Software Project Plan for Mobile Games Development: A Quasi-Systematic Review

Fernando Melo Nascimento[1]* Antônio José Alves Neto[1] Beatriz Trinchão Andrade[2]
Rogério Patrício Chagas do Nascimento[1]

[1] Federal University of Sergipe, Pos Graduation in Computer Science Program, Brazil
[2] Federal University of Sergipe, Computer Department, Brazil

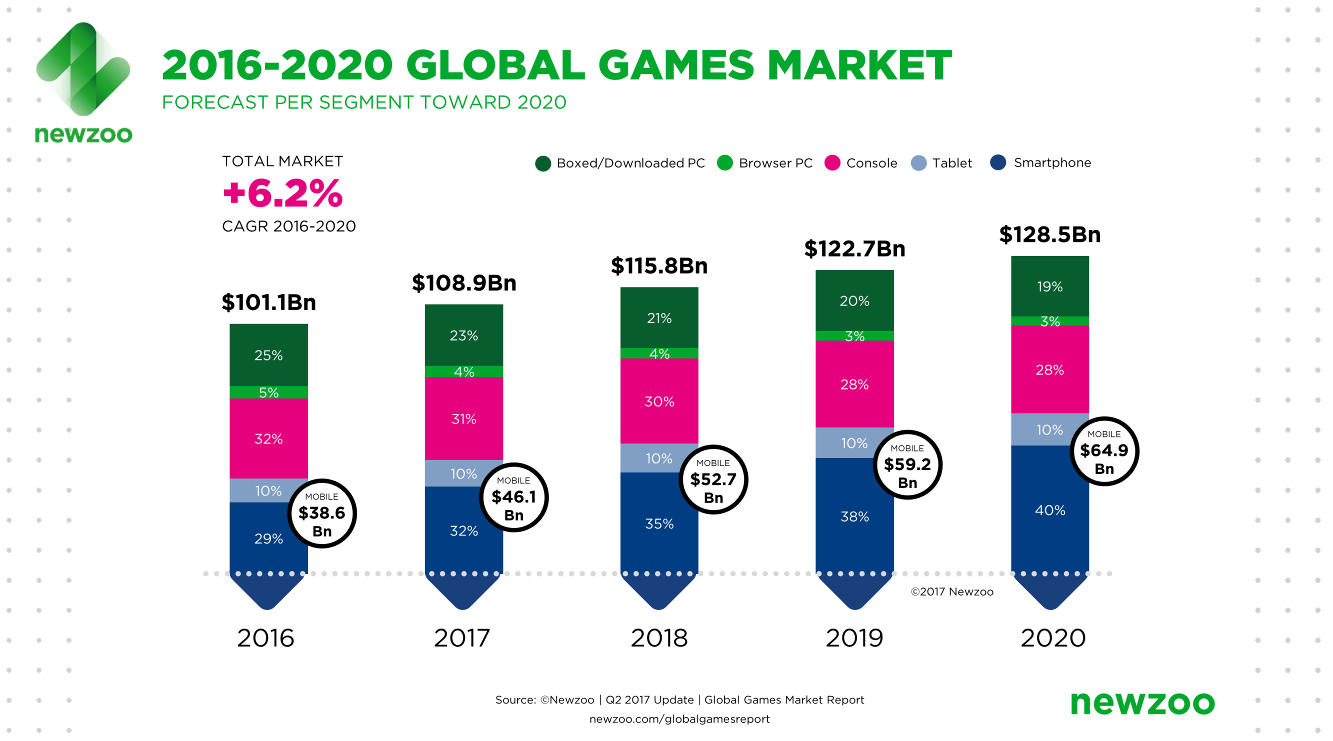

Figure 1: Forecast of global games revenue from 2016 to 2020. Dark blue area corresponds to mobile games. Adapted from McDonald [12].

Abstract

Observing the growth of mobile games industry in the past few years, many companies turned their eyes to this market. Each year, the number of released mobile games grows, but the successful ones do not follow the same pace. One of the reasons of this phenomenon is the lack of correct planning and estimation. This paper performs a quasi-systematic review on general software project plans and techniques used both in academy and industry in order to identify which plan is most suited for mobile games. The results show that most mobile games are usually developed using agile methodologies, which suggests that agile planning techniques can be very useful in this context. Among the several works researched and techniques found, we highlight the GAMED methodology, created for the development of educational games. With some adaptations aiming the mobile games' market, this methodology can provide a robust way to develop mobile games, especially when allied with agile planning techniques.

Keywords: software project plan, mobile game development, agile planning techniques, GAMED methodology.

*email: nascimentofm@ufs.br

1 Introduction

Planning a software is a crucial and complex task. Ideally, it should be performed during the conception phase of the project in order to guide the developers to build the right product and maximize the project's success. This can be accomplished by correctly estimating a set of project's features like schedule, resources, risks and costs [14] [17]. There are different techniques used to identify and estimate those features in a software project, ranging from even artistic or ad-hoc methods, as pointed by Pressman [14], to sophisticate solutions as machine learning algorithms [13].

In industry, it is mandatory to increase the success rate of a project as much as possible, since this usually implies higher profits. However, as noticed by Garousi et al. [9], the collaboration in software engineering between industry and academy is still very rare, despite the large size of both communities. This gap has a huge impact on the development of successful projects, since the promising innovations and best results of one community is hardly applied on the other.

One of the fastest growing industries in recent years is the mobile games industry. As stated by Filho et al. [7], in 2013 global game industry revenue exceed 93 billion dollars, growing even more on subsequent years. This revenue refers to console, browser, Massive Multi-player Online (MMOs) and mobile games, this last one being responsible for the larger part of the revenue.

As reported by McDonald [12], mobile games' revenue in 2017 is expected to be higher than 46 billion dollars. Figure 1 presents a forecast of the global games revenue for the next years, with mobile industry (represented in dark blue by smartphones and tablets) occupying the larger revenue proportion and growing higher each year.

As simple as a mobile game can be in comparison with other software applications, its development process must not be conducted in an ad-hoc manner. First, because the life cycle of a mobile game is different from a traditional software [3] and second, due to the low success rate observed on software projects developed with inadequate planning or no planning at all [18].

Observing the software project plans developed for industrial and academic usage, the gap between industry-academy collaboration, and also the growth of mobile games industry in the past years, this paper has the following goals:

- Present a review of the software project plans used in academy, industry and both;
- Identify the most used plans in academy and industry;
- Highlight which plan is most suited for mobile games development.

This paper is organized as follows: Section 2 presents a theoretical reference about software project plans and mobile game development process. Section 3 describes a methodology (called GAMED) created to cover the whole development of a digital educational game and the proposals to adapt it to the mobile games development. Section 4 presents the results of this work and a short discussion. Finally, conclusions and suggestions for future work are made in Section 5.

2 Theoretical Reference

This section presents a theoretical reference on software project plans and mobile games development. It is divided in four subsections: search method (2.1), presenting the search string used and the selection criteria of relevant papers; academic software project plans (2.2), where the most used plans in academy are presented; industry software project plans (2.3), which describes the most used plans in industry and the different techniques used for estimation; and mobile game development process (2.4), that introduces some strategies of mobile game development and the issues related to it.

2.1 Search Method

Scopus database was used to search for relevant works about software project plans used in academy, industry and both. This database was chosen due to its coverage, containing works of main databases such as IEEE, ACM and Springer. The database was accessed through the CAPES journals portal [1] and the search string built was:

"(TITLE-ABS-KEY ((software AND engineering AND project AND planning AND management) AND (method) OR (technique) OR (framework))) AND (LIMIT-TO (SUBJAREA , "COMP"))".

This query aimed papers that contain in their titles, abstracts or keywords the words "software", "engineering", "project", "planning" and "management" together, allied with "method", "technique" or "framework". Also, the query was limited to the papers in computer science area.

The search string returned a total of 491 papers. After reading their respective abstracts, 394 were excluded, resulting in 97 works selected for further reading. From these, only 15 papers were considered relevant to this work.

The selection criteria adopted to identify and discard non relevant works were:

1. Exclude duplicated papers;
2. Exclude works published before 2005;
3. Exclude papers with title, abstract or keywords not related to this paper's objectives.

2.2 Academic software project plans

In academy, the most used software project plans are the ones proposed by Pressman [14] and Sommerville [17]. These plans are widely referenced in literature and often used on real applications in order to measure specific metrics of projects, such as their size, complexity, duration and quality. Also, a novel plan proposed by Frailey [8] fit this category.

Pressman [14] defines the software project planning as a process that aims to provide to the project manager a framework capable of performing estimates of a project, as its risks, costs, resources and life time. To obtain realistic estimations, the scope of the project must be well defined, in order to acquire correct information about the previously cited features.

In similar way, Sommervile [17] describes a software project plan as a divide and conquer task. Managers should split the project into smaller parts and assign them to the team members. The project plan itseft can be divided into different plans, as quality, maintenance and validation plans.

Frailey [8] proposed a novel technique for teaching project planning on academy by treating the course itself as the project. This way, students can experience the project planning in their own activities without a real software project. The students created a spreadsheet with the course's plan and track all the progress on it.

2.3 Industry software project plans

Literature review on software project plans used in industry points to a variety of methods of estimation for different project features, in most cases following a standard plan model as the ones proposed by Pressman [14] and Sommerville [17].

Albadarneh et al. [2] made a comparative study of risk management techniques used in Agile Software Development. They analyze how risk management is performed on SCRUM, XP and DSDM methodologies. A comparative table was filled with the advantages and disadvantages pointed by different works using one of the methodologies mentioned. They conclude that DSDM methodology provides the most comprehensive approach to risk management.

Drury et al. [6] showed that the obstacles faced on decision making in agile development are critical, yet poorly understood. Their research examined decisions made across four stages of the iteration cycle: planning, execution, review and retrospective. Through interviews, initially with 43 agile managers and developers and subsequently with 18 individuals from five different organizations, six decision obstacles were identified. Their effects include a lack of longer-term and strategic focus for decisions, an ever-growing backlog of delayed work from previous iterations, and a lack of team engagement. The obstacles are listed below:

1. Unwillingness to commit to decisions;

2. Conflicting priorities;

3. Unstable resource availability;

4. Lack of implementation;

5. Lack of ownership;

6. Lack of empowerment.

Malhotra and Chung [11] made a study to provide insights on the impact of using the agile framework with Scrum on projects compared to the Iterative Enhancement Model (IEM). The same product developed using Scrum and IEM was compared through diverse metrics. Due to the customer involvement, it was observed that agile methodology encourages better planning.

Logue and McDaid [10] performed a case study of releasing planning using a statistical methodology. Releasing plans are documents that specify which features should be developed and released first in a iterative development. They observed that their methodology was able to manage uncertainty in project releases, identifying the best subset of features to be developed on each release.

Chang [4] proposed an approach to estimate and manage contingency reserves in software project using agile methodologies. The contingency reserves refer to the quantity of time, person, or capital that is reserved to the project development. A resources capability formula was developed to indicate how much and when the resources should be used.

Palacios et al. [5] presented a tool able to predict the competence level of team leaders based on their developed features. Based in Artificial Neural Networks (ANN), this framework allows the forecast and anticipation of competence needs, thus articulating personnel development tools and techniques.

ZurMuehlen and Ho [19] described in their work the use of frameworks to help planning the project iterations, such as CobIT, COBO and ERM. These frameworks possess native resources that contribute to risk identification, but the effectiveness of risk management in projects that uses agile methodology would depend on the team's experience.

Aslan and Balci [3] proposed two methodologies to improve educational game development process called GAMED (diGital educAtional gaMe dEvelopment methoDology) and IDEALLY (dIgital eDucational gamE softwAre quaLity evaLuation methodologY). These methodologies have a body of methods, rules and postulates, being embedded within the digital educational game life cycle. The life cycle itself describes a framework for the organization of the phases, processes, work products, quality assurance and project management activities required to develop, use, maintain and evolve a digital educational game from birth to retirement.

2.4 Mobile game development process

This section introduces some strategies of mobile game development and the issues related to them. It is presented the mobile game life cycle, the most used development methodologies and how a good software project plan can improve the success rate of a mobile game.

Due to the mobile games high revenue, presented in Section 1, many companies of diverse sizes and even independent developers turned their eyes to the mobile game market. This attention increased the quantity of games released, but the amount of successful ones did not follow the same pace. There are a lot of reasons that explain this phenomenon, some of which directly related to the development process, as the adoption of wrong methodology, lack of a project plan and unrealistic estimates [16].

The traditional game development process is composed of three main phases [15]:

1. Pre-production: design and concept development.

2. Production: implementation.

3. Pos-production: tests and deployment.

These phases are related with traditional software development phases. Prototyping is generally used in the pre-production phase to help the game designers visualize the game idea and features. With the concept (i. e. scope) defined, the production phase consists in implementing the actual game. With the software implemented, tests are performed in pos-production, being deployed/released in this phase [15].

Diverse authors point to agile methodologies as best suited for mobile game initial development, specially during prototyping [15] [16]. However, in production phase, more robust methodologies are preferred, as spiral methodologies [3] [16]. Figure 2 illustrates the spiral game development process proposed by Aslan and Balci [3].

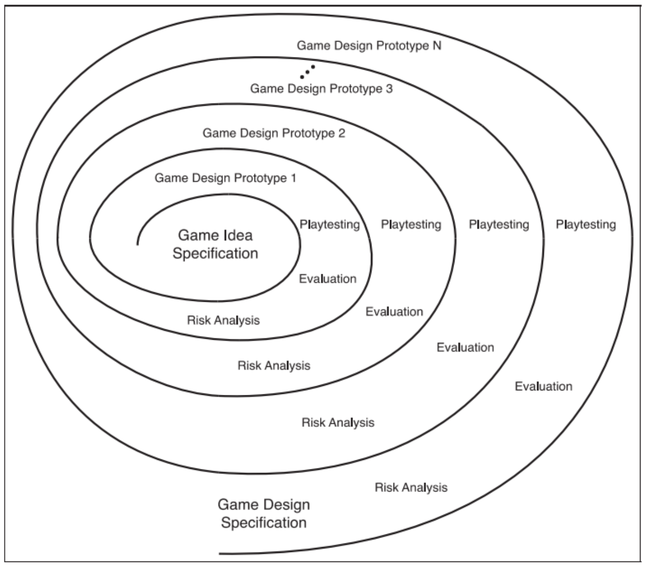

Figure 2: Spiral game development. Adapted from Aslan and Balci [3].

One important difference from traditional software development processes to mobile games development lies after deployment stage. Mobile games are constantly being updated, fixed and incremented with new features after the release, which increase the complexity of planning these projects. This way, software project plans suited for mobile games should be prepared for changes and yet provide realistic estimates. It is also important to stress that the mobile game life-cycle tends to be bigger than regular software, mixing maintenance with development [3].

Three stages are important to the success of a mobile game. They are usually treated as a funnel (called ARM funnel), as shown in Figure 3 since early stages are easier to accomplish compared to the latter ones. The stages are [7]:

1. Acquisition: stage focused on trying to acquire new players, i. e., attract new users to play the game. This is often responsibility of the publisher, being more an advertisement stage.

2. Retention: stage where new players are motivated to keep playing the game, normally by addictive mechanics and/or engagement with the game.

3. Monetization: stage where a player spends money inside the game, generating revenue to the developers.

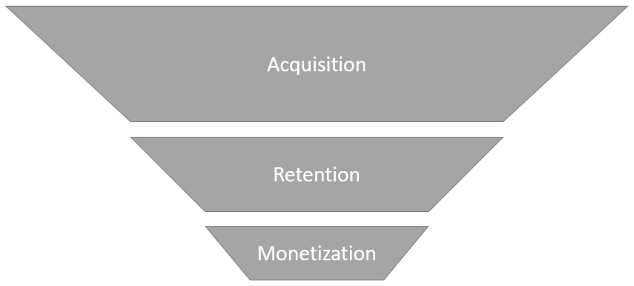

Figure 3: Important stages of mobile games success represented as a funnel. Adapted from Filho et al. [7].

There are diverse techniques that can be used to improve each stage. Also, since mobile games revenue is strictly tied to these stages, each feature on the game should be planned paying attention to these stages [7].

3 Adapting GAMED methodology to mobile games

Observing the software project plans and the diverse techniques for estimation present in the literature, GAMED methodology covers the whole educational game and serious game development process. Thus, this methodology can be used for mobile game development with some adaptions, since educational games usually do not have financial profit as main goal. This section explains GAMED methodology with more details, highlighting the differences between the process and goals of educational and mobile games.

Figure 4 presents the GAMED Methodology Development Life Cycle. Table 1 summarizes the four phases and their respective stages based on the digital educational game life cycle. Each stage may have an individual and distinct quality assurance step (QA) associated to it, in order to guarantee final product's quality. Also, this methodology can be applied using both traditional and agile development.

Table 1: GAMED Methodology phases and stages.

Phase	Stage
Game Design	1. Problem Formulation 2. Game Idea Generation 3. Game Design
Software Design	1. Requirements Development 2. Archtecting 3. Software Design
Implementation	1. Programming Structure 2. Integration 3. Publishing
Feedback	1. Game-based Learning 2. Feedback

Even though the stages are numerated sequentially, the life cycle is not strictly sequential. Some stages can be suppressed and reverse transitions can always be performed if the output of a stage does not meet the specified quality. This stresses the iterative nature of the process.

The following subsections explains the goal of each phase and what is done on their respective stages. Also, we suggest the necessary changes for fitting GAMED methodology to the mobile games development. It is important to stress that each stage produces documentation used by subsequent ones [3].

3.1 Game Design Phase

The first phase consists in generating the game idea or concepts for new features to be incorporated to the game. The first stage is the *Problem Formulation*, where a problem is identified and clearly defined. In an educational game, this stage consists in identifying subjects that pose serious challenges for learning. On mobile games, this stage is less restricted, so general challenges can be identified.

Once the problem is formulated, it should be solved inside the game. *Game Idea Generation* is the stage where the team produce game ideas that can be transformed into a game mechanic capable to explore and solve the formulated problem.

Finally, *Game Design* stage takes the selected game ideas and turn them into game mechanics, normally by using prototyping and performing risk analysis. When a mechanic is successfully generated, the first phase ends.

One important observation is that this phase should take into account how the new feature will impact the retention and monetization for mobile games. Thus, new features should be developed planning these mobile game concepts.

3.2 Software Design Phase

This phase consists in gathering the needed information to implement the game idea generated in the previous phase and building the ground for the implementation. It can be executed in the same way for both educational and mobile games.

During *Requirements Development* stage, both functional and non-functional requirements are identified and documented. The next stage is *Architecting*, which is the process of creating and specifying an architecture, which is more important for the games that requires network connection. Then, *Software Design* stage creates a software design from the architecture using design patterns.

3.3 Implementation and Publishing Phase

The third phase turns the game idea generated in the first phase into code. As the previous phase, this one have no clear distinction from educational to mobile games development.

Programming Structure stage consists in coding the software designed using a programming language. Next, this new code is integrated within the rest of the game during *Integration* step. After connecting all the components developed on the current version, the game is published to a server like Apple Store and Google Play on *Publishing* stage.

3.4 Game-based Learning and Feedback Phase

The last phase consists in analyzing the impact of the game. Here, there is a clear distinction of educational games analysis to mobile games. *Game-based Learning* only makes sense for educational games, since this stage assess the impact of the game on the students for learning a subject [3]. For mobile games, this stage should be replaced by an *ARM Analysis* stage, where acquisition and especially retention and monetization are assessed in terms of the whole game and/or new implemented feature(s).

The *Feedback* stage focuses on documenting the results of previous analysis in order to improve the game when the cycle restarts. For mobile games, community feedback, i. e. requests and suggestions, is also included.

3.5 Adapted GAMED

With the modifications suggested, GAMED methodology adapted to mobile games development can be summarized as in Table 2, where the highlighted stages are the ones with proposed modifications.

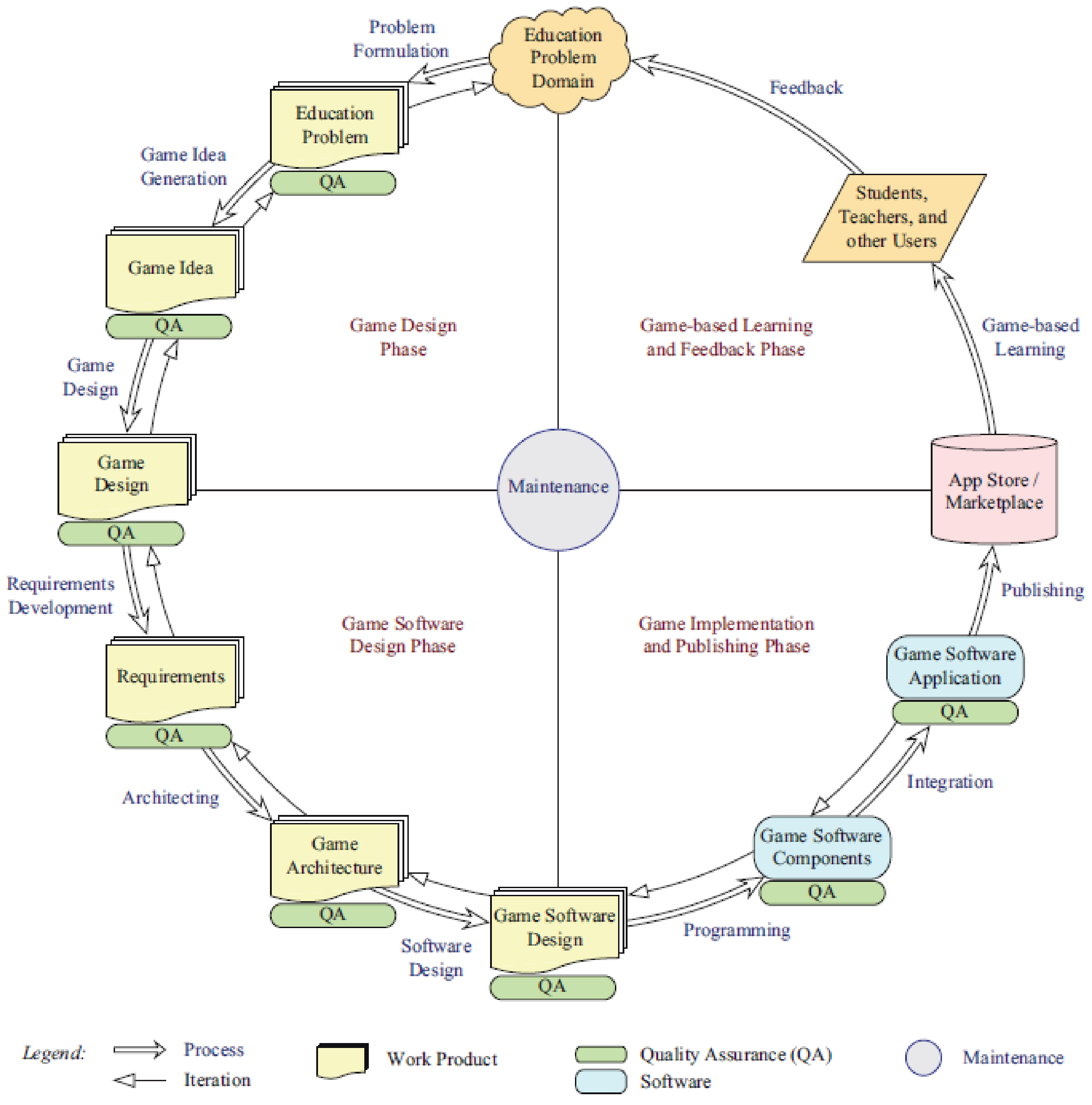

Figure 4: Software Development Life Cycle using GAMED methodology. Adapted from Aslan and Balci [3].

4 Results and Discussion

This section presents the results obtained through the quasi-systematic literature review and the discussion.

On academic software project plans literature, it was observed that Pressman [14] and Sommerville [17] plans were the most used. In addition, Frailey [8] proposed a method to teach academic project planning without a software project. Table 3 compares the academic plan documents presented on Section 2.2, where the highlighted sections present the main differences between them.

It can be observed that Pressman's and Sommerville's plans are very similar, and most sections of one document is present on the other with a different name and/or index order. However, Sommerville's plan organization suggests a clearer division of the work tasks on *Work breakdown* section, while Pressman's plan has a dedicated section to estimates. On the other hand, Frailey's plan also emphasizes work division, but it is more focused on tracking project progress than the other ones.

Regarding industry software project plans, it was observed that papers are focused on comparing and presenting new techniques to perform estimates, rather than proposing a full plan. Those techniques, reviewed on Section 2.3, are summarized on Table 4.

During literature review was also observed that mobile games usually are developed using agile methodologies, like Scrum and XP. This indicates that planning techniques suited for agile methodologies can be successfully applied to mobile games development. However, as noted by Ramadan and Widyani [16], more robust methodologies, like spiral development, should be considered on latter stages of the development, specially on bigger projects.

As detailed on Section 3, GAMED methodology covers the whole educational game development process, what makes it possible to use it for mobile games with some adaptions. Major changes were suggested on Game Design and Feedback phases, respectively

Table 2: GAMED Methodology adapted to Mobile Games Development.

Phase	Stage
Game Design	**1. General Problem Formulation** 2. Game Idea Generation 3. Game Design
Software Design	1. Requirements Development 2. Archtecting 3. Software Design
Implementation	1. Programming Structure 2. Integration 3. Publishing
Feedback	**1. ARM Analysis** **2. Community Feedback**

Table 3: Academic software project plan documents and their structural organization.

Author	Software Project Plan Document
Pressman [14]	1. Introduction 2. **Project Estimates** 3. Risk Management 4. Project Schedule 5. Staff Organization 6. Tracking and Control Mechanisms 7. **Appendix**
Sommerville [17]	1. Introduction 2. Project organization 3. Risk analysis 4. **Hardware and software resource requirements** 5. **Work breakdown** 6. Project schedule 7. Monitoring and reporting mechanisms
Friley [8]	1. Work-breakdown structure 2. Estimation 3. Allocation to schedule 4. Tracking progress 5. Display progress 6. Revisions

the first and the last ones, to consider the mobile games concepts of acquisition, retention and monetization. Table 2 shows the proposed changes in GAMED Methodology to cover a mobile game development process.

5 Conclusion

This paper presented a review of software project plans used in academy and industry with the intent of identify which plan is more suited to mobile games development.

On the quasi-systematic literature review was noticed that there are few, but well established academic software project plans: Sommerville's and Pressman's. However, on the industry side was observed a focus on development of techniques to improve estimations rather than proposing a full project plan.

Specifically on planning mobile games development, it was not found a large amount works. Only two specific methodologies proposed to educational games development appeared on our research: GAMED and IDEALLY, respectively a development and a quality measure methodology.

It was observed that GAMED methodology can be adapted to fit the mobile games development, using agile methodologies' planning techniques and more robust development processes to improve the final product's success.

Table 4: Researched industry planning techniques

Author	Planning task	Technique
Albadarneh et al. [2]	Risk Management	Comparative study of risk management on different agile methodologies
Drury et al. [6]	Decision making obstacles identification	Agile managers and developers interviews
Malhotra and Chung [11]	Impact of agile methodologies	Development of same product using Scrum and IEM
Logue and McDaid [10]	Release Planning	Statistical method for selecting the optimal set of tasks to be developed
Chang [4]	Contingency Resources Management	Use of a resources capability formula to indicate how much and when the resources should be used
Palacios et al. [5]	Team competence	Use of ANN to forecast competence needs
ZurMuehlen and Ho [19]	Risk Management	Use of COSO and CobIT frameworks
Aslan and Balci [3]	Complete Project Planning	GAMED and IDEALLY methodologies

As a future work, it is suggested to apply the proposed adapted version of GAMED methodology in a mobile game development process and compare with traditional methods present in literature. Also, a possible way to measure the mobile game's quality is to use IDEALLY methodology as well.

Also, in order to verify which plan is the most suited for mobile games development, it is suggested to perform a field research with mobile games development companies through a survey.

6 Acknowledgement

This work is partially supported by INES, grant CNPq/465614/2014-0.

References

[1] Portal de peridicos da capes. http://www.periodicos.capes.gov.br/, 2017.

[2] A. Albadarneh, I. Albadarneh, and A. Qusef. Risk management in Agile software development: A comparative study. In *2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies, AEECT 2015*, 2015.

[3] S. Aslan and O. Balci. GAMED: digital educational game development methodology. *SIMULATION*, 91(4):307–319, 2015.

[4] P. H. Chang. Applying resource capability for planning and managing contingency reserves for software and information engineering projects. In *2012 IEEE International Conference on Electro/Information Technology, Indianapolis, IN, USA, May 6-8, 2012*, pages 1–8, 2012.

[5] R. Colomo-Palacios, I. González-Carrasco, J. López-Cuadrado, A. Trigo, and J. Varajao. I-Competere: Using applied intelligence in search of competency gaps in software project managers. *Information Systems Frontiers*, 16(4):607–625, 2014.

[6] M. Drury, K. Conboy, and K. Power. Obstacles to decision making in Agile software development teams. *Journal of Systems and Software*, 85(6):1239–1254, 2012.

[7] V. V. Filho, A. V. M. Moreira, and G. L. Ramalho. Deepening the understanding of mobile game. In *2014 Brazilian Symposium on Computer Games and Digital Entertainment*, pages 183–192, Nov 2014.

[8] D. Frailey. Teaching project planning with no project. In *IEEE 29th Conference on Software Engeneering Education and Training, CSEEandT 2016*, pages 37–45, 2016.

[9] V. Garousi, K. Petersen, and B. Özkan. Challenges and best practices in industry-academia collaborations in software engineering: A systematic literature review. *Information & Software Technology*, 79:106–127, 2016.

[10] K. Logue and K. McDaid. Agile release planning: Dealing with uncertainty in development time and business value. In *Proceedings - Fifteenth IEEE International Conference and Workshops on the Engineering of Computer-Based Systems, ECBS 2008*, pages 437–442, 2008.

[11] R. Malhotra and A. Chug. Comparative analysis of agile methods and iterative enhancement model in assessment of software maintenance. In *Proceedings of the 10th INDIACom; 2016 3rd International Conference on Computing for Sustainable Global Development, INDIACom 2016*, pages 1271–1276, 2016.

[12] E. McDonald. The global games market will reach $108.9 billion in 2017 with mobike taking 42%. https://newzoo.com/insights/articles/the-global-games-market-will-reach-108-9-billion-in-2017-with-mobile-taking-42, apr 2017.

[13] A. B. Nassif, L. F. Capretz, D. Ho, and M. Azzeh. A treeboost model for software effort estimation based on use case points. In *11th International Conference on Machine Learning and Applications, ICMLA, Boca Raton, FL, USA, December 12-15, 2012. Volume 2*, pages 314–319, 2012.

[14] R. Pressman. *Software Engineering: A Practitioner's Approach*. McGraw-Hill, Inc., New York, NY, USA, 7 edition, 2010.

[15] M. R. and S. C. M. Chakradhar Raju M. Software engineering challenges in mobile application development. *International Journal for Research in Applied Science and Engineering Technology (IJRASET)*, 4, 2016.

[16] R. Ramadan and Y. Widyani. Game development life cycle guidelines. In *2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, 2013.

[17] I. Sommerville. *Software Engineering*. Always learning. ADDISON WESLEY Publishing Company Incorporated, 2015.

[18] J. Verner, J. Sampson, and N. Cerpa. What factors lead to software project failure? In *2008 Second International Conference on Research Challenges in Information Science*, pages 71–80, 2008.

[19] M. Zur Muehlen and D.-Y. Ho. Risk management in the BPM lifecycle. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3812 LNCS:454–466, 2005.