Western University
Scholarship@Western

Electrical and Computer Engineering Publications   Electrical and Computer Engineering Department

9-2016

Critical Success Factors to Improve the Game Development Process from a Developer's Perspective

Saiqa Alemm
Zayed University

Luiz Fernando Capretz
University of Western Ontario, lcapretz@uwo.ca

Faheem Ahmed
Thompson Rivers University

Follow this and additional works at: https://ir.lib.uwo.ca/electricalpub

Part of the Software Engineering Commons

Critical Success Factors to Improve the Game Development Process from a Developer's Perspective

Saiqa Aleem [1], Luiz Fernando Capretz [2], *Senior Member, IEEE, Member, ACM*, and
Faheem Ahmed [3], *Senior Member, IEEE*

[1] *College of Technological Innovation, Zayed University, Abu Dhabi, 144534, United Arab Emirates*

[2] *Department of Electrical and Computer Engineering, Western University, Ontario, N6A 5B9, Canada*

[3] *Department of Computing Science, Thompson Rivers University, Kamloops, BC V2C OC8, Canada*

E-mail: saiqa.aleem@zu.ac.ae; lcapretz@uwo.ca; fahmed@tru.ca

Abstract   The growth of the software game development industry is enormous and is gaining importance day by day. This growth imposes severe pressure and a number of issues and challenges on the game development community. Game development is a complex process, and one important game development choice is to consider the developer's perspective to produce good-quality software games by improving the game development process. The objective of this study is to provide a better understanding of the developer's dimension as a factor in software game success. It focuses mainly on an empirical investigation of the effect of key developer's factors on the software game development process and eventually on the quality of the resulting game. A quantitative survey was developed and conducted to identify key developer's factors for an enhanced game development process. For this study, the developed survey was used to test the research model and hypotheses. The results provide evidence that game development organizations must deal with multiple key factors to remain competitive and to handle high pressure in the software game industry. The main contribution of this paper is to investigate empirically the influence of key developer's factors on the game development process.

Keywords   developer's perspective, software game, empirical investigation, good-quality game, game development process, game developer's factor

1   Introduction

The first software game was created half a century ago. In the world of software gaming, many things have changed during this time period. Now the software game industry has reached the point that it rivals other well-established industries such as music and cinema. As a result, the software gaming business has grown enormously, has made billions of dollars in profit, and has started to mature over time[1]. The game development process has also had an impact on the industry, which now counts on special methodologies and mature processes for its development, ultimately leading to an enhanced game development process. Game developers try to produce games that are different from any other game in the market. This difference can be achieved through introducing new perspectives, new gameplays, new genre combinations, enhanced graphics, or new characters. Therefore, almost all games must be novel, and their success depends on their overall quality[2]. Only good-quality games are able to retain their players, and this has become an important factor for any software game to succeed commercially. In other words, if a game is not of good quality, players can easily switch to another game. Hence, it has become mandatory for the software game industry to try to morph and adapt to the preferences and demands of its players.

One of the main concerns in game development process is that developers need to follow best practices and procedures from software engineering disci-

---

pline to develop good-quality games. The game development process involves four main phases: concept, pre-production, production, and post-production[3]. It consists of various activities such as synopsis, background research, script writing, visualization and concept art, level and interaction design, animation, programming, media editing, integration, testing, and publishing. Software games are also characterized based on the category into which they fall, which is called the genre of the game. Genres include action, shooters, fighting, racing, adventure, sports, role playing, strategy, simulations, puzzles, dance, music, and others. Each genre has its own requirements which must be taken into consideration during the pre-production phase. For this reason, software game development is considered as a complex process that involves multidisciplinary collaborative team efforts and processes (including sound, gameplay, art, artificial intelligence, control systems and human factors) to develop a creative product. Fundamentally, game development is a form of software development process with several additional requirements such as creative design, artistic aspects, and visual presentation[4-5]. In this context, game development organizations can apply the same software engineering principles to improve their development processes. However, many studies have discussed the challenges of applying software engineering principles to the game development process[5].

Kultima[6] highlighted these challenges from the game design perspective. Blow[4] discussed their implications from the perspective of technical frameworks and development techniques. Blow[4] and McGill[7] discussed issues even for the required technical skills for game development. Software game development also requires a range of skills that include design, project management, development, and asset creation. It also involves team members from heterogeneous disciplines, e.g., game designers, artists, programmers, and software developers. Knowledge of best practices for game development is very important and has become crucial to sustain the growth of the software game industry. Finally, this knowledge will help game developers make correct game development decisions at the right time. An investigation of key success factors from a developer's perspective will contribute to the understanding of current development process implications and will help developers improve the game development process.

Exploring diverse developers' preferences for software game development will provide a significant benefit to improve the development process by generating valuable insights. No research has been done to date on including developer-centred factors in the software game development process. This study will help identify key factors empirically from the developer's perspective, an effort that will ultimately help improve the software game development process to produce good-quality software games. To identify key factors, a quantitative survey was conducted, and the results are reported here. The survey was used to test the research model and several hypotheses. Finally, the results show that the consideration of key factors from a developer's perspective helps identify important game development choices and their implications for the current process.

1.1 Research Background

The software game domain covers a great variety of player modes and genres[8-10]. The complexity of digital games has posed many challenges and issues in software development because it involves diverse activities in creative arts disciplines (storyboarding, design, refinement of animations, artificial intelligence, video production, scenarios, sound effects, marketing, and finally sales), besides technological and functional requirements[11]. This inherent diversity leads to a greatly fragmented domain from the perspectives of both underlying theory and design methodology. The software game literature published in recent years has focused mainly on technical issues. Issues of game production, development, and testing reflect only the general state of the art in software engineering. Pressman[12] stated that a game is a kind of software which entertains its users, but game development faces many challenges and issues if only a traditional software development process is followed[5,13].

Many researchers have discussed game development challenges. Pertillo et al.[13] surveyed the problems faced by game development organizations. The overall game development process combines both an engineering process and the creation of artistic assets. Ramadan and Widyani[14] compared various game development strategies from a management perspective, and some researchers[15-17] have proposed frameworks for game development. To effectively manage and improve the game development, key developer's factors are required. Tschang[18] and Petrillo et al.[13] highlighted the issues in the game development process and its differences from traditional software development practices.

In traditional software engineering, the development phase usually involves activities like application design and implementation, and the production phase

is when the software actually runs and is ready for use. However, in the game development, the production phase includes the development process, which is the pre-production phase of the software engineering process, and the production phase of software engineering is actually the post-production phase of the game development life cycle[20]. Therefore, the game development is different from the traditional software engineering process, and many researchers[5] have studied the challenges faced by this domain. Kanode and Haddad[5] stated that an important incorrect assumption has been made that game development follows the waterfall method. More recently, researchers have agreed that it must follow the incremental model because it combines the waterfall method with an iterative process. Petrillo et al.[13] reported a major concern that developers for software creation in the game industry commonly use very poor development methodologies. The game development life cycle (GDLC) is the object of questions on many forms, which attempt to determine what types of practices are used. However, this question has no single answer. The most prominent observation made in above studies is that to address the challenges faced by GDLC, more rigorous software engineering strategies must be used. However, the proposed GDLCs[14,19-21] do not ensure the quality of the development process. O'Hagan et al.[22] published a systematic literature review of software process models used for game development. They concluded that agile and hybrid approaches are used by most organizations for game development. They also reported that Scrum[5], Kanban[23], rapid development application (RAD)[24], XP[25], and incremental[5] methodologies are used by game development organizations. The major difference in software development and game development is in the design phase because the design of game may undergo major change in late development phase. The other differences are content development and quality criteria. Managing game development has become a much harder process that anyone could have initially imagined, and because of the fragmented nature of the domain, no clear picture of its advancement can be found in the literature.

From the above discussion, it can be easily concluded that game development process is different from traditional software development process. Kasurinen et al.[26] argued that current software engineering knowledge is unable to bridge the gap between software engineering and certain aspect of game development. The overall development process to produce a game in-

cludes art, audio and gameplay other than software development discussed above. In the game development process, the content and production activities are performed in tandem with the development and engineering activities. Further, it is well agreed that the game development process is a multidisciplinary activity that involves the merging of creative and technical talent to bring a concept to life, where the main activities can be categorized into content and production, and engineering at each phase of the development process.

Moreover, sometime game development organizations reduce their development process due to high competition and extreme market demand so that they can be first to market[27]. This reduction of the development process definitely affects game quality. Therefore, they do not strictly follow the software engineering standards and practices. Because of these types of complex project-management tasks, the game development process diverges from traditional software development. Nevertheless, the differences between software engineering and game development are not exclusive; it seems that traditional software development neither fully supports game development activities nor provides process assessment procedures[28]. Therefore, we need key success factors to improve game development process that may overlap with traditional software development factors or just exclusive to game development. Consequently, it has become important now to investigate the critical success factors for game development organizations in developing good-quality games from a developer's perspective.

1.2 Research Motivation

Game development has become incredibly challenging due to rapid changes in game technology such as game platforms, game engines, and reuse of code modules for different genres. During the 1990s, game development was usually carried out by small teams and involved simple architectures consisting of 2D graphics, sound, simulation, and input/output streaming. The first software games were developed by a few talented individuals with diverse backgrounds like mathematics, computer science, and physics with no educational background in engineering or computer science. At that time, developers mainly focused on how to develop interesting games rather than on architecture or software engineering principles. The current success of the game industry, continuous enhancements in game technology, and the need to meet the ever-higher expectations of

the players resulted in a complex game development process.

The main research motivations behind this study are the rapid and continual changes in technology and the severity of competition in game development organizations. Ultimately, these factors will not only affect the business, but also have a major impact on the game development process. Nowadays, games are developed by large teams because game projects have grown in size and complexity[4]. Various stakeholders are involved in the development process and have different expectations and world views. For example, the game designer does not know the level of complexity involved in implementing artificial intelligence to represent the behaviour of a non-player character. A software engineer may think that some features in the game design document are infeasible to implement due to time deadlines or technical constraints. Another important requirement that must be part of the game is the fun, flow, and enjoyment factors. The game development processes have different phases and are influenced by many factors. Identifying the key success factors in a game development process is extremely important for sustaining the economic growth of the software game industry.

However, very little research has been reported in the academic literature about key success factors for the game development process. Many topics in software games need attention from researchers and are highlighted by some studies[22,29-30]. Moreover, game developers and researchers have different points of view. Basically, game developers prioritize the game development process by rapid creation and implementation of content. On the other hand, scientists and researchers prioritize investigation and research into the individual components of a system. Researchers do not have resources to develop a standard game, whereas developers never publish the results of their experience. This indicates that there is a need for the collaboration between researchers and developers that will be ultimately beneficial to game industry standards. This study also attempts to fill this communication gap between researchers and developers. Above discussed facts, motivated us to carry out empirical investigation of key success factors that can help developers to improve their development practices. It will ultimately enable them to develop good-quality games.

The rest of the paper is organized as follows. Section 2 provides a literature review of identified factors, Section 3 presents the research model, Section 4 describes the research methodology used for this study, Section 5 presents the results of the empirical investigation, Section 6 provides a discussion, and finally, Section 7 concludes the study.

2   Literature Review and Proposed Hypotheses

In recent times, the software game industry (SGI) has witnessed unprecedented growth. To succeed in a highly competitive environment, game developers must bring innovative, good-quality games to the table. Identifying key success factors to improve the game development process will help developers maintain the pace. Key factors in the game development process are the least addressed area in software game research. Various factors have been identified from a literature review of published articles on software games as a basis for discussion of the game development process.

Table 1 briefly presents the identified factors, with references for each. The identified factors and the related literature are described in the following subsections.

2.1   Team Configuration and Management

The development of software games involves multidisciplinary team configuration and management. More specifically, team configuration and management are considered critical to the success of any game development project. Game development requires intensive team management[31]. Team management can be defined as the process of administration and coordination among groups of individuals who are performing specific tasks[32]. It involves forming different groups, establishing collaboration among them, setting objectives for a common set of interpersonal dynamics among team members, and performance appraisals. The game development process also involves the configuration and management of multidisciplinary teams or teamwork projects and the management of collaboration among them. The term "teamwork" refers to a group of individuals who are completing a specific task[33]. The term "collaboration" can be defined as the level of shared understanding and coordination among teams and the maintenance of this level[34].

Very few research studies have investigated the importance of multidisciplinary team configuration and management in software game development. Musil *et al.*[25] highlighted the importance of heterogeneous team collaboration in the video game development process. They proposed a method based on the Scrum

Table 1. Identified Factors from a Developer's Perspective

Factors	References
Team configuration & management	Claypool and Claypool[31]; Sundstrom *et al.*[32]; Musil *et al.*[25]; Tran and Biddle[35]; Stacey *et al.*[36]; de Barros *et al.*[37]
Game design document management	Kasurinen *et al.*[38]; Bosser[39]; Callele *et al.*[40]; Callele *et al.*[41]; Reyno and Cubel[42]; Almeida and da Silva[43]; Ahmed and Jaafar[44]; Bringula *et al.*[45]
Game engine development	Robin[3]; Sherrod[46]; Cowan and Kapralos[47]; Hudlicka[48]; Wu *et al.*[49]; Rodkaew[50]; Vanhutupa[51]; Sousa and Garlan[52]; Aitenbichler *et al.*[53]; Pimenta *et al.*[54]; Neto *et al.*[55]; Peker and Can[56]
Game asset management	Llopis[57]; Hendrikx *et al.*[58]; De Carli *et al.*[59]; Phelps[60]; Pranatio and Kosala[61]; Lasseter[62]; Xu and Wang[64]; Chehimi *et al.*[65]; Manocha *et al.*[66]; Pichlmair and Kayali[67]; Migneco *et al.*[68]
Quality of game architecture	Wang *et al.*[69]; Amendola *et al.*[70]; Lukashev *et al.*[72]; El Rhalibi *et al.*[73]; Jhingut *et al.*[74]; Kosmopoulos *et al.*[75]; Al-Azawi *et al.*[76]; Segundo *et al.*[77]
Game test management	Redavid and Farid[78]; Helppi[79]; Schultz *et al.*[80]; Wilson[81]; Marri and Sundaresaubramanian[82]; Kasurinen and Smolander[83]; Al-Azawi *et al.*[84]; Omar and Jaafar[85]; Strååt and Warpefelt[86]
Programming practices	Robins[3]; Sarinho and Apolinario[87]; Czarnecki and Kim[88]; Chen *et al.*[89]; Anderson[90]; Xu and Rajlich[91]; Zhang *et al.*[92]; Wang and Norum[93]; Meng *et al.*[94]

methodology to improve workflow integration and collaboration between heterogeneous game development team members. The proposed process separates the pre-production, production, and post-production phases. Management through the collaboration and integration of heterogeneous disciplines in game development is achieved by executing daily heterogeneous discipline-specific workflows in a sprint iteration adjusted by daily scrums. They claimed that this approach will enable each discipline to use the workflows in which they are most proficient in accordance with the demands and pace of other involved disciplines.

Tran and Biddle[35] discussed the collaboration factor for team management in serious game development. They explained that the collaborative process is based on ethnography and a qualitative approach. The proposed model includes many factors such as physical resources, social relationships, organizational goals, and team knowledge. They conducted a case study that determined that collaboration among multidisciplinary teams requires teams to communicate frequently, to respect each other's contributions, and to share the same model and goals for game development. Stacey *et al.*[36] and Barros *et al.*[37] also investigated the collaboration factor in multidisciplinary game development teams and the development of computer games.

To determine whether proper team configuration and its management have any impact on the game development process, "team configuration and management" was selected as an independent variable, as shown in Fig.1. Hence, Hypothesis 1 and corresponding null hypothesis can be stated as follows.

Hypothesis 1 (H1). *Team configuration and management have a positive influence on the enhanced game development process.*

Null Hypothesis. *Team configuration and management have no influence on the enhanced game development process.*

2.2 Game Design Document Management

The game design document (GDD) has also been identified as an important factor in improving the game development process. The GDD is the outcome of the pre-production phase of game development. It is developed and edited by the game design team to organize the team's efforts and development process. The form of the GDD varies widely across studios and genres. Basically, the GDD includes the goals of the game, the genre of the game, the overall flow, the story behind the game, the characters and their dialogue, special effects, the number of elements and feature fits within the game, and feature creeping information if required. Typically, this document is developed to express the concept of the game and to provide a basis for requirements engineering in the game development process. Game designers can trace back all their efforts to the requirement analysis in the GDD.

In the game development process literature, researchers have explored the importance of the game design document and its management in various ways. Some of them have highlighted the importance of the GDD by discussing the importance of requirements engineering in game development. For example, Kasurinen *et al.*[38] highlighted the importance of requirements
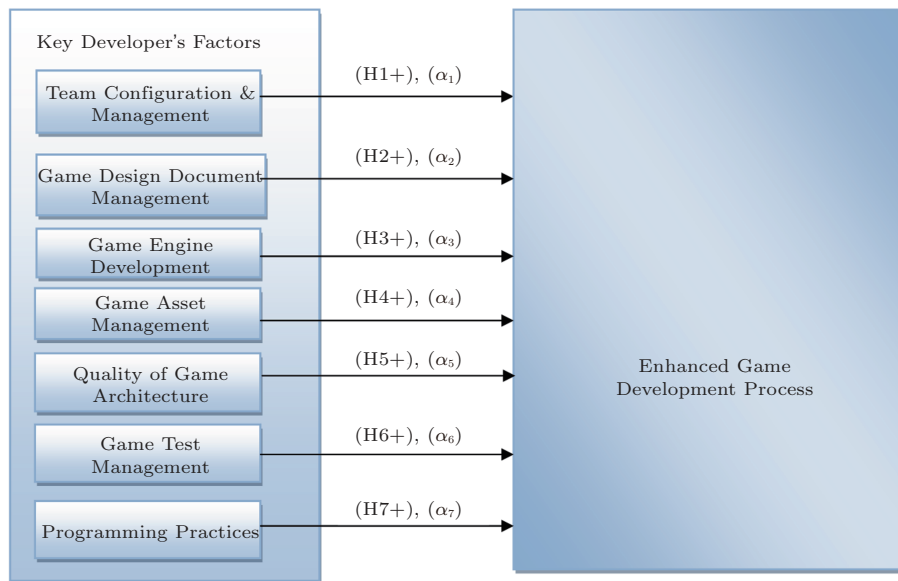
Fig.1. Research model.

engineering in the game development process. They interviewed 27 software professionals from game development organizations to obtain an insight into their development process. The findings of the study showed that the professionals follow approaches or methods that are somewhat comparable to requirements management and engineering, but not to particular requirements engineering practices. Bosser[39] suggested that massively multi-player game design needs a prototyping tool and proposed a framework model to facilitate its design. He also suggested that game prototyping is important and helpful for better game design. Callele et al.[40] also investigated the importance of requirements engineering in the video game development process. They suggested that the reasons for the failure of any game may be rooted in problems of transforming the pre-production phase document, i.e., the GDD, with any implied information and application of domain knowledge from the pre-production phase into the production phase.

An understanding of upcoming media and technology developments, gameplay, and non-functional requirements is also considered important for the GDD. Callele et al.[41] described how the GDD is helpful in obtaining a better understanding of the game design process and explained the definition of gameplay process in cognitive game development. Reyno and Cubel[42] proposed a model-driven game development method that ultimately accelerates game design. Almeida and da Silva[43] performed a systematic review of game design methods and of various available tools. They empha-

sized the use of standardized tools to develop the GDD. Other researchers have emphasized the inclusion of the user perspective and have provided game design guidelines. Ahmed and Jaafar[44] emphasized the importance of user-centered game design and proposed that it should be considered at the concept phase of game development. Bringula et al.[45] gathered user perceptions to determine how a serious game should be developed. Based on their study, they suggested some design guidelines for four-dimensional game design, including storyline, aesthetics, reward systems, and the game objective.

To develop a good-quality game, the GDD must be properly managed so that production team members can easily move it into game production. GDD management has also been selected as an independent variable in this study, and therefore the following hypothesis and its corresponding null hypothesis are proposed:

Hypothesis 2 (H2). *Proper management of the game design document has a positive and significant effect on the overall game development process.*

Null Hypothesis. *Proper management of the game design document has no effect on the overall game development process.*

2.3 Game Engine Development

Game engines are considered to be a powerful tool by game developers and have been in use for more than two decades. A game engine is a software layer that helps in the development process by enabling develop-

ers to focus solely on game logic and experimentation[3]. Many commercial game engines are available to help game developers with advanced rendering technologies and code reuse, resulting in shorter development time and reduced cost. Sherrod[46] defined the game engine as a "framework comprised of a collection of different tools, utilities, and interfaces that hide the low-level details of the various tasks that make up a video game". Overall, the game engine represents the basic structure of the game as it appears in the middle layer, between the application layer and the various underlying platforms.

In the literature, most researchers often use the terms "game engine" and "game development framework" interchangeably. This study uses the term "game engine" to refer to the development tool that includes most of the functionality and features that become part of any software game. The list of primary features that can be part of any modern game engine includes scripting, rendering, animation, artificial intelligence, physics, audio, and networking. Cowan and Kapralos[47] performed a survey on frameworks and game engines for serious game development only. They compared all the commercially available game engines and their various features. The results of their study suggested that most of the game engines that have been developed to create entertainment games can also be used for serious game development. Hudlicka[48] suggested a set of requirements that are necessary for game engine development, specifically for affective games. Research has been also done on the development of game engines specific to different platforms, such as for the Android platform[49], a 3D role-playing game for cross-platform development[50], and the Browser games[51].

A few researchers have explored the means of addressing the challenges faced by developers in supporting and building development tools[52-53]. However, they were not successful in achieving the required feature and design flexibility. Researchers proposed different solutions for game engines to address the challenges they faced. Pimenta et al.[54] proposed that game engines enable fast learning for game developers and include the ubiquitous characteristics of the game design and development process. Neto et al.[55] discussed the issue of game engine standardization in software game development. Game developers are interested in producing the same game for different platforms and rely mostly on the same game engine. They suggested that commonality and variability assessment must be done

to enable game engine reuse. Peker and Can[56] proposed a methodology for developing game engines for mobile platforms based on design goals and design patterns. They emphasized the need to design goals and strategies for implementation of game engines. For mobile platforms, the basic design goals suggested by them were usability, efficiency, portability, and adaptability. To determine whether standard game engine development has a positive impact on the overall game development process, game engine development was considered as an independent variable in this study. Hence, the following hypothesis and its corresponding null hypothesis are proposed.

Hypothesis 3 (H3). *Game engine development has a significant impact on the game development process.*

Null Hypothesis. *Game engine development has no impact on the game development process.*

2.4 Game Asset Management

Anything can be considered as a game asset that contributes to the visual appearance of a game, such as artwork (including 3D elements or textures), music, sound effects, dialogue, text, or anything else. Llopis[57] stated that "game assets include everything that is not code: models, texture, materials, sound, animations, cinematics, scripts, etc." Actually, game assets include any piece of data that can be used by a game engine aside from code, scripts, and documentation. The elementary unit of game assets can be referred to as a game bit[58] and typically has no value when considered independently. There are two categories of bits: characters, which can be an asset that interacts in a simulated environment, and abstract bits, which are kinds of sound and texture that can be used together to produce a concrete bit. The main nine kinds of game bits are texture, sound, vegetation, buildings, fire, water, stone, clouds (concrete), and behaviour. Game space definition is another game asset, which is part of content generation for any game. It provides a kind of game environment where game bits can be placed.

In the literature covering game asset creation and management, researchers have explored game assets in term of animation, audio processing libraries for different genres, and content generation for games. De Carli et al.[59] and Hendrikx et al.[58] carried out a survey of procedural content generation techniques for game development. Animation in games is considered an important asset because it has a great impact on game performance[60]. Studies have been done to explore

animation models for different genres of games. Pranatio and Kosala[61] performed a comparative study of keyframes[62] and skeletal animations[66] for multi-player games. Their results indicated that skeletal or bone-based frames are better than keyframe models in term of memory load and frames per second. Xu and Wang[64] reviewed currently used 3D accelerators for graphics animation. A wide variety of graphics cards are available to programmers. Hence, they discussed the current benefits and limitations of APIs such as OpenGL and DirectX. Chehimi *et al.*[65] described the evolution of 3D graphics for mobile platforms. They concluded that the current market presents challenges regarding the graphics quality and battery life of mobile devices. These need to be addressed by standardizing successful game development for mobile platforms.

Sound within a game is one of the game assets that enable developers to build responsive, interactive, and attractive games. Currently, game development relies on pre-recorded sound clips that can be triggered during any game event[66]. These can be managed through dynamic audio processing libraries. Researchers have also studied the use of audio processing libraries in software game development. Pichlmair and Kayali[67] studied music games and determined that they can be classified into two categories, rhythm and instrument games. Their analysis showed that music in video games has seven qualities: rhythm, active score, quantization, synesthesia, play as performance, sound agents, and free-form play. Migneco *et al.*[68] proposed an audio processing library to enable the use of sound in Web-based games using a Flash development tool. They claimed that this approach provided flexibility and great functionality for developing games using Flash technology.

For the reasons discussed above, the creation and management of a number of assets required for game development has become challenging. Mechanisms are needed to control different versions of assets that are developed for games. Commercially, a number of tools are available, such as 3D Studio Max, Maya, and Adobe Photoshop, which can also create various assets like textures, 3D models, animations, sound effects, music, voice recordings, levels, and scenes. Modern game engines also include modules for asset management. Based on a literature review of game asset management, this study has considered game asset management as another independent variable that is considered important for the game development process. Hence, the following hypothesis and its corresponding null hypothesis are proposed.

Hypothesis 4 (H4). *Game asset management is important for enhancing the game development process.*

Null Hypothesis. *Game asset management is not important for enhancing the game development process.*

2.5 Quality of Game Architecture

The primary function of game architecture is to support gameplay. It helps to define challenges by using constraints, concealment, exploration, and obstacles or skill testing. Game architecture is a kind of blueprint for the underlying complex software modules. It is used to delineate design, perform trade-off analysis, and investigate system properties before implementation and potential reuse. Basically, it draws together gameplay factors and technical requirements. A perfect game architecture would have modularity, reusability, robustness, and tractability features.

The importance of software architecture in game development has rarely been researched. Only Wang *et al.*[69] have explored this topic. Their finding was that software architecture plays an important role in game development, with the focus mainly on achieving high performance and modifiability. They also stated that most developers use game-specific engines, middleware, and tools for game development. A number of studies have explored these various development frameworks. The proposed game development frameworks can help game developers to define their game architecture. Amendola *et al.*[70] proposed a framework for experimental game development called GLIESE. They proposed that a game architecture should have at least three sub-systems: a game logic processing system (view and model), a graphic processing system (graphic interface and view interface), and an input processing system (event manager, controller, and event publisher). These sub-systems must be clearly separated so that they can work independently. The authors suggested a model-view-controller (MVC)[71] pattern for the architecture. Basically, this pattern divides the application into three components: model, view, and controller. The defined relations and collaboration among these components help in game deployment because ultimately the code associated with each sub-system's logic will operate in the desired manner.

Lukashev *et al.*[72] proposed a mobile platform development framework specifically for 3D application. They claimed that their suggested approach will help developers improve the development process. The first

stage of the proposed framework is the design phase for the creation of the initial model (2D or 3D) and the selection of the right modeling tool and graphic format. The second stage, the integration stage, enables developers to put together already-created models into scenes and create animation. The authors of [72] suggested that a structural optimization technique can be used to create scenes. The third stage is the utilization stage, in which the created models are converted to mobile format. Implementation is the final step of the framework, where developers put together source code, auto-generated source code, and created resources. Several other studies have also been performed to propose development frameworks for various platforms based on different technologies for defining the system architecture. For example, Rhalibi *et al.*[73] proposed a 3D Java framework for Web-based games, Jhingut *et al.*[74] and Kosmopoulos *et al.*[75] proposed a framework for mobile platforms, Al-Azawi *et al.*[76] proposed an agent-based agile methodology for game development, and Segundo *et al.*[77] proposed a game development framework specific to the Ginga middleware.

From the preceding discussion, it is clear that the quality of the game architecture is important for the game development process, and therefore it was considered as another independent variable in this study. The following hypothesis and its corresponding null hypothesis are therefore presented.

Hypothesis 5 (H5). *Quality of game architecture has a positive impact on the enhanced game development process.*

Null Hypothesis. *Quality of game architecture has no impact on the enhanced game development process.*

2.6 Game Test Management

Game testing is a very important phase of game development. A game can be tested at different levels of development because game testing is different from software testing[78]. There are many steps involved in game testing other than test-case definition because most game testing is based on black-box testing. Hence, the management of overall game test methods becomes crucial. In the pre-production phase, a test plan document should be established to set standards for the game software. Game quality can be evaluated according to the graphics, sounds, and code that are compiled into the game code. Proper documentation of testing

helps developers fix problems more quickly and cheaply. Delays in testing can result in project failure.

Helppi[79] discussed many game test methods that can be used during the development phase, such as smoke testing that is used to test the user interface logic. Regression testing is performed to check that game quality is still good after any change such as the addition of features or add-ons. Connectivity testing is used for networking games and mobile games to test client-server interaction. Performance testing can ensure the real performance of the game. Abuse testing is performed by giving multiple inherent inputs through the controller and determining game performance. Compliance testing makes sure that any compliance standards enforced by any stakeholder are met. Finally, functional testing verifies overall gameplay and reveals issues related to stability, game flow, game mechanics, integration of graphic assets, and user interface. Redavid and Farid[78] also discussed game testing methods used to detect interactions failures and listed them under the term combinatorial testing[79]. The second approach involves test flow diagrams, which are used to develop models of game behaviour from a player's perspective. The third is cleanroom testing, which helps to determine game reliability. The test tree is another testing method discussed by the authors of [78], which can used to organize test cases.

Wilson[81] also argued that no one testing method is better than another. He suggested that good testing is a combination of 30% of ad-hoc testing, 40% test cases, and 30% alternating between the two until the strengths of both are determined. Marri and Sundaresaubramanian[82] discussed game test methods and suggested that the game tester should test game quality by verifying gameplay, logical consistency, observability, progressive thinking, and reasoning ability, as well as exhaustively testing features, game strategy, and functionality. Kasurinen and Smolander[83] interviewed seven game development teams from different organizations and studied how they test their games based on grounded theory. They concluded that all participating organizations had the resources to perform technical testing, but that they relied mostly on exploratory and usability testing rather than on using a pre-planned approach. Al-Azawi *et al.*[84] proposed a set of evaluation heuristics that could be used in game development methodologies for most game genres. Omar and Jaafar[85] proposed a tool to evaluate the usability of educational games, and Straat and Warpefelt[86] suggested the use of the two-factor theory

to evaluate game usability.

Management of game testing during the game development process has clearly come to be of crucial importance for game developers. Hence, test management was selected as another independent variable in this study, and the following hypothesis and its corresponding null hypothesis are proposed.

Hypothesis 6 (H6). *Game test management has a positive impact on the enhanced game development process.*

Null Hypothesis. *Game test management has no impact on the enhanced game development process.*

2.7 Programming Practices

Good programming practices are a very important factor in successful game development. A programming team with the necessary skills is definitely considered as the backbone of the game development process. The programmer must select it the right coding architecture for each game project. Basically, the lead programmer must select it between two types of coding style: either game-specific code (the programmer has to develop everything by him/herself) or game-engine code (where the game engine is the foundation for a game-specific code). The game code can then be organized in various ways[3], such as an ad-hoc architecture where the programmer must deal with tightly coupled code. Another choice is a modular architecture based coding style, where the programmer identifies and separates the code into different modules or libraries. In this type of programming, reuse and maintainability are improved over ad-hoc-based coding. However, dependencies between different modules cannot be controlled, which may lead to tight coupling. The directed acyclic graph (DAG) is another way of organizing code. This is also a modular architecture based coding scheme in which dependencies between modules are tightly controlled. Layered-style coding is also based on a DAG architecture, but modules are arranged in rigid layers, and each can interact only with the modules in the layer directly below.

Game programming involves a wide range of issues and considerations. Most researchers have tried to address them individually. The first is the issue of coupling between different modules. Sarinho and Apolinario[87] tried to address this problem using a proposed generative programming approach. Generative programming aims to automate the software development process using a number of static and dynamic technologies including reflection, meta-programming, and program and model analysis[88]. The proposed method was based on a game feature model that could represent both common and variable implementation aspects of software games. Meta-programming resources were used to generate and represent compatible source code for available game frameworks and game engines. The authors of [87] concluded that the proposed approach would result in the loss of the coupling development strategy between game implementation and its domain software artifacts. Code cloning in open-source games is another issue discussed by Chen *et al.*[89] They provided a detailed study of the issues of code clones in more than 20 open-source game projects based on C, Python, and Java for various game genres. Selection of a scripting language is another issue in game programming. Anderson[90] discussed the classification of scripting systems used for software games. Xu and Rajlich[91] described a study that explored pair programming practices and concluded that paired programmers completed their task faster with higher quality. They suggested that pair programming is a good approach for game development.

Selection of a programming language is another challenge for today's game developers. Many studies have been done to explore different programming languages for different platforms. Zhang *et al.*[92] performed experiments on five industrial RPG mobile games developed using the object-oriented programming paradigm. Optimization strategies with structural programming were applied to the same code. The results of the study showed that object-oriented programming must be used with great care and that structural programming is also a good option for mobile game development. Another study[93] highlighted the issues for game development posed by wireless peer-to peer games in a J2ME environment using an available Bluetooth API. The issues discussed include slow device discovery, Bluetooth transfer speed, extra resource consumption, and Bluetooth topology. Meng *et al.*[94] developed a peer-to-peer online multiplayer game using DirectX and C# to achieve playability in a .Net environment.

According to the above discussion, programming practices were selected as an independent variable in this study, and the following hypothesis and its corresponding null hypothesis are proposed:

Hypothesis 7 (H7). *Good programming practices are important for the enhanced game development process.*

Null Hypothesis. *Good programming practices are not considered important for the enhanced game development process.*

3   Research Model

The main objective of the proposed research model is to analyze the interrelationship between key factors and game development and also to understand the influence of these factors on overall game quality in the SGI market. The model's theoretical foundation is based on existing concepts found in the game development literature. Note that most studies in the literature discuss one or two of the factors mentioned above for software games and their impact on the overall game development process. To the best of our knowledge, this is the first study in the game development literature that highlights key factors in game development. This study proposes to investigate empirically the influence and association of key game development factors. Fig.1 presents a theoretical research model used in this study, which will be empirically investigated. The theoretical model evaluates the relationships of various independent variables emerging from software engineering and management concepts such as project management, theory, and behaviour with the dependent variable, enhanced game development, in the context of the game development process. This study mainly investigates and addresses the following research question.

Research Question: how can game developers improve the game development process?

The research model includes seven independent variables: team configuration and management, game design document management, game engine development, game asset management, quality of game architecture, game test management, and programming practices, and one dependent variable: the enhanced game development process.

The multiple linear regression equation of the model is given as (1):

$$\text{Enhanced game development process}$$
$$= \alpha_0 + \alpha_1 f_1 + \alpha_2 f_2 + \alpha_3 f_3 + \alpha_4 f_4 + \alpha_5 f_5 +$$
$$\alpha_6 f_6 + \alpha_7 f_7, \qquad (1)$$

where α_0, α_1, α_2, α_3, α_4, α_5, α_6, α_7 are coefficients and $f_1 \sim f_7$ are the seven independent variables.

4   Research Methodology

Developing a software game involves phases such as pre-production, production and post-production, in which each phase contains a number of activities. Some of these activities are dependent on others, whereas some are independent. Employees of game development organizations or studios were selected as the targeted respondents of this study. In this study, the term "developer" is used to refer to any game development team member. For purposes of data collection, the authors initially joined various game development community forums. The respondents participating in the study were part of multinational organizations in Asia, Europe, and North America; statistics describing them are presented in Fig.2.
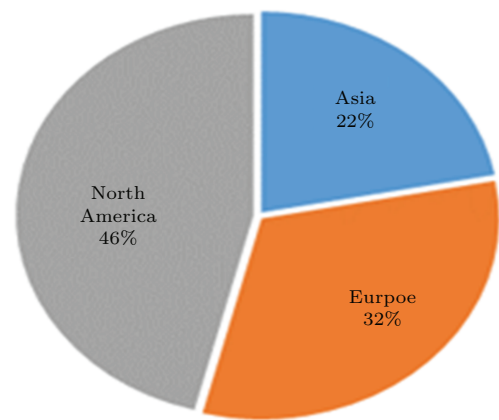

Fig.2.   Number of respondents by continent.

The organizational participants agreed to take part in the study based on a mutual agreement that their identities would be kept confidential. The size of the game project development teams varied from 10 to 50. Fig.3 shows the total time period of the game development projects considered by respondents while answering the measuring instrument. Fig.4 represents the percentage of respondents based on their development role in the game project. Fig.5 shows the percentage of development methodologies used by respondents for any particular game project.

The participants in the study were mainly part of game projects that were developed for different platforms such as kiosks and standalone devices, the Web, social networks, consoles, PC/Macs, and mobile phones. The game genres implemented in most of their projects included action or adventure, racing, puzzles, strategy/role playing, sports, music-based, and other categories. The qualifications for this study were that the respondent must be a part of a development team that had at least three full-time developers, that the

936

J. Comput. Sci. & Technol., Sept. 2016, Vol.31, No.5

respondent worked on a project for at least one-third of its total duration, and that the project was either completed or cancelled within the last three years.
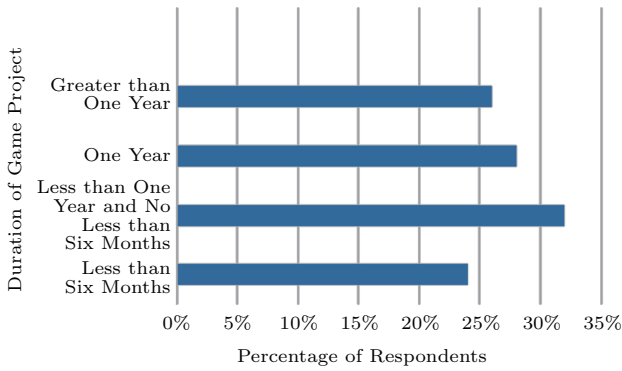

Fig.3. Total software game development duration of particular game projects considered by respondents.
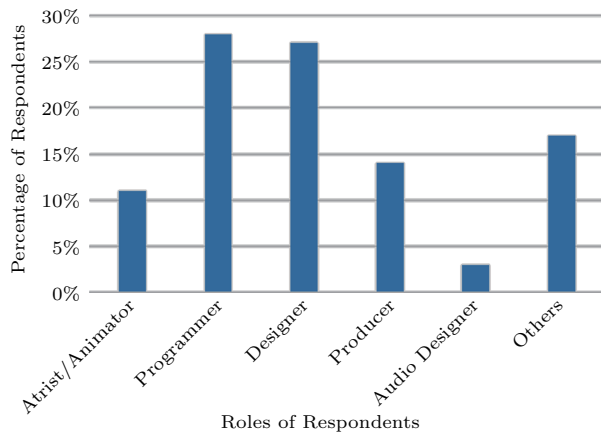

Fig.4. Percentage of respondents based on their roles in the development process.
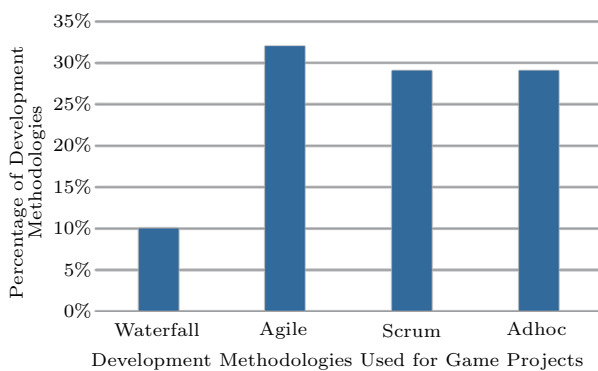

Fig.5. Percentage of development methodologies used by respondents.

Finally, respondents must have worked in the development team in some sort of development role, such as a

designer, artist, animator, programmer, producer, and sound designer. The survey respondents worked in various capacities such as game designer, artist, programmer, audio designer, and producer. The total number of survey respondents was 118, including a minimum of one and a maximum of four responses from each organization. Although the collected sample size is relatively large, it is still considered as a small sample as compared with the population size.

4.1 Measuring Instrument

This study gathered data on the key developer's factors and the perceived level of enhanced game development process identified in the research model depicted in Fig.1. To learn about these two topics, the questionnaire presented in Appendix was used as a data collection instrument. First, organizations involved in the game development process were asked to what extent they practiced the identified key developer's factors for the game development project in questionnaire. Second, they were asked what they thought of the enhanced game development process for different games in the software game industry. The five-point Likert scale was used in the questionnaire, and with each statement, the respondents were required to specify their level of agreement or disagreement. Thirty-four items were used to measure the independent variables (the key factors), and for the dependent variable (enhanced game development process), nine items were used. The literature related to key developer's factors was reviewed in detail to ensure a comprehensive list of measurement items for each factor from the literature. A multi-item, five-point Likert scale was used to measure the extent to which each key developer's factor was practiced for the game development project. The Likert scale ranged from 1 meaning "strongly disagree" to 5 meaning "strongly agree" and was associated with each item. The items for each identified factor were numbered from 1 to 34 in Appendix and also labelled sequentially. They were measured for each project that was completed within the last three years based on a multi-item five-point Likert scale. The enhanced game development process was the dependent variable, and designated items for the dependent variable were numbered separately from 1 to 9 and labelled sequentially. All the items specifically written for this study are presented in Appendix. To the best of our knowledge, this is the first empirical study of key software game developer's factors for the enhanced game development process in the SGI.

4.2 Reliability and Validity Analysis

To perform reliable and valid research, quantitative analysis was carried out. Two integral measure of precision, reliability and validity analysis, were used to conduct empirical studies. The consistency or reproducibility of a measurement is referred to as reliability. On the other hand, the valid inference or agreement between the measured and the true values is referred to as validity. The measuring instrument designed for this study was also tested by reliability and validity analysis. The test was based on common practices usually used for empirical analysis. Reliability analysis was performed to determine the internal consistency of the multi-scale measurement items designed for the seven identified factors. To evaluate internal consistency, Cronbach's alpha[95] coefficient was used. Criteria for Cronbach's alpha ranging from 0.55 to 0.70 were considered satisfactory. Researchers have reported different ranges of satisfactory criteria for Cronbach's alpha based on their findings. Osterhof[96] suggested that a value of 0.60 or higher was satisfactory for reliability coefficients based on his findings. Nunnally and Brenste[97] reported that a value of 0.70 or higher for a reliability coefficient can be considered satisfactory for any measuring instrument.

Van de Ven and Ferry[98] recommended that a value of 0.55 or higher of the reliability coefficient could be considered satisfactory. A first calculation was performed on a sample dataset to determine the reliability of the dataset using Cronbach's alpha coefficient. Some of the assessment items for each factor were excluded if they affected the desired value of Cronbach's alpha coefficient. In the sample dataset, other than item No. 1 of team configuration & management, item No. 10 of game design document management, item No. 18 of game engine development, item No. 22 of game asset management and item No. 30 of programming practices, all assessment items were found reliable. Thus, we removed item No. 1, item No. 10, item No. 18, item No. 22 and item No. 30 from the instrument. After this, the whole dataset was evaluated using Cronbach's alpha coefficient. The results of these calculations showed that reliability coefficients for the seven factors ranged from 0.61 to 0.76. These coefficients are reported in Table 2. Hence, all variables developed for this study could be considered reliable.

Validity analysis was performed for the dataset using principal component analysis (PCA)[99]. PCA is usually used for convergent validity analysis and was calculated here for seven factors. Campbell and Fiske[100] suggested that convergent validity has occurred in a given case only if the scale items in a measurement instrument are highly correlated and if they move in the same direction in a given assembly. The construct validity of PCA-based analysis was determined using the eigenvalue criterion[101]. Here, a criterion value greater than 1 was used to retain any component based on the Kaiser criterion[102]. Eigenvalue analysis showed that out of the seven variables, five together formed a single factor, whereas game design document management and programming practices loaded on a second factor, and both eigenvalues were greater than 1. The reported convergent validity of this study was considered adequate.

4.3 Data Analysis Techniques

To perform the empirical investigation for this study, various statistical approaches were used. Initially, the research activity was divided into three phases to evaluate the significance of the proposed hypotheses H1~H7. In phase 1, parametric statistical and normal distribution tests were performed. A non-parametric statistical approach was used in phase 2, and for the analysis, a partial least squares (PLS) analysis was carried out.

To address external threats to validity, both parametric and non-parametric approaches were used.

Table 2. Cronbach's Alpha Coefficient and Principal Component Analysis of Seven Variables

Developer's Factor	Item No.	Coefficient α	PC Eigenvalue
Team configuration & management	1~6 (excluded 1)	0.63	1.48
Game design document management	7~11 (excluded 10)	0.60	1.51
Game engine development	12~18 (excluded 18)	0.68	1.49
Game asset management	19~22 (excluded 22)	0.81	1.57
Quality of game architecture	23~25	0.84	1.01
Game test management	26~29	0.64	1.79
Programming practices	30~34 (excluded 30)	0.86	1.25

Parametric approach is used to measure the strength of the linear relationship between normally distributed variables. When the relationship between the variables is not linear or the variables are not normally distributed, then it may be more appropriate to use the non-parametric approach. Due to the small sample size, both parametric and nonparametric approaches were used to address the threats to external validity and we found results of both approaches are consistent.

The measuring instrument contains multiple items for each independent and dependent variable, and respondent ratings were aggregated to obtain a composite value. Using a parametric statistical approach in phase 1, the Pearson correlation coefficient was calculated for the tests, with a one-tailed t-test for each hypothesis of H1∼H7. For phase 2, the Spearman correlation coefficient was used to test hypotheses H1∼H7 using a non-parametric statistical approach. Phase 3 of the empirical investigation was carried out to address issues of non-normal distribution and complexity or small sample size of the dataset. Fornell and Bookstein[103] and Joreskog and Wold[104] reported that if non-normal distribution, complexity, small sample size, and low theoretical information are issues, then partial least squares (PLS) analysis will be helpful.

The PLS technique was used in Phase 3 to increase the reliability of the results and deal with the limitation of small sample size. For statistical calculations, the Minitab 17 software was used.

5   Data Analysis and Results

5.1   Phase 1 of Hypothesis Testing

To test hypotheses H1∼H7, parametric statistics were used in this phase. The Pearson correlation coefficient was determined between the independent variables (developer's factors) and the dependent variable (the enhanced game development process) of the research model, as illustrated in Fig.1. The level of significance to accept or reject the hypotheses was then selected. Each hypothesis was accepted if its p-value was less than 0.05 and rejected if its p-value[105] was greater than 0.05. In Table 3, calculated results for the Pearson correlation coefficient are listed.

Hypothesis H1 was accepted because the Pearson correlation coefficient between team configuration and management and the enhanced game development process was positive (0.29) at $p < 0.05$. For hypothesis H2 concerning game design document management

and the enhanced game development process, the Pearson correlation coefficient was also positive (0.79) at $p < 0.05$, and therefore hypothesis H2 was also accepted. Hypothesis H3 concerning game engine development and the enhanced game development process was accepted due to a positive (0.59) correlation coefficient at $p < 0.05$. Hypothesis H4 concerning game asset management and the enhanced game development process was accepted based on its positive Pearson correlation coefficient (0.45) at $p < 0.05$. Hypothesis H5 concerning quality of game architecture and the enhanced game development process was rejected based on its positive correlation coefficient (0.13), but $p > 0.05$. Hypothesis H6 regarding game test management and the enhanced game development process was accepted due to its positive Pearson correlation coefficient (0.42) at $p < 0.05$. The last hypothesis (H7) relating programming practices to the enhanced game development process was also found to be significant (0.52) at $p < 0.05$ and was therefore accepted. Hence, in summary, hypotheses H1, H2, H3, H4, H6, and H7 were accepted and found to be statistically significant. Hypothesis H5 was not supported statistically and was therefore rejected.

5.2   Phase 2 of Hypothesis Testing

Hypotheses H1∼H7 were tested based on the non-parametric Spearman correlation coefficient in phase 2. Table 3 reports the results for the Spearman correlation coefficient. Hypothesis H1 regarding team configuration and management was accepted because of its positive Spearman correlation coefficient (0.29) at $p < 0.05$.

The Spearman correlation coefficient for game design document management and the enhanced game development process (hypothesis H2) was also positive (0.74) at $p < 0.05$ and was also found to be significant. The relationship between game engine development and the enhanced game development process game (hypothesis H3) was found to be statistically significant due to its Spearman correlation coefficient (0.64) at $p < 0.05$ and was accepted. For hypothesis H4 regarding game asset management and the enhanced game engine development, the Spearman correlation coefficient was positive at $p < 0.05$, and therefore H4 was accepted. Hypothesis H5 concerning quality of game architecture and the enhanced game development process was rejected due to its positive coefficient (0.19) at $p > 0.05$. Hypothesis H6 concerning game test management and the enhanced game development process

was accepted due to its positive Spearman correlation coefficient (0.37) at $p < 0.05$. The last hypothesis (H7) relating programming practices to the enhanced game development process was also found to be significant (0.48) at $p < 0.05$. In summary, hypotheses H1, H2, H3, H4, H6, and H7 were accepted and found to be statistically significant. Hypothesis H5 was not supported statistically and was therefore rejected.

5.3 Phase 3 of Hypothesis Testing

Hypothesis testing in phase 3 was performed using the partial least squares (PLS) technique. The main reason for using the PLS method in this phase was to cross-validate the results obtained from the parametric and non-parametric statistical approaches used in Phases 1 and 2 and to overcome their associated limitations.

Tests were also performed on hypotheses H1~H7 to check their direction and significance. The dependent variable, i.e., the enhanced game development process, was designated as the response variable and other individual factors (independent variables) as the predicate variables for PLS examination. The observed results of the structural hypothesis tests are presented in Table 4. The table also includes the values of the path

coefficient, R^2, and the F-ratio. The path coefficient for team configuration and management (H1) was observed to be 0.29, with an R^2 of 0.08 and an F-ratio of 11.35, and H1 was therefore found to be significant at $p < 0.05$. Game design document management (H2) had a positive path coefficient of 0.74, $R^2 = 0.56$, and F-ratio = 148.9 and was also found to be statistically significant at $p < 0.05$. Game engine development (H3) had a path coefficient of 0.59, a very low R^2 of 0.34, and an F-ratio of 62.09 and was found to be significant at $p < 0.05$. Game asset management (H4) had a positive path coefficient of 0.07, a low R^2 of 0.06, and an F-ratio of 0.72 and was judged to be significant because the p-value was less than 0.05. Quality of game architecture (H5) (path coefficient: 0.13, R^2: 0.02, and F-ratio: 2.3) was found to be statistically insignificant at $p > 0.05$. Game test management (H6) (path coefficient: 0.42, R^2: 0.18, and F-ratio: 26.20) and programming practices (H7) (path coefficient: 0.52, R^2: 0.27, and F-ratio: 44.45) were found to be significant at $p < 0.05$.

5.4 Research Model Testing

The linear regression equation for the research model is given by (1). The research model was tested to provide empirical evidence that factors important to

Table 3. Hypothesis Testing Using Parametric and Non-Parametric Correlation Coefficients

Hypothesis	Key Factor	Pearson Correlation Coefficient	Spearman Correlation Coefficient
H1	Team configuration and management	0.29*	0.29*
H2	Game design document management	0.79*	0.74*
H3	Game engine development	0.59*	0.64*
H4	Game asset management	0.45*	0.47*
H5	Quality of game architecture	0.13**	0.19**
H6	Game test management	0.42*	0.37*
H7	Programming practices	0.52*	0.48*

Note: * means significant at $p < 0.05$, and ** means insignificant at $p > 0.05$.

Table 4. PLS Regression Results for Hypothesis Testing

Hypothesis	Factor	Path Coefficient	R^2	F-Ratio
H1	Team configuration and management	0.29	0.080	11.35*
H2	Game design document management	0.74	0.560	148.90*
H3	Game engine development	0.59	0.340	62.09*
H4	Game asset management	0.07	0.006	0.72*
H5	Quality of game architecture	0.13	0.020	2.30**
H6	Game test management	0.42	0.180	26.20*
H7	Programming practices	0.52	0.270	44.45*

Note: * means significant at $p < 0.05$, and ** means insignificant at $p > 0.05$.

game developers play a considerable role in improving the overall game development process in the SGI. The test procedure examined the regression analysis, the model coefficient values, and the direction of the associations. The dependent variable (the enhanced game development process) was designated as the response variable and the other independent variables (all the other key developer's factors) as predicate variables. The regression analysis model results are reported in Table 5. The path coefficients of six of the seven variables (team configuration and management, game design document management, development of a game engine, game asset management, game test management, and programming practices) were positive and were found to be statistically significant at $p < 0.05$. The path coefficient for quality of game architecture was positive, but was found not to be statistically significant at $p < 0.05$. The overall R^2 value of the research model was 0.83, and the adjusted R^2 value was 0.68 with an F-ratio of 36.97, which was significant at $p < 0.05$.

6 Discussion

Software game development is a multidisciplinary activity that has its roots in the management and software engineering disciplines. The software game industry has become a mass phenomenon, supplemented by a number of possible strategies and exciting questions for game development companies. More and more companies are entering the market, and hence the intensity of competition is increasing. Established and new entrants must both pay attention to the key factors that help to improve their game development processes and keep them competitive in the market. Now it is time to

understand the perspective of game developers and to learn what they think is important to improve software game quality and how the developed game can become successful in the market. This research is a first step towards this understanding because it will help developers and game development organizations to understand the relationships and interdependences between key factors from a developer's perspective and to understand the enhanced game development process. This research is the first empirical investigation of factors important to developers in relation to improving the current development process and provides an opportunity to explore associations between them empirically. The observed results support the theoretical assertions made here and provide the very first evidence that the consideration of key developers' factors while developing games is important for software game success. This could well result in institutionalizing the software game development approach, which in turn has a high potential to maximize profits.

Especially in the game development process, multidisciplinary team configuration and management is a huge challenge. Basically, producing high-quality games relies on a high level of planning, communication, and organization of multidisciplinary teams to avoid costly delays and failures. Many factors have been identified by researchers as being important to implement a successful collaboration between any kinds of multidisciplinary team. These factors include interpersonal factors such as trust among team members and ability to communicate[34], willingness to collaborate, and mutual respect[106]. Others are organizational factors, including establishing appropriate protocols and supporting collaboration[107]. These factors can be implemented by using various software applications that

Table 5. Linear Regression Analysis of the Research Model

Model Coefficient Name	Model Coefficient	Coefficient Value	t-Value
Team configuration and management	α_1	0.06	1.14*
Game design document management	α_2	0.50	7.44*
Game engine development	α_3	0.31	5.19*
Game asset management	α_4	0.21	0.38*
Quality of game architecture	α_5	0.03	6.57**
Game test management	α_6	0.13	2.24*
Programming practices	α_7	0.10	1.58*
Constant	α_0	0.01	1.13*
R^2	0.83	Adjusted R^2	0.68
F-ratio	36.97*		

Note: * means significant at $p < 0.05$, and ** means insignificant at $p > 0.05$.

are specifically designed for collaborating on commercial software development projects. The main concern when using these software applications is that they must fit in with the existing computing and workflow environment[108]. Management of the members of various multidisciplinary teams can be evaluated and maintained mainly by examining values and practices, for example, what each individual team member brings to the table, how they use material or assets produced by other team members, how they reconcile conflicting priorities, and finally how their personal relations influence the collaboration. The multidisciplinary team can use management or collaboration software for task tracking, version control, file sharing, and continuous integration. Successful collaboration between team members enables them to manage easily all phases of game development from start-up, creating a concept, creating a proof of concept, the production phase, and so forth until the game is published.

This study has explored the importance of team configuration and management factors from a developer's perspective. It has found positive associations between team configuration and management and the enhanced game development process. Hence, proper configuration and management of multidisciplinary teams is a crucial part of the game development process. However, it must be balanced with other development issues in the game development process.

Game design document management has been found to be positively associated with the enhanced game development process. The GDD is mainly a pre-production artifact which is defined by the pre-production phase team to capture a creative vision of the game. Game developers generally feel that imposing too much structure at the start of a game may be highly detrimental[40], resulting in reduced creativity, constraining expression, and risking the intangibles that create an enjoyable feeling or experience. At the same time, the importance of structure has been highlighted by many researchers, as discussed in the literature review section (Section 2). Management of the game design document and its transition into a requirements and specifications document is challenging.

One way to handle this during the pre-production phase is to produce two documents. The first one is the GDD, and the second one is a document of requirements and specifications based on the GDD. Managing and transforming the GDD into a production document is complex because the two require different documentation styles. Supportive documentation is also required to help the development team in its transition from pre-production to the production phase. The author of the GDD may not have the requisite writing skills to produce a document that is understandable by the production team (technical people). Basically, there is a long list of required skills for a GDD developer, such as knowledge of game design, technical communication, and requirements engineering. Hence, a formal process is needed to support the transition and would likely increase the reliability of the game development process. The results of this study have shown that the development and transformation of the GDD is very important and also requires strong management skills to reduce documentation effort. Hence, the results presented here have shown that a good GDD is the greatest contributor to the success of a game development project.

Game asset management was also found to have a positive association with the enhanced game development process. Game assets, defined as any piece of data that is in a format that can be used by the game engine, will be presented to the user. To create and manage game assets, a realistic content generator must be developed that can fill in the missing bits. Trade-offs between realism and performance and between realism and control must also be investigated for any asset created. For graphical animation, a number of 3D model formats can be used by game developers. These can generally be divided into two categories: frame-based animation and skeletal-based animation. Determination of the perfect animation model for a game has become crucial because a diversity of format types for graphics are available. Eventually, a poor choice could limit the performance of the game itself. For sound effects in games, certain problems are faced by developers because of unexpected or complex scene configurations. A number of asset management tools exist, but selecting the appropriate one is a challenge because each has its own limitations and benefits.

Improvements in the game development process have been greatly aided by the emergence of game development tools, specifically game engines. A game engine facilitates the game development process by providing various sets of features that help decrease development time and cost. These are available for most game genres (e.g., role-playing games or serious games for training) and vary in cost and complexity. Not all game engines support the entire feature set of all the game genres. Hence, integrating all the technological aspects into one framework is a prohibitively difficult

task. It is understandable, therefore, that confusion exists among game developers with regard to selecting the appropriate game engine. Game development tools should be selected only after determining the game concept and the GDD[109]. Most researchers in the area of game development tools have proposed their own architectures for specific genres and platforms. Anderson *et al.*[110] raised some important open questions for the academic community that are specific to the game engine development research field. The first is the main issue of the lack of a development language. The second question is how to define the boundaries between the game loop and the game engine. For example, what technical aspects should a game engine cover in a game? The third problem is that there is no standardization for game engines because most of them are specific to a particular game genre and game project. The fourth issue involves design dependencies, and the last the need for best practices when creating game engines. It was generally agreed that a game engine should handle a diversity of inputs and outputs, a restricted set of customizations based on each genre, and an asset and resource management system. The results of this study have also showed that development of a game engine has a positive impact on the enhanced game development process. In other words, game engine development is an important factor that needs more consideration from a developer's perspective.

It is a common perception that a good-quality or even perfect game architecture is a very important part of the game development process because reworking architecture afterwards is always hard. A game architecture identifies the main structural components of the underlying software and their relationships. In the game development literature, many researchers have proposed different frameworks for different platforms and based on different technologies. As a developer it is difficult to select among these because all provide a kind of reference architecture and their validity is still in question. The findings of the study do not support a statistically positive relation between the quality of game architecture and the enhanced game development process. The direction of association was found to be positive, but the required level of confidence was not supported. Hence, the hypothesis that the quality of game architecture has a positive impact on the enhanced game development process was statistically rejected.

Testing in game development is done mainly at a very late stage or the end of development to ensure the quality and functionality of the finished product. Typically, in a particular game project, the leader dedicates a specific amount of time for quality assurance or a beta tester to test the game. Various development methodologies are used to develop games, such as the agile methodology and the waterfall model, but testing must form part of all processes. Every aspect of a game should be tested during the development and production phases. In addition, certain foundational elements should also be tested during the pre-production phase, such as frameworks and platform set-up. The most important aspect of testing for game developers is to integrate testing as part of the production phase to improve efficiency. To ensure continuous quality and delivery of good games to the market, developers must consider majority testing options during the production phase. Helppi[79] also researched the possibility that mobile game robustness can be improved by continuous integration, delivery, and testing and concluded that this approach can improve the outcome of games and results as a more robust end-product. Therefore, testing plays an important role in each step of the development phase, and its management throughout the game development process is important. The results of this study have also supported the hypothesis that game testing management is important for the enhanced game development process. At the same time, testing techniques have matured over time, but still need improvement.

Game programming strategy has a direct effect on game performance. There are many concerns associated with today's game programming practices. Game developers must look for solutions to common problems in game programming such as coupling of modules, availability of different scripting and programming languages, platform compatibility issues, memory management, and code optimization strategy, specifically to improve game performance and quality. Hence, game developers must consider various aspects of the game such as speed, flexibility, portability, and maintainability while still coding. Ultimately, the skilled programming team will be able to develop and implement the full functional game. Matching of required skills to the abilities of developers is very important to improve the overall game development process, which is a conclusion also supported by this study.

Overall, the findings of this study are important for the development of good-quality software games. Rapid and continual changes in technology and intense competition not only affect the business, but also have a great impact on development activities. To deal with

this strong competition and high pressure, game development organizations must continually assess their activities and adopt an appropriate evaluation methodology. The use of a proper assessment methodology will help the organization identify its strengths and weaknesses and provide guidance for improvement. However, the fragmented nature of the game development process requires a comprehensive evaluation strategy, which has not yet been entirely explored. The findings of this study will help game development organizations to look for contributing key success factors from a developer's perspective. This study is a part of a large project aiming to propose a software game maturity assessment model. The developer's perspective was one of the important dimensions identified among the consumers, the business[111], and the process itself. The findings of this study also provide a justification to include these factors in the process assessment methodology.

6.1 Limitations of the Study

For software engineering processes or product investigations, various empirical approaches are used, such as case studies, metrics, surveys, and experiments. However, certain limitations are associated with empirical studies and with this study as well. Easterbrooks *et al.*[112] suggested four criteria for validating empirical studies: internal validity, construct validity, external validity, and reliability. Wohlin *et al.*[113] stated that generalizing experimental results to industrial practice by researchers is mostly limited by threats to external validity. In this study, measures were taken to address external threats to validity. The random sampling method was used to select respondents from all around the world. Open-ended questions were also included in the questionnaire.

The choice and selection of independent variables was one of the limitations of this study. To analyze the association and impact of factors affecting software game success, seven independent variables were included. However, other key factors may exist and have a positive association with and impact on the game development process, but due to the presence of the selected seven variables in the literature, they were excluded in the study. In addition, other key factors may exist, such as regionally or environmentally based choices, which may have a positive impact on the game development process, but were not considered in this study. Furthermore, the focus of this study was only

on developers' factors affecting the enhanced game development process.

Another notable limitation of the study is the small sample size. Although the collected number of responses is large in number, they can still be considered small compared with overall population size. The vast majority of game developers worked in one- to three-person teams, did not have the required level of experience (three years), and were therefore excluded from this empirical investigation. Most respondents refused to answer the questionnaire because they were too busy in the game development process or launching their games in the market. Therefore, data collection from the game industry was limited, resulting in small sample size. There are some approaches discussed by researchers such as by Zhang and Zhang[117] to handle the small sample size by using different machine learning techniques. However, one of the objectives to divide data analysis section into three phases is to address the small sample size issue. The main effect of small sample size is on its statistical power, Type II error, significance and on distribution[115]. Therefore, the important thing is while making conclusion avoid strong statements. The small samples size studies results can be difficult to replicate or generalize[116] but they do provide some interplay between variables. The well designed small studies seem to be ok to conduct as they provide quick results, but they need to be interpreted carefully[114]. The results of small studies should be used to design large confirmatory studies which is the case of this study as well.

In software engineering, the increased popularity of empirical methodologies has raised concerns about ethics. This study has adhered to all applicable ethical principles to ensure that it would not violate any experimental ethics guidelines. Regardless of its limitations, this study has contributed to the software game development process and has helped game development organizations understand the developer's dimension of software games.

7 Conclusions

Game development is a complex process, and for successful development of good-quality software games, game developers must consider and explore all related dimensions as well as discussing them with all the stakeholders involved. This study provided a better understanding of the factors important to developers in the software game development process and explored the

impact of key factors on the success of software games from a developer's perspective. This study mainly tried to answer the research question that was posed earlier in this paper and to analyze the impact of developers' key factors for game development process improvement. The results of this empirical investigation demonstrated that developers' key factors are very important and play a key role in improving the software game development process. The results showed that team configuration and management, game design document management, game engine development, game test management, and programming practices are positively associated with the enhanced game development process. The empirical investigation found no strong association or impact between the quality of game architecture and the enhanced game development process. In the game development field, this research is the first of its kind and will help game developers and game development organizations achieve a better understanding of key factors for improving the game development process. To improve the current game development process and develop good-quality games, it is important for developers to consider the identified key factors as well as others. Currently, the authors are working on developing a software game maturity model for game development process assessment. This study has provided the empirical evidence and justification to include factors from the developers' perspective in evaluating the developer's dimension of game development process maturity.

References

[1] Petrillo F, Pimenta M. Is agility out there? Agile practices in game development. In *Proc. the 28th ACM International Conference on Design of Communication*, September 2010, pp.9-15.

[2] Delmestri G, Montanari F, Usai A. Reputation and strength of ties in predicting commercial success and artistic merit of independents in the Italian feature film industry. *Journal of Management Studies*, 2005, 42(5): 975-1002.

[3] Rabin S. Introduction to Game Development (2nd edition). Cengage Learning, 2010.

[4] Blow J. Game development: Harder than you think. *Queue*, 2004, 1(10): 28-37.

[5] Kanode M C, Haddad H M. Software engineering challenges in game development. In *Proc. the 6th International Conference on Information Technology: New Generations*, April 2009, pp.260-265.

[6] Kultima A. The organic nature of game ideation: Game ideas arise from solitude and mature by bouncing. In *Proc. the International Academic Conference on the Future of Game Design and Technology*, May 2010, pp.33-39.

[7] McGill M M. Defining the expectation gap: A comparison of industry needs and existing game development curriculum. In *Proc. the 4th International Conference on Foundations of Digital Games*, April 2009, pp.129-136.

[8] Harris P R, Gredler M. Designing and evaluating games and simulations: A process approach. *Behavioral Science*, 1995 40(1): 76-77.

[9] Gredler M E. Games and simulations and their relationships to learning. In *Handbook of Research on Educational Communications and Technology (2nd edition)*, Spector M, Merrill M D, van Merrienboer J, Driscoll. M P (eds.), Lawrence Erlbaum Associates, 2004, pp.571-581.

[10] Rieber L P. Multimedia learning in games, simulations, and microworlds. In *The Cambridge Handbook of Multimedia Learning*, Mayer R E (ed.), Cambridge University Press, 2005, pp.549-567.

[11] Keith C. Agile Game Development with Scrum (1st edition). Addison-Wesley, 2010.

[12] Pressman R S. Software Engineering: A Practitioner's Approach (5th edition). Wiley, 2001.

[13] Petrillo F, Pimenta M, Trindade F, Dietrich C. What went wrong? A survey of problems in game development. *Computers in Entertainment*, 2009, 7(1): 13:1-13:22.

[14] Ramadan R, Widyani Y. Game development life cycle guidelines. In *Proc. the 5th International Conference on Advanced Computer Science and Information Systems*, September 2013, pp.95-100.

[15] Lee S H, Lee G H, Cho H H, Song D H, Rhew S Y. An empirical model of the game software development processes. In *Proc. the 4th International Conference on Software Engineering Research, Management and Applications*, August 2006, pp.371-377.

[16] Ayyad R, Masood M, Hosam A S, Abbas M. An optimization of CPS model in computer games development for non programmers. In *Proc. the 2nd International Conference on Computational Intelligence, Modelling and Simulation*, September 2010, pp.125-128.

[17] Pena J. Collaborative framework for browser games development. In *Proc. the 2011 Workshop on Open Source and Design of Communication*, July 2011, pp.65-72.

[18] Tschang F T. Beyond normal products and development processes: Computer games as interactive experiential goods and their manner of development. In *Proc. What do we Know about Innovation? A Conference in Honor of Keith Pavitt*, November 2003, pp.1-31.

[19] McGrath J. The game development lifecycle: A theory for the extension of agile project methodology, 2011. http://blog.dopplerinteractive.com/2011/04/game-development-lifecycle-theory-for.html, May 2014.

[20] Bethke E. Game Development and Production. Wordware, 2003.

[21] Hendrick A. Project management for game development. 2009. http://mmotidbits.com/2009/06/15/project-management-for-game-development/, Aug. 2016.

[22] O'Hagan A O, Coleman G, O'Connor R V. Software development processes for games: A systematic literature review. In *Systems, Software, and Services Process Improvement*, Barafort B, O'Connor R V, Poth A, Messnarz R (eds.), Springer, 2014, pp.182-193.

[23] Polk R. Agile and Kanban in coordination. In *Proc. the 2011 Agile Conference*, Aug. 2011, pp.263-268.

[24] Birchall J, Gatzidis C. The periodic table of elements via an XNA-powered serious game. In *Transactions on Edutainment IX*, Pan Z G, Cheok A D, Müller W, Liarokapis F (eds.), Springer, 2013, pp.1-28.

[25] Musil J, Schweda A, Winkler D, Biffl S. Improving video game development: Facilitating heterogeneous team collaboration through flexible software processes. In *Systems, Software and Services Process Improvement*, Riel A, O'Connor R, Tichkiewitch S, Messnarz R (eds.), Springer, 2010, pp.83-94.

[26] Kasurinen J, Laine R, Smolander K. How applicable is ISO/IEC 29110 in game software development? In *Lecture Notes in Computer Science 7983*, Heidrich J, Oivo M, Jedlitschka A, Baldassarre M T (eds.), Springer, 2013, pp.5-19.

[27] Kaitilla C. How to Learn Ouya Gamedev. 2014. http://gamedevelopment.tutsplus.com/articles/how-to-learn-ouya-gamedev–gamedev-9197, Aug. 2016.

[28] Penzenstadler B, Bauer V, Calero C, Franch X. Sustainability in software engineering: A systematic literature review. In *Proc. the 16th International Conference on Evaluation & Assessment in Software Engineering*, May 2012, pp.32-41.

[29] Ampatzoglou A, Stamelos I. Software engineering research for computer games: A systematic review. *Information and Software Technology*, 2010 52(9): 888-901.

[30] Viana J R M, Viana N P, Trinta F A M, de Carvalho W V. A systematic review on software engineering in pervasive games development. In *Proc. the 2014 Brazilian Symposium on Computer Games and Digital Entertainment*, Nov. 2014, pp.51-60.

[31] Claypool K, Claypool M. Teaching software engineering through game design. In *Proc. the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, June 2005, pp.123-127.

[32] Sundstrom E, De Meuse K P, Futrell D. Work teams: Applications and effectiveness. *American Psychologist*, 1990, 45(2): 120-133.

[33] Muchinsky P. Psychology Applied to Work (7th edition). Thomson, Wadsworth, 2003.

[34] Rossen E K, Bartlett R, Herrick C A. Interdisciplinary collaboration: The need to revisit. *Issues in Mental Health Nursing*, 2008 29(4): 387-396.

[35] Tran M Q, Biddle R. Collaboration in serious game development: A case study. In *Proc. the 2008 Conference on Future Play: Research, Play, Share*, November 2008, pp.49-56.

[36] Stacey P, Brown A, Nandhakumar J. Making sense of stories: The development of a new mobile computer game. In *Proc. the 40th Hawaii International Conference on System Sciences*, January 2007.

[37] de Barros R L B, Alves C F, Ramalho G L. Investigating the communication process in multidisciplinary game development teams. In *Proc. the 2009 Simposio Brasileiro de Sistemas Colaborativos*, October 2009, pp.61-69.

[38] Kasurinen J, Maglyas A, Smolander K. Is requirements engineering useless in game development? In *Lecture Notes in Computer Sciences 8396*, Salinesi C, van de Weerd I (eds.), Springer, 2014, pp.1-16.

[39] Bosser A G. Massively multi-player games: Matching game design with technical design. In *Proc. the 2004 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, June 2004, pp.263-268.

[40] Callele D, Neufeld E, Schneider K. Requirements engineering and the creative process in the video game industry. In *Proc. the 13th IEEE International Conference on Requirement Engineering*, August 29-September 2, 2005, pp.240-250.

[41] Callele D, Neufeld E, Schneider K. A proposal for cognitive gameplay requirements In *Proc. the 5th International Workshop on Requirements Engineering Visualization*, September 2010, pp.43-52.

[42] Reyno E M, Cubel J Á C. Automatic prototyping in model-driven game development. *ACM Computers in Entertainment*, 2009, 7(2): Article No. 29.

[43] Almeida M S O, da Silva F S C. A systematic review of game design methods and tools. In *Lecture Notes in Computer Science 8215*, Anacleto J C, Clua E W G, da Silva F S C, Fels S, Yang H S (eds.), Springer, 2013, pp.17-29.

[44] Ahmed I, Jaafar A. Games design and integration with user's emotion. In *Proc. the 2011 International Conference on User Science and Engineering*, Nov. 29-Dec. 1, 2011, pp.60-72.

[45] Bringula R P, Alcid A S, Bandril L B P, de Guzman A E, Lopez L J C. Development of game design guidelines. In *Proc. the 2nd International Conference on Technology, Informatics, Management, Engineering & Environment*, August 2014, pp.234-239.

[46] Sherrod A. Ultimate 3D Game Engine Design & Architecture. Charles River Media, 2006.

[47] Cowan B, Kapralos B. A survey of frameworks and game engines for serious game development. In *Proc. the 14th International Conference on Advanced Learning Technologies*, July 2014, pp.662-664.

[48] Hudlicka E. Affective game engines: Motivation and requirements. In *Proc. the 4th International Conference of Future of Digital Games*, April 2009, pp.299-306.

[49] Wu Y H, Yao X X, He J. Design and implementation of the game engine based on Android platform. In *Proc. the 2011 International Conference onInternet Technology and Applications*, August 2011.

946

J. Comput. Sci. & Technol., Sept. 2016, Vol.31, No.5

[50] Rodkaew Y. The last eternity a 3D role-playing game with a cross-platform development. In *Proc. the 2013 International Computer Science and Engineering Conference*, September 2013, pp.313-318.

[51] Vanhutupa J M. On the development of browser games-technologies of an emerging genre. In *Proc. the 7th International Conference on Next Generation Web Services Practices*, October 2011, pp.363-368.

[52] Sousa J P, Garlan D. Aura: An architectural framework for user mobility in ubiquitous computing environments. In *Software Architecture: System Design, Development and Maintenance*, Bosch J, Gentleman M, Hofmeister C, Kuusela J (eds.), Springer, 2002, pp.29-43.

[53] Aitenbichler E, Kangasharju J, Mühlhäuser M. MundoCore: A light-weight infrastructure for pervasive computing. *Pervasive Mobile Computing*, 2007, 3(4): 332-361.

[54] Pimenta M S C S, Buzeto F N, Santos L H O, Castanho C D, Jacobi R P. A game engine for building ubigames. In *Proc. the 13th Annual Workshop onNetwork and Systems Support for Games*, December 2014, pp.1-3.

[55] Neto B, Fernandes L, Werener C, de Souza J M. Reuse in digital game development. In *Proc. the 4th International Conference on Ubiquitous Information Technologies & Applications*, December 2009, pp.1-6.

[56] Peker A G, Can T. A design goal and design pattern based approach for development of game engines for mobile platforms. In *Proc. the 16th International Conference on Computer Games*, July 2011, pp.114-120.

[57] Llopis N. Optimizing the Content Pipeline. Game Developer Magazine, 2004.

[58] Hendrikx M, Meijer S, van der Velden J, Iosup A. Procedural content generation for games: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2011, 9(1): Article No. 1.

[59] De Carli D M, Bevilacqua F, Pozzer C T, d'Ornellas M C. A survey of procedural content generation techniques suitable to game development. In *Proc. the 2011 Brazilian Symposium on Games and Digital Entertainment*, November 2011, pp.26-35.

[60] Phelps A. Graphics don't matter (and other assertions). 2005. http://gotgame.cornate.com/archives/2005/-06/13/graphics_dont_matter_and_other_assertion.php, Aug. 2016.

[61] Pranatio G, Kosala R. A comparative study of skeletal and keyframe animations in a multiplayer online game. In *Proc. the 2nd International Conference on Advances in Computing, Control and Telecommunication Technologies*, December 2010, pp.143-145.

[62] Lasseter J. Principals of traditional animation applied to 3D computer animation. *ACM SIGGRAPH Computer Graphics*, 1987, 21(4): 35-44.

[63] Maestri G. Digital Character Animation 3. New Riders Publishing, 2006.

[64] Xu H, Wang C P. A review and development of 3D accelerator technology for games. In *Proc. the 2nd International Symposium on Intelligent Information Technology and Security Informatics*, June 2009, pp.59-63.

[65] Chehimi F, Coulton P, Edwards R. Advances in 3D graphics for smartphones. In *Proc. the 2nd International Conference on Information and Communication Technologies*, April 2006, pp.99-104.

[66] Manocha D, Calamia P, Lin M C, Manocha D, Savqia L, Tsingos N. Interactive sound rendering. In *Proc. the ACM SIGGRAPH 2009 Courses*, August 2009, Article No. 15.

[67] Pichlmair M, Kayali F. Levels of sound: On the principles of interactivity in music video games. In *Proc. DiGRA 2007 Conference*, 2007, pp.424-430. http://publik.tu-wien.ac.at/files/pub-inf_4783.pdf, Aug. 2016.

[68] Migneco R, Doll T M, Scott J J, Hahn C, Diefenbach P J, Kim Y E. An audio processing library for game development in flash. In *Proc. the 2009 International IEEE Consumer Electronics Society's Games Innovations Conference*, August 2009, pp.201-209.

[69] Wang A I, Norum M S, Lund C H W. Issues related to development of wireless peer-to-peer games in J2ME. In *Proc. the Advanced International Conference on Telecommunications and International Conference on Internet Web Applications and Services*, February 2006, p.115.

[70] Améndola F, Fernández M, Favre L. GLIESE, a framework for experimental game development. In *Proc. the 12th International Conference on Information Technology: New Generations*, April 2015, pp.528-533.

[71] Gamma E, Helm R, Johnson R, Vilssides J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional, 1995.

[72] Lukashev D, Puresev A, Makhlushev I. 3D applications for 3G mobile phones: Design, development, resource utilization. In *Proc. the 2006 IEEE International Symposium on Consumer Electronics*, July 2006.

[73] El Rhalibi A, Merabti M, Carter C, Dennett C, Cooper S, Sabri M A Fergus P. 3D Java Web-based games development and deployment. In *Proc. the International Conference on Multimedia Computing and Systems*, April 2009, pp.553-559.

[74] Jhingut M Z, Ghoorun I M, Nagowah S D, Moloo R, Nagowah L. Design and development of 3D mobile games. In *Proc. the 3rd International Conference on Advances in Computer-Human Interactions*, February 2010, pp.119-124.

[75] Kosmopoulos A, Karamichali I, Kemerlis V P, Polyzos G C. Fueling game development in mobile P2P environment. In *Proc. the 18th International Symposium on Personal, Indoor and Mobile Radio Communication*, September 2007.

[76] Al-Azawi R, Ayesh A, Al-Obaidy M. Towards agent-based agile approach for game development methodology. In *Proc. the 2014 World Congress on Computer Applications and Information Systems (WCCAIS)*, January 2014.

[77] Segundo R M C, da Silva J C F, Tavares T A. ATHUS: A generic framework for game development on Ginga middleware. In *Proc. the 2010 Brazilian Symposium on Games and Digital Entertainment*, November 2010, pp.89-96.

[78] Redavid C, Farid A. An overview of game testing techniques. 2011. http://www.idt.mdh.se/kurser/ct33-40/ht11/MINICONFERENCE/FinalPapers/ircse11_submission_15.pdf, Aug. 2016.

[79] Helppi V V. The agile process for mobile game development and testing, Testdroid Blog. June, 2015. http://testdroid.com/tech/the-agile-process-for-mobile-game-devlopment-and-testing, Aug. 2016.

[80] Schultz C P, Bryant R D, Langdell T. Game Testing: All in One. Thomson Course Technology, 2005.

[81] Wilson D. Quality quality assurance: A methodology for wide-spectrum game testing, Gamasutra Blog. April 2009. http://www.gamasutra.com/view/feature/132398/quality_quality_assurance_a_.php, Aug. 2016.

[82] Marri K K, Sundaresasubramanian G. ExPLORE: Testing the game. 2015. http://www.infosys.com/IT-services/independent-validation-testing-services/Documents/test-games-users-perspective.pdf, Aug. 2016.

[83] Kasurinen J, Smolander K. What do game developers test in their products? In *Proc. the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, September 2014, Article No. 1.

[84] Al-Azawi R, Ayesh A, Obaidy M A. Generic evaluation framework for games development methodology. In *Proc. the 3rd International Conference on Communications and Information Technology*, June 2013, pp.55-60.

[85] Omar H M, Jaafar A. AHP_HeGES: Tools to evaluate usability of educational computer game (UsaECG). In *Proc. the 2011 International Conference on User Sciences and Engineering*, Nov.29-Dec.1, 2011, pp.73-76.

[86] Strååt B, Warpefelt H. Applying the two-factortheory to the play heuristics. In *Proc. DiGRA*, May 2015. http://www.digra.org/digital-library/publications/applying-the-two-factor-theory-to-the-play-heuristics/, Aug. 2016.

[87] Sarinho V T, Apolinário A L. A generative programming approach for game development. In *Proc. the 8th Brazilian Symposium on Games and Digital Entertainment*, October 2009, pp.83-92.

[88] Czarnecki K, Kim C H P. Cardinality-based feature modeling and constraints: A progress report. In *Proc. the International Workshop on Software Factories*, October 2005.

[89] Chen Y W, Keivanloo I, Roy C K. Near-miss software clones in open source games: An empirical study. In *Proc. the 27th Canadian Conference on Electrical and Computer Engineering*, May 2014.

[90] Anderson E F. A classification of scripting systems for entertainment and serious computer games. In *Proc. the 3rd International Conference on Games and Virtual Worlds for Serious Applications*, May 2011, pp.47-54.

[91] Xu S C, Rajlich V. Empirical validation of test-driven pair programming in game development. In *Proc. the 5th IEEE/ACIS International Conference on Computer and Information Sciences and 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse*, July 2006, pp.500-505.

[92] Zhang W S, Han D, Kunz T, Hansen K M. Mobile game development: Object-orientation or not. In *Proc. the 31st Annual International Computer Software and Applications Conference*, July 2007, pp.601-608.

[93] Wang A I, Nordmark N. Software architectures and the creative processes in game development. In *Lecture Notes in Computer Science 9353*, Chorianopoulos K, Divitini M, Hauge J B, Jaccheri L, Malaka R (eds.), Springer International Publishing, 2015, pp.272-285.

[94] Meng L S, Kelvin, Parakash E C, Loh P K K. Design and development of a peer-to-peer online multiplayer game using DirectX and C#. In *Proc. the 2004 IEEE Region 10 Conference*, November 2004, pp.278-281.

[95] Cronbach L J. Coefficient alpha and the internal structure of tests. *Psychometrika*, 1951, 16(3): 297-334.

[96] Osterhof A. Classroom Applications of Educational Measurement. Prentice-Hall, 2001.

[97] Nunnally J C, Bernstein I H. Psychometric Theory (3rd edition). McGraw-Hill, 1994.

[98] Van de Ven A H, Ferry D L. Measuring and Assessing Organizations. Wiley, 1980.

[99] Comrey A L, Lee H B. A First Course in Factor Analysis (2nd edition). Lawrence Erlbaum, 1992.

[100] Campbell D T, Fiske D W. Convergent and discriminant validation by the multitraitmultimethod matrix. *Psychological Bulletin*, 1959 56(2): 81-105.

[101] Kaiser H F. The application of electronic computers to factor analysis. *Educational and Psychological Measurement*, 1960, 20(1): 141-151.

[102] Kaiser H F. A second generation little jiffy. *Psychometrika*, 1970, 35(4): 401-417.

[103] Fornell C, Bookstein F L. Two structural equation models: LISREL and PLS applied to consumer exit-voice theory. *Journal of Marketing Research*, 1982 19(4): 440-452.

[104] Joreskog K, Wold H. Systems under Indirect Observation: Causality, Structure, and Prediction. North Holland, 1982.

[105] Westfall P H, Young S S. Resampling Based Multiple Testing (1st edition). Wiley, 1993.

[106] San Martin-Rodriguez L, Beaulieu M D, D'Amour D Ferrada-Videla M. The determinants of successful collaboration: A review of theoretical and empirical studies. *Journal of Interprofessional Care*, 2005, 19(S1): 132-147.

[107] Pietroburgo J, Bush B. Coming to terms: A case study of hospice collaboration challenges. *American Journal of Hospice & Palliative Medicine*, 2008, 24(6): 487-492.

[108] Huh J, Ackerman M S, Erickson T, Harrison S, Sengers P. Beyond usability: Taking social, situational, cultural, and other contextual factors into account. In *Proc. Extended Abstracts on Human Factors in Computing Systems* (*CHI EA 2007*), April 28-May 3, 2007, pp.2113-2116.

[109] Iuppa N, Borst T. End-to-End Game Development. Focal Press, 2010, pp.225-229.

[110] Anderson E F, Engel S, Comninos P, McLoughlin L. The case for research in game engine architecture. In *Proc. the 2008 Conference on Future Play: Research, Play, Share*, November 2008, pp.228-231.

[111] Aleem S, Capretz L F, Ahmed F. Empirical investigation of key business factors for digital game performance. *Entertainment Computing*, 2016, 13: 25-36.

[112] Easterbrooks S, Singer J, Storey M A, Damian D. Selecting empirical methods for software engineering research. In *Guide to Advanced Empirical Software Engineering*, Shull F, Singer J, Sjøberg D I K (eds.), Springer, 2008, pp.285-311.

[113] Wohlin C, Runeson P, Höst M, Ohlsson M C, Regnell B, Wesslén A. Experimentation in Software Engineering: An Introduction. Kluwer Academic Publishers, 2000.

[114] Chow S C, Shao J, Wang H S. Sample Size Calculations in Clinical Research (2nd edition). Tylaor & Francis Group, 2008.

[115] Bryman A. Social Research Methods (3rd edition). Oxford University Press, 2008.

[116] Heckshaw A. Small studies: Strengths and limitations. *European Respiratory Journal*, 2008, 32(5): 1141-1143.

[117] Zhang C, Zhang J. Analysing Chinese citizens' intentions of outbound travel: A machine learning approach. *Current Issues in Tourism*, 2014, 17(7): 592-609.

Saiqa Aleem received her M.S. degree in computer science from University of Central Punjab, Pakistan, in 2004, M.S. degree in information technology from UAEU, United Arab Emirates University, in 2013, and Ph.D. degree in software engineering from University of Western Ontario, Canada, in 2016. Currently, she is an assistant professor at Zayed University. She had many years of academic and industrial experience, holding various technical positions. She is Microsoft, CompTIA, and CISCO certified professional with MCSE, MCDBA, A+ and CCNA certifications.

Luiz Fernando Capretz has vast experience in the software engineering field as practitioner, manager and educator. Before joining the University of Western, Ontario, Canada, he worked at both technical and managerial levels, taught and did research on the engineering of software in Brazil, Argentina, England, Japan, and the United Arab Emirates since 1981. He is currently a professor of software engineering and assistant dean (IT and e-Learning), and was former director of the Software Engineering Program at the Western University, Ontario, Canada. His current research interests are software engineering, human aspects of software engineering, software analytics, and software engineering education. Dr. Capretz received his Ph.D. degree in computer science from the University of Newcastle upon Tyne, U.K., 1991 and M.Sc. degree from the National Institute for Space Research (INPE-Brazil), and B.Sc. degree from Universidade Estadual de Campinas (Brazil). He is a senior member of IEEE, a distinguished member of the ACM, an MBTI certified practitioner, and a certified professional engineer in Canada (P.Eng.).

Faheem Ahmed received his M.S. degree and Ph.D. degree in software engineering from the Western University, Ontario, Canada, in 2004 and 2006 respectively. Currently he is an associate professor and a chair at Thompson Rivers University, Kamloops, Canada. Ahmed had many years of industrial experience, holding various technical positions in software development organizations. During his professional career he has been actively involved in the life cycle of software development process including requirements management, system analysis and design, software development, testing, delivery and maintenance. Ahmed has authored and coauthored many peer-reviewed research articles in leading journals and conference proceedings in the area of software engineering. He is a senior member of IEEE.

Appendix Measuring Instrument

This survey attempts to evaluate key success factors in the game development process statistically from a developer's perspective. This survey captures the opinions of game developers who have completed game projects regarding factor collaboration, game design documents, game engines, game asset creation, game architecture, game testing, and programming.

If you are a game developer with at least one team project under your belt, please help us by taking the survey below for the game project you completed most recently and also give your opinions about enhanced the game development process.

Section 1 – Qualifying Questions

1. Please take this survey for the most recent game development project for which you can answer "yes" to ALL the following questions:

☐ There were at least three fulltime developers on this team.

☐ I worked on the project for at least one-third of its total duration.

☐ The project was either completed or cancelled sometime within the last three years.

☐ I worked in the development team in some sort of development role, such as a designer, artist, animator, programmer, producer, or sound designer.

This survey should not take more than 10~15 minutes to complete.

Your answers will be kept confidential, but the AGGREGATE data will be released to the public along with our conclusions.

Section 2: Background Questions

2. What is your region? ⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

3. What was the total duration of your game development project? Enter the whole number.

Years: ⎯⎯⎯⎯⎯⎯⎯⎯ Months: ⎯⎯⎯⎯⎯⎯⎯⎯

4. Approximately, what is the size of development team?⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

5. Please describe your primary role in the development process. Please select all that apply.

Artist/animator ☐   Programmer ☐   Designer ☐   Producer ☐

Audio Designer ☐   Other (please specify) ☐

6. The developed game was released for which platform? Select all that apply.

Any desktop ☐   Any handheld device ☐   Any console Web ☐

Any mobiles ☐   Other ☐

7. What was the genre of the developed game?

Please Answer: ⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

8. Which software development methodology was used to develop the game?

Please pick the approach that seems closest based on the descriptions below.

☐ Don't know.

☐ Waterfall: the project was divided into phases which included upfront planning, requirements, design, development, and testing phases.

☐ Agile: project leaders evaluate the project priorities on weekly or monthly sprints. Iterative development was focussed on individual features, and frequent feedback was emphasized rather than requirements, specifications, or design documents.

☐ Agile using Scrum: the project followed the "Scrum" implementation of Agile. Priorities were determined by self-organizing cross-disciplinary teams. These teams were responsible for their own tasking and held daily scrum meetings to identify the work being done and bottlenecks to development.

☐ Other/Ad-hoc.

Section 3: Evaluation of Enhanced Game Development Process Success Factors Identified Through Literature Review

The questionnaire objective is to find out which factors have a positive impact on the game development process. Please select the correct scale based on your best knowledge.

950

J. Comput. Sci. & Technol., Sept. 2016, Vol.31, No.5

Key Factors for the Game Development Process from A Developer's Perspective							
Likert scale (1 = strongly disagree; 2= disagree; 3 = neutral; 4= agree; 5= strongly agree)		1	2	3	4	5	N/A
Team Configuration & Management							
1	The team must be organized into sub-teams by disciplines (art, programming, design) rather than by features.						
2	Team members must have a similar vision of the game throughout the development process.						
3	There must be support from lead management to the team members.						
4	The entire team should be involved in prioritizing the work to be done for each milestone or sprint.						
5	In case of any significant change in the game design or architecture, then all stakeholders must participate in the decision process.						
6	The development plan for the game should be clear and well communicated to the team.						
Game Design Document Management							
7	There must be a design document available to the team near the beginning of development that clearly specifies the game goals.						
8	Priorities must be given to different components so the team will know which part is more important.						
9	Details about storyboard, script writing, characters, and major and minor goals must be included in the GDD.						
10	The GDD was understandable because it was well written.						
11	Transformation of the GDD from the pre-production phase to the production phase was not problematic.						
Game Engine Development							
12	The development platform and tools must be familiar to game developers.						
13	The selected development tool provided asset and resource management.						
14	The selected game engine was able to handle diverse types of input and output						
15	Integration of all technological components was easy.						
16	The game engine provided support for multi-platform development.						
17	The development tool enables use of other embedded tools that are helpful in extension of current capabilities.						
18	Reuse of the game engine is highly desirable.						
Game Asset Management							
19	Realism and performance analysis must be a part of asset creation.						
20	Realism and control investigation before asset creation is important.						
21	Integration of sound effects into complex and unexpected scenes can usually be done by using available audio processing libraries.						
22	Asset version control management must be performed to track different versions.						
Quality of Game Architecture							
23	Gameplay was divided into different modules, and each module could be modified and tested independently without impacting other modules.						
24	Different game modules should be easily portable and extensible so that they can be plugged into other game projects.						
25	The game architecture included robustness features that enable a game to be functional under unexpected circumstances.						
Game Test Management							
26	Game testing steps were usually established during the pre-production phase and documented properly.						
27	Game testing was performed throughout the game development process.						
28	A suitable testing approach was selected to test game performance and quality.						
29	The game was tested for performance under various loads.						
Programming Practices							
30	Programming team responsibilities and job roles were carefully matched with their particular programming skills and abilities.						
31	Programming style must be uniform among all programmers.						
32	Good commenting reduces the errors in code and speeds up the code review process.						
33	Standard naming and coding conventions should be used.						
34	Performance and optimization techniques (such as methodological and code optimization and datatype optimization) were applied to the code.						
Enhanced game development process							
1	The game engine should allow rapid prototyping of new levels, behaviour, and scenarios and support dynamic content loading.						
2	Game architecture should be easy to understand, change, reuse, and debug.						
3	The game design document should be developed in a formal way and have all specifications such as executive summary, product, game and art specifications.						
4	Game assets should be created to fit into the game concept and must have a positive effect on the appearance of the game.						
5	Coding priorities must be established as a part of technical design and must be properly documented						
6	Before selection of a programming strategy, issues such as coupling between modules, performance, memory management, and availability of different programming paradigms should be taken into consideration.						
7	All aspects of the game were tested, such as game play, functionality, interaction control, connectivity issues, input controller, and platform compatibility.						
8	The entire team should meet frequently to openly discuss topics of interest, ask questions, and identify production bottlenecks.						
9	Game testing should be performed properly to ensure game performance and quality.						