points, and agile approaches to contain risk on their projects. Among the risk factors mentioned, two are specific to the unique context of video game development. The risk of failing to match the development strategy to the project was identified as a major cause of problems during the development process, and a new risk – the 'fun factor' – was a key element threatening the success of the final game release.

1. Introduction

In May 2012, entertainment company 38 Studios experienced a high-profile collapse, laying off nearly 400 employees when the company was unable to make payments on a government-backed loan [6]. While the failure of 38 Studios is exceptional in scale and public visibility, this kind of problem is not uncommon in the video game industry, which has a reputation for volatile, chaotic projects. Many become "vaporware," never seeing the light of day, while others eschew the traditional success measures of schedule, budget, and performance [11]. Surprisingly little academic attention has been paid to management of entertainment software projects despite reported global revenues for this industry of $65 billion [5].

Projects in the entertainment software industry – typically involving the development and production of video games – may not have been the focus of academic research, but there is an abundance of work on project management in general, and on information technology (IT) projects in particular. While IT project failures are not uncommon [16, 18], we do have a lot of guidance on how to minimize the risks that threaten these projects, both in the practitioner and academic literatures [see, for example, 13, 17, 21]. However, little work has examined how risk management is used on projects in the entertainment software industry. Thus the purpose of this research was to investigate the practices of entertainment software project managers, and particularly to describe their approach to risk management on their projects. We were interested to understand more about what risk management techniques they use and the challenges they face in anticipating and mitigating threats to their projects.

2. Literature review

The role of project manager has been known for millennia, though the term was first popularized in the 1950s. In the ancient world, a project may have been the creation of a public structure, managed by a master builder. Today, project management is often performed by certified professionals, but any individual charged with the success of a project can be considered a project manager, regardless of title [2]. While project management is rooted in older concepts of engineering, the processes and best practices for software development project management have been studied for decades and are still constantly evolving. Very little work, however, has addressed management of projects in the specific context of entertainment software studios.

At entertainment software studios, project management means bringing a game from concept to market. A game's project manager – usually called the producer – must bring together the art, technical, and management aspects of the process to successfully create a game. In addition to the

978-1-4799-2504-9/14 $31.00 © 2014 IEEE
DOI 10.1109/HICSS.2014.534

employees and management directly involved in producing the game, the producer must also coordinate with external stakeholders such as a licensor and financier, and external play-testers. Thus, while entertainment software projects face many of the same risk factors that threaten traditional IT projects, new risks are introduced by the creative aspects of game development, relationships with additional external stakeholders, and the external environment in which these firms operate.

2.1. Entertainment software development

Creators of entertainment software often view their occupation as a merging of art and technology. In their articles of incorporation, the International Game Developers Association (IGDA) states that one of their goals is to "increase the recognition and respect of digital game developers and the *art form* of digital games" (emphasis added) [4]. The focus on the 'art form' suggests that management of entertainment projects may share more in common with research projects than with standard development projects, in that the creativity required for entertainment projects is similar to the requirements for originality and innovation that are hallmarks of research projects [8]. Entertainment software has many creative components, often involving the creation of graphics, audio, and user interface elements, and management of these creative components is an important aspect of the game and entertainment development endeavor [22].

The creative context and external environment of entertainment software development projects introduces a number of specialized project team roles. While these roles exist for practically every game creation project, a single individual or organization can play several of these roles, depending on the specific project. Key roles include the following:

- Financier: the entity providing capital for the project
- Licensor: the entity owning the intellectual property used in the project
- Publisher: the entity responsible for distribution of the game in electronic and/or physical form
- Platform Owner: the entity owning the platform on which the game will be released (e.g., Facebook for the Facebook Web site, Apple for iOS devices, and Microsoft for the Xbox game console)
- Producer: the individual(s) with project management responsibilities on the project
- Game Designer: the individual(s) responsible for designing the game's rules – or mechanics – and general flavor
- Software Engineer: the individual(s) responsible for producing the code assets for the project
- Audio Engineer: the individual(s) responsible for producing the audio assets for the project
- Graphic Artist: the individual(s) responsible for producing the user interface (UI) and graphical elements for the project.

While publishers have been historically important to the industry, serving as financiers and distributors of entertainment software, their role may be diminishing. Platform owners have implemented electronic marketplaces (Microsoft's Xbox Live Marketplace, Apple's iOS App Store, Nintendo's Wii Shop Channel, Sony's PlayStation Store), Facebook emerged as a major online game platform [14], and crowd-funding has grown in popularity and effectiveness [12], reducing the need for third-party finance and physical distribution.

2.2. Game project management: The producer role and context

In the entertainment software industry, the job title of producer is often assigned to an individual with project management responsibilities for a game creation project. Sometimes working on a team with other producers, the producer coordinates the activities of game designers, artists, audio engineers, and software engineers working on a game project [9]. In addition to his or her own employer, a producer is often responsible to a publisher and, when working on a game based on another company's intellectual property, a licensor. A producer may also coordinate the process of earning a platform owner's approval (such as Nintendo Co., Ltd. approving a title for the Wii home video game console) and the game's rating from the Entertainment Software Ratings Board (ESRB), depending on how the final title will be marketed and distributed.

Producers often wear multiple hats, especially when working on smaller projects. For one of the game titles included in this research, a single individual was company co-owner, publisher, designer, and artist as well as executing the role of producer. Producers typically have a broad and poorly defined skill set. "Historically, many programmers and artists got promoted into producer and project management roles which they were ill-equipped to fulfill" [9, p. 91].

2.3. The current study

As discussed above, while there is a strong body of knowledge about the management of risk on IT projects in general, little attention has been paid to projects in the specific context of the entertainment software industry. Thus the aim of the current study was to begin an exploration of the practice of entertainment software project management, with a view to surfacing similarities and differences for this specific type of project that could contribute to a better understanding of the management of these projects. In this first, exploratory study, we chose to focus specifically on risk management in video game projects, with the expectation that understanding more about how practitioners in this area manage risk on their projects can provide a starting point for further theoretical and best practice development.

3. Method

Since the goal of this research was exploratory and descriptive, we chose an in-depth interview method in order to surface contextual details and gain a rich understanding of the dynamics of risk management on video game projects. We applied an interpretive perspective to the data analysis, to develop insights into the practice of video game project management. By comparing these insights with a framework of risk factors drawn from the literature [21], we were able to highlight key similarities with other types of IT projects, and identify context-specific differences in risk management of video game projects.

3.1. Sample

Our research sample included eight producers in the entertainment software industry who manage video game design and development projects. To find our participants, we used personal contacts as a starting point, and then relied on referrals from snowball sampling to find other participants who would be appropriate for this research [15].

In order to gain a breadth of insight about these types of project, we sought participants from a range of companies, seeking variation in size, types of games produced, and length of time in the industry. Participants ranged from an independent contractor employed by a game publisher and working with contract developers to producers employed as part of a large development team by a company which self-financed and self-published. Only one of the eight participants was female, and five of the eight had more than 10 years' experience in the industry. Interestingly, none of the producers had a background in computers or technology: one had an MBA and the others were humanities graduates. The trend noted above towards reduced publisher involvement was reflected in the current study, where six of the eleven projects discussed involved self-published games and five of the games were self-financed, removing management of a publisher relationship and contract from the producers' responsibilities.

3.2. Data collection

Data collection was in the form of critical decision interviews [10, 20] in which we asked interviewees a series of questions about an applicable project that they had recently managed and the challenges they faced in the course of that project. This process aims to elicit "expert knowledge in situations where the experts have difficulty accessing their knowledge" [21, pg. 52] and also reveals differences in perspectives by focusing on the context of a critical situation. To ensure consistency in the interview structure, we used a semi-structured script. The first two interviews were conducted by the two lead authors, after which all three authors reviewed the two interview transcripts and agreed on clarifications to the script. The first or second author individually conducted the remaining interviews.

The interviews were professionally transcribed, identifying information was removed, and each participant given the opportunity to review his or her transcript for accuracy. No participants requested any major changes, although some offered minor clarifications, which we adopted. The final report was reviewed with one producer, who commented that the findings 'ring true'.

3.3. Coding and analysis

The large body of work on risks in IT projects suggested that a thematic analysis approach to coding would be appropriate, drawing on previous research rather than allowing codes to surface through an emergent grounded theory approach. However, most prior studies focus on risks in in-house IT projects [see, for example, 17] and as noted above the entertainment software industry operates in the very different context of development by a producer for the purposes of sale to game-playing customers. One prior IT project risk study, [21], focused on risks for IT projects implemented by vendors for clients, a situation which does have some similarity to that of

the producers in the current study, although it lacks the external customer aspect. Hence, we developed an initial coding framework from Taylor's framework, with modifications to better fit the unique context of our study, where software studios are working to develop video game software for sale to customers, often in association with external partners such as financiers.

Taylor's original framework [21] considered risk factors within four broad themes of risks – project management, relationships, solution ambiguity, and environment – arising from three different sources – vendor, third party, and client. These source designations were determined by the specific business context of the study: Taylor's project managers all worked for software vendors, handled projects that involved fulfilling a contract for the vendor's product with a client company, and frequently managed interactions with third-party companies that were used to execute portions of the project. We adapted these risk sources for the current study, as shown in Table 1. Software studio (vendor) designates the business directly employing the producer being interviewed, and contractor (third party) refers to any external business over which the software studio has control. Rather than clients, the software studios have partners and users. Partners are businesses, such as licensors, publishers, financiers, and platform owners, with some measure of external control over the software studio or the project. Users, rather than being employees of the client, are any customers or players of the finished game.

The specific risk factors identified in Taylor's framework were used initially without modification, but, as discussed below, some definitions were modified and extra factors were added as coding progressed to better reflect the particular contexts of our respondents and their projects. We used the risk themes primarily as a coding tool to assist with a broad classification of a risk factor before attempting a specific factor identification, which turned debate over classification into a two-step process, positively limiting term choices and easing final risk factor selection. In cases where the researchers were initially uncertain about how to code a statement, the risk theme could be debated and agreed upon first and the risk factor then followed more readily.

In the first pass of coding, the first and second authors coded each transcript separately, and then all three authors compared the versions to reconcile any differences. Disagreements between coders on the exact nature of an identified risk factor sometimes resulted in more careful definition of a risk factor as it applied in the context of the software entertainment industry. For example, the *competition* risk factor's definition was refined to focus on "the need to be competitive with other games in the marketplace" rather than on "over-selling and under-bidding to get projects". In total, we made similar minor modifications to the definitions of six of the original 43 risk factors (*top management support, development strategy, competition, business changes, understanding of requirements*, and *staffing*) in order to focus these factors more fully on the software entertainment context.

Table 1: Risk source definitions

Risk Source - current study	Risk Source – Taylor [21]	Definition
Software studio	Vendor	The organization employing the producer (project manager)
User	Client	The expected end user, player, or consumer of the finished entertainment software project
Partner	Client	Any external organization having a measure of control over the producer or the producer's organization, such as investor, licensor, publisher, or platform owner
Contractor	Third party	An external organization contributing to the project but under the control of producer

In other cases, we realized that our consistent disagreement arose because no strong match could be identified with any existing risk factor, thus indicating the need for a new factor and definition. We added three new factors, two arising from the user source – *audience match* and *'fun factor'* – and one arising from the software studio source – *extent of originality*. The *audience match* factor was defined as "risks associated with reaching the intended audience, gaining their participation, and matching the level of game difficulty with the desired audience". The *'fun factor'* referred to "risks associated with ensuring that the game is enjoyable and 'fun' for the desired audience, and with generating sufficient 'fun' to ensure audience willingness to be monetized". Finally, the *extent of originality* factor refers to "risks arising from the degree of original development required, ranging from the relatively low risk associated with modifying an existing title, to the high risk of aiming to create something entirely new and innovative".

Once all transcripts had received an initial coding pass, the researchers completed a second pass to identify any inconsistencies that may have arisen as the researchers' understanding and the coding framework evolved. Outstanding disagreements in early coding work were often solved with the application of terms which emerged late in the initial coding pass. The goal was to use Taylor's framework accurately while adapting it to accommodate risks outside of those encountered in the context of Taylor's vendor-driven IT projects. The final coding framework is available from the authors on request.

In order to facilitate our analysis, each transcript was color coded to easily identify which project the subject was discussing. In addition to the discussion of specific projects, each transcript also had a "general" subject to identify comments made outside the context of a specific project. A spreadsheet was created to track the findings, with each row consisting of a source-theme-factor triad and each column a specific project that had been discussed, plus a "general" column for each participant. Each factor appearing one or more times in the context of a project was counted as one occurrence in the respondent's project-specific context. Similarly, each factor appearing one or more times in a respondent's general comments was counted as one occurrence in the respondent's general context.

4. Results

The team's analysis involved identifying all risk factors observed in respondents' descriptions of their projects or in their general comments. We compared the risk factors identified by respondents with Taylor's findings for vendor-driven projects. We also examined the transcripts for evidence of formal and informal risk management practices. From this particular group of respondents we found very little evidence of any formal procedures, but informal strategies were frequently applied in lieu of a formal process. We first describe some summary findings about the occurrence of factors, and then examine key factors in more detail. Finally, we describe the informal strategies used by these respondents to keep their projects on track and to mitigate key context-specific risks that are particularly prevalent in this video game development environment.

4.1. High occurrence risk factors

As shown in Table 3, nine factors were identified in specific project contexts by at least half the respondents. The source of six of these factors was the software studio, with one factor arising from user sources and two from partner sources.

For the most part, and not surprisingly, these factors correspond closely to the top risk factors identified in Taylor's study. Two of Taylor's top eight factors are missing here: *client understanding of requirements* and *vendor competition*. The first of these missing factors has diminished correspondence in the current study because software studios are not necessarily working with a client to meet client requirements. Even when working for a client, the goal is to produce a product for market, and hence the key specifier of requirements is often the software studio itself, based on its knowledge of market demand. The second missing factor, *competition*, was observed in three of our producers' project-specific comments. Also similar to Taylor's findings, no contractor (third party) risk factors appeared among our top factors.

Table 3: Top nine risks noted by at least four respondents in specific project contexts

Rank (Taylor rank)	Risk source	Risk factor	Interview count, project context	Project count
1	Software Studio	*Development strategy*	8	10
2 (4)	Software Studio	*Staffing*	7	10
3 (1)	Software Studio	*Schedule and budget management*	7	8
3 (6)	Software Studio	*Inadequate specification*	7	8
5	User	*'Fun factor'*	5	5
5 (2)	Software Studio	*Change management*	5	5
5 (2)	Partners	*Expectations*	5	5
8 (6)	Partners	*Trust*	4	5
8	Software Studio	*Top management support*	4	5

There are, however, two interesting and context-specific factors shown in Table 3. The first is the highest ranked factor in our study, the software studio-sourced risk, *development strategy*. This risk factor, while identified in the Taylor study, did not rank in any of the top 17 spots. The second factor is the new user-sourced risk, the *'fun factor'*, observed

in five of our eight respondents, and not identified at all in Taylor's study. We discuss these factors in more detail below. Finally the *top management support* factor, which features highly in most risk factor studies [see, for example, 17], rounded out our list, but was not ranked in the Taylor study.

4.2. Key risk factors

4.2.1. Software Studio: *Development strategy*. The original Taylor definition [19] referred to committing to the wrong development strategy. We expanded this definition slightly to provide examples of development strategy issues, such as choosing the wrong development platform, incorrectly prioritizing work, implementing an inefficient project structure, failure to allocate time for quality assurance, and insufficient prototyping before production. In such cases, while the specification for the game may have been well designed to capture the original intent and requirements, the underlying decisions about the development approach created problems later in delivering on the intent and the requirements. The *development strategy* risk factor was observed in all eight of our respondents and clearly represented a significant concern for them as they navigated their project developments and reflected on their experiences. Much of the entertainment software industry operates at an intensive pace, with market pressures often dictating the launch schedule. In this environment, development strategy decisions are often made in haste, and key steps such as pre-production prototyping, quality assurance and testing are frequently short-changed:

> *I told them, "You guys are going to want this stuff later." But, you know, it was a case of, my feature could not impact the overall launch schedule of the project. […] So, the outcome was, we shipped basically a robust, very functional shopping experience that simply didn't have the extended feature set that I wanted. And it did cause issues later on, where management was like, you know, "Let's do a sale!" And I'm like, "I can do a sale, and it will go live in a week to 10 days. You can't get a sale tomorrow, because we didn't engineer the system for that." So there were consequences [….] –M05*

4.2.2. Software Studio: *Staffing*. This factor was modified slightly from the original Taylor definition: "All problems with s/w studio staffing – not enough, wrong skills, staff turnover, size of team, *interpersonal issues among staff*" (addition shown in italics). Staffing concerns were raised by seven of the eight respondents, and mentioned in ten of the eleven projects. While most concerns revolved around problems related to engaging enough suitably qualified staff, this category also included producers' criticisms of their own skills in addition to those of other project staff. None of the producers had a background that included formal training in the industry or in project management methodologies. Three producers specifically mentioned their lack of technical and programming skills, noting that this prevented them from being able perform or, more importantly, to evaluate software engineering duties. In one case, this inability to evaluate the work of a programmer almost doomed the project. One producer (M02) noted that while staff in other positions were evaluated for their suitability for the project, producers were not:

> *M02: […We] identified road blocks, either between functional groups or between experience levels within those functional groups. So, we may not get art as pretty as we want it to be because we don't have a high quality artist, or the animations as fluid as they need to be because we don't have experienced animators. So there's that sort of risk assessment that's done for the internal part [….]*
>
> *Interviewer: It sounds like you were analyzing things like whether or not the art staff was up to the project as specified. Was there an analysis of whether or not the producing staff was experienced enough?*
>
> *M02: Uh, no. No.*

Concerns about staffing, and particularly the suitability of staff skill-sets, were also raised with respect to partner and contractor staff, though less frequently. One subject speculated that the high level of concern for staff competency was caused, in part, by the low level of standardization in industry jobs and job titles.

4.2.3. Software Studio: *Schedule & Budget*. This factor and its definition remained unchanged from the original Taylor definition [19]: "Risks associated with keeping to preset schedule and budget, including failure to develop work breakdown structure; failure to adequately monitor progress on tasks; and blow-outs caused by unexpected problems, beyond the pre-set contingencies". And, as in the Taylor study, schedule and budget risks were of frequent concern, being observed in seven of our eight respondents.

With little variation, producers referred exclusively to labor costs when asked about project budgets:

> *[We] outsource to other countries where the exchange rate helps. So we're getting these done for about [external labor cost] per, plus my salary, plus the internal resources of the team, so I'd probably say it comes in—the total cost of the project with everybody's pay and whatever, probably comes to about [total cost], because those guys live in California, so they make a decent check. –A01*

However, the entertainment software projects in this study varied greatly in budget, schedule, and features, and we did observe differences with the schedule and budget factor in terms of the type of project being discussed. Projects which received funds from a publishing partner were seen as most sensitive, with great attention applied to budget and schedule while protecting a minimum feature set. In contrast, in self-funded projects, schedule and budget were given low priority compared to ensuring a desirable feature set and maximizing the audience appeal of the game:

> *The risk, that we all understood, was that it's going to be more money and more time. It's going to be harder to recoup our investment in this thing. […] But for a long time, we didn't have a publisher, so we didn't have anyone, any outside check on what we were doing. There were no outside expectations for ship date or anything else going on. So we were just kind of our own weird little autonomous unit. –M06*

4.2.4. Software Studio: *Inadequate Specification*. This factor is based on Taylor's *understanding of requirements* factor, with the addition shown in italics to reflect key concerns raised by current study respondents: "Software studio has inadequate grasp of requirements, e.g. specification is at too high a level, or unclear, *or fails to identify features that are required,* or has omissions, or is based on unfounded assumptions". Seven of the eight respondents referred to this as a risk that arose frequently in their projects.

The *inadequate specification* factor focuses primarily on the specification, often referred to as a game design document (GDD) in entertainment software. Participants had a wide variety of options for defining their game, from small projects with no formal documentation to large, formal GDDs for massively multiplayer games based on a partner's intellectual property. Lighter specifications seemed generally preferable at all scales as producers traded strongly bounded designs for flexibility. However, precise specifications are required when dealing with a licensed property or outside sources of funding.

> *[When] I was working for-hire, it was a traditional waterfall-type approach. This is our schedule, these are our tasks, these are our completion dates, this is when we deliver it, this is the milestone schedule, these are the things that I will give to you in order to get paid. When you're working for the publisher, we kind of drifted into a much more agile scrum type of thing. These are things that we want to focus on this week, and the following week, okay, well, let's iterate on this, and what about this other thing? It was a much more fluid kind of development process. –M01*

4.2.5. User: *'Fun Factor'*. A key component to building a successful video game is the "fun factor." Producers not only had to worry about delivering a completed project, but they also had to chase an elusive quality of fun that would draw in an audience. Seven of the eight producers interviewed in this study mentioned *'fun factor'* as a key risk, either in a specific project context, or in their general comments. Making a game "fun" was seen as key to success and was an ongoing concern for the producers. This concern relates not only to usability aspects of a product, which are present in all software development, but, we believe, also to a unique aspect of developing entertainment software because of the artistic elements that must be fostered. A video game producer balances technology and art, trying to make the best game with the technology they have to use.

Producers must not only deliver a strong creative product, but have to create a product that their audience will buy. This involves targeting an audience and making products that will appeal to them. The game must be designed and calibrated so that it is challenging, but not too difficult. A style that will appeal to the target audience must be selected for the art, and the characters must be engaging. Game play must be smooth and intuitive and draw the player into the world constructed by the game. All of these things must be done within the constraints of the development technology. Additionally, producers must liaise with marketing specialists to ensure that these key aspects are well communicated to the target user audience. Even with all these efforts, 'fun' is an ephemeral trait, and the challenge of finding and attracting an audience that will enjoy the game is a foremost issue as producers develop their games, as M04 explained:

A design risk is a market utility risk. It's a, "Do I make—"? It's both a creative risk and a technical risk and a business risk all mushed into one. Because the question is, "Will I make a product that the market will love and find acceptable and want more of?" And it's really, really easy, particularly in game development, to get, to start developing in your own little universe, never test the game externally, and end up with some horrible piece of crap. And so, by iterating regularly, by play-testing regularly, you end up ensuring that the thing that you make is appealing to the outside world. -M04

4.3. Strategies for Managing Risks

Participants indicated informal risk practices at most studios. Rather than formal risk management techniques, producers used an indirect approach to risk management through project structure and development practices. Most indicated an emphasis on prototyping and pre-production phases of development to reduce budget and schedule risks. Since production includes costly art and audio asset generation, a pre-production phase which generates a thorough design will minimize production overruns due to revisions. One subject, with nine years of experience in the producer role, rejected the utility of formal risk management even though he performed a required risk analysis for the project we discussed:

This notion of risk management is coming at it from, "How are we going to handle risks?" is kind of coming at it from the wrong angle, because the only way to handle risk is either to cut scope or to just not have a product. And when you're dealing with physical media, when you're dealing with console developers, you can't cut scope. There's a floor to the point of cutting scope, and so you can have a viable game that will pass their certification standards, with Nintendo or Microsoft, or Playstation or whatnot. So I always laugh when someone says, "Well, what's your risk assessment?" Well, my risk assessment is, I have a game or not, and here's what I need to have those things. –A02

4.3.1. Prototyping and "Waste". As noted above, the producers interviewed for this study did not use many formal risk management practices. However, the project structure itself often mitigated risk by emphasizing prototyping, pre-production and the early jettison of troubled projects. One producer estimated that as much as 75% of the games at the studio were cut before they went into production. Another made a similar estimate, adding that ideas that failed once were often filed away in case they could be revisited later. This indicates a huge amount of waste, in terms of the number of projects that are started but do not make it to production. But this approach also allows studios to take risks without having to move forward if the investment becomes too great.

I read, and at first when I read it I couldn't believe it. Seventy-four percent of games that get started never see the light of day. They never get published. I thought, "God, that seems really high." Then I started thinking about the places I've worked, and yeah, that's probably, probably true. I mean, a lot of companies will spend a certain amount of time doing a prototype, a beta, a demo, that for whatever reason they never get published and never make it all the way through. -M01

[It] seems very radical, but these are also very small projects. So canceling projects when we have five or six going on at once, canceling—my expectation going into making this company was that we would cancel four out of five projects, that four out of five projects would be bad games, and we would just cancel them. That hasn't ended up being the case; we've canceled maybe one out of five. But it's not a crazy thing, it's not an unknown thing to cancel a project. -M04

With the rise of Internet game distribution, studios can make small-budget games and release them online. If the game becomes popular and is bringing in profits based on different schemes for monetization, then the studio will keep developing it and adding features. This is a lower-risk way to test ideas and see what sticks with the audience:

And we also kind of have a unique business model. We're a premium company. Basically, we create this game and you can download it and play it for free. Totally free. But there are things in the game that, hey, you want the magic unicorn? Oh, well, that'll cost you a quarter. You want to go to the new level? Oh, that'll cost you a quarter. So the idea is that I provide you with entertainment, and hopefully it's compelling and addictive enough that you want to actually spend some money to play that game. So, that's new. That's different. I'm used to getting paid up front for

delivering the game. It's a much different thing. -M01

The project was a mobile game, small—the intention was to release on iOS and then branch into Android. -M02

4.3.2. Agile practices. Agile practices – though rarely implemented in a strict manner - were indicated as a risk mitigator by four of the eight producers. Agile software development "promise[s] increased customer satisfaction, lower defect rates, faster development times, and a solution to rapidly changing requirements" [1]. Many consider agile to be a response to risk in software development by giving designers tools to combat unforeseen challenges. Agile development is designed for flexibility, but it brings its own inherent risks, such as losing sight of a project's original purpose through constant adaptation [7]. Some scholars and professionals have offered suggestions for overcoming the potential risks of agile adoption. For example, Concha, Visconti, & Astudillo [3] suggest a "commitments framework" for agile project managers, designed to help developers stay on course to fulfill client needs. In the current study, we rarely observed 'pure' agile practices, but the modified practices used by these producers worked well with the emphasis on pre-production and prototyping as risk mitigation approaches. Producers took what they knew about agile and designed a system to meet their project management needs. In addition to mitigating risk, agile practices seem to fit well with the emerging process of publishing a game and then continuing to develop it even after it is live, responding to user feedback to improve the game:

They had the agile backlog and burn-down charts and things like that, to show just how the teams were performing, but there was not any true risk management. -M05

In more agile methods... these things can overlap. For example, one of your, one part of your project may be in production—for example, level design or art assets—and another part may continue to be in pre-production—for example, you might have some of the core combat mechanics that are actually, you're tuning and figuring out if they're fun for a reasonably long period of time. - M04

5. Discussion

While entertainment software projects have a lot in common with other types of IT projects, their emphasis on creativity and the 'art form' coupled with their focus on delivering a product for sale result in a unique development environment, which is reflected in the different risk concerns found in this study. In particular, the three new risk factors identified - *audience match, 'fun factor'* and *extent of originality* – illustrate the critical challenges for video game developers of operating in a highly competitive market environment with heavy dependence on volatile consumer sentiment and demand.

Perhaps the most surprising finding was that we found very little evidence of any formal risk management in this environment. However, although participants eschewed notions of formal risk management, we found considerable evidence of informal practices at most studios. Specifically, producers used an indirect approach to risk management through project structuring and development practices. For example, most respondents indicated an emphasis on prototyping and pre-production phases of development, in order to clarify requirements and demonstrate proof of concept. Since video game production includes costly art and audio asset generation, this pre-production phase, which generated a thorough design, also helped to reduce overall budget and schedule risk by minimizing production over-runs due to revisions.

There was also a clear distinction between projects funded by a publishing partner and those that were self-funded. When a partner was involved, producers reported greater sensitivity to the need for control of budget and schedule while still protecting a minimum feature set. In self-funded projects, schedule and budget were given low priority, and emphasis was placed on including features that would best support the 'fun factor'.

In general, studios which internalize the roles of licensor (by creating a new intellectual property), financier, and publisher tend to have much greater flexibility in terms of schedule and game features while accepting full exposure to project profits or losses. In contrast, work-for-hire studios have the comfort of a known return on their labor investments, but have contractual obligations to one or more external organizations for a game's features and schedule. For independent studios, the economy, flexibility, and adaptability afforded by electronic distribution have allowed them to shift from big-budget, contract-defined games to smaller games that can be treated as eternal prototypes and later modified to be released on multiple platforms.

6. Conclusions

The entertainment software developers in this study placed a relatively low value on formal risk management, but informal practices, such as the prototyping, agile approaches, and pre-production decision points observed here, were well suited to the particular demands of producing a product for market rather than developing or implementing systems for in-house or client use. Of course, conclusions cannot be drawn from such a limited sample, but the results here do point up the need for further investigation into how best to match project management and risk management methodologies to the type of project work being undertaken.

The volatility of the entertainment software industry may necessitate informal project management practices in order to allow for rapid and flexible adaptations and responses to the market. However, other factors observed among this set of producers may indicate areas that software studios could focus on in order to increase project success. These include the need for project management and technical training for internally promoted personnel, and recognition that while agile development practices may address the demand for flexibility in response to the marketplace, careful risk management is still required to ensure projects meet their success targets. Finally, the view of entertainment software as a blend of art and technology suggests the need for careful constitution of development teams to ensure the right balance of skills.

7. References

[1] Boehm, B. and R. Turner, "Using Risk to Balance Agile and Plan-Driven Methods." Computer, 2003. 36, (6) pp. 57-66.

[2] Chiu, Y. C., An Introduction to the History of Project Management: From the earliest times to A.D. 1900. Eburon, Delft, 2010.

[3] Concha, M., M. Visconti, and H. Astudillo, "Agile Commitments: Enhancing Business Risk Management in Agile Development Projects." Lecture Notes in Computer Science, 2007. 4536, pp. 149-152.

[4] Game Developers' Association, Inc., "Amended and Restated Articles of Incorporation of International Game Developers' Association, Inc." 2005. [Available from: http://www.igda.org/sites/default/files/IGDA_ArticlesOfIncorporation_05.pdf]

[5] Gaudiosi, J., "New Reports Forecast Global Video Game Industry Will Reach $82 Billion By 2017." July 18, 2012. [Retrieved from http://www.forbes.com/sites/johngaudiosi/2012/07/18/new-reports-forecasts-global-video-game-industry-will-reach-82-billion-by-2017/]

[6] Hidalgo, J., "Fallen Kingdom: 38 Studios' collapse and the pitfalls of using public money to support tech companies." September 7, 2012. [Retrieved from http://www.engadget.com/2012/09/07/38-studios-collapse-and-risk-of-public-private-partnerships/]

[7] Hijazi, H., T. Khdour, and A. Alarabeyyat, "A Review of Risk Management in Different Software Development Methodologies." International Journal of Computer Applications, 2012. 45 (7), pp. 8-12.

[8] Keller, R. T., "Transformational Leadership, Initiating Structure, and Substitutes for Leadership: A longitudinal study of research and development project team performance." Journal of Applied Psychology, 2006. 91(1) pp. 202-210.

[9] Kerr, A., The Business and Culture of Digital Games: Gamework/gameplay, Sage, London, 2006.

[10] Klein, G. A., R. Calderwood, and D. MacGregor, "Critical Decision Method for Eliciting Knowledge". IEEE Transactions on Systems, Man, and Cybernetics, 1989. 19(3), pp. 462-472.

[11] Kohler, C., "Hi, I'm the Game Industry, and I'm addicted to Vaporware." Wired Magazine. January 25, 2013. [Retrieved from http://www.wired.com/gamelife/2013/01/vaporware-addiction/]

[12] The New York Times Company, "Three Years of Kickstarter Projects", April 30, 2012. [Retrieved from http://www.nytimes.com/interactive/2012/04/30/technology/three-years-of-kickstarter-projects.html?ref=technology&_r=0]

[13] Project Management Institute, A Guide to the Project Management Body of Knowledge (4th ed.), Project Management Institute, Newton Square, PA, 2008.

[14] Rose, M., "As EA Bows Out, Facebook Says Games are Strong as Ever", April 16, 2013. [Retrieved from http://gamasutra.com/view/news/190577/As_EA_bows_out_Facebook_says_games_are_strong_as_ever.php]

[15] Sarantakos, S., Social Research, Macmillan, Basingstoke, 1998.

[16] Sauer, C., A. Gemino, and B. H. Reich, "The Impact of Size and Volatility on IT Project Performance", Communications of the ACM, 2007. 50(11), pp. 79-84.

[17] Schmidt, R., K. Lyytinen, M. Keil, and P. Cule, "Identifying Software Project Risks: An international Delphi study", Journal of Management Information Systems, 2001, 17(4), pp. 5-36.

[18] Standish Group, Chaos Rising. West Yarmouth, MA, 2005.

[19] Taylor, H., Risk Management and Tacit Knowledge in IT Projects: Making the implicit explicit. (PhD Dissertation), Queensland University of Technology, Brisbane, 2004.

[20] Taylor, H., "A Critical Decision Interview Approach to Capturing Tacit Knowledge: Principles and application. International Journal of Knowledge Management, 2005. 1(3), pp. 25-39.

[21] Taylor, H., "Risk Management and Problem Resolution Strategies for IT Projects: Prescription and practice. Project Management Journal, 2006. 37(5), 49-63.

[22] Zackariasson, P., M. Walfisz, and T. Wilson, "Management of Creativity in Video Game Development". Services Marketing Quarterly. 2006. 27 (4) pp. 73-97.