EAI Endorsed Transactions
on Serious Games

Research Article EAI.EU

Digital Games' Development Model

Tiago Cardoso[1,2,*], João Sousa[1] and José Barata[1,2]

[1]Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Monte da Caparica – Portugal
[2]Uninova – Centero de Desenvolvimento de Novas Tecnologias, Monte da Caparica – Portugal

Abstract

Nowadays technology follows us everywhere. However, it is often overlooked when dealing with social issues, despite how easily it might improve the life of countless handicapped individuals. In other words, small pieces of software, under the form of games, can help the individual remain focus and go through treatments and therapies effortlessly. But game development is tricky because simply adopting a software development approach is not enough, as it assumes that games as a software can be developed through the commonly used systematic processes, yet these do have specific requirements. This paper proposes a new game development model to successfully deliver a quality serious game, towards social causes, bearing in mind the specific aspects both from games as a special kind of software and from the target players introducing a therapeutic / clinical perspective in the games development model.

Keywords: Digital Games, Model, Software Development, Game Development, Social Causes, Gamification.

Received on 08 October 2017, accepted on 18 November 2017, published on 8 December 2017

doi: 10.4108/_____

1. Introduction

Technology rules over everyday life, anywhere we go we find ourselves surrounded by it. In most, one can find devices embedded with software, capable of performing numerous tasks effortlessly, designed to satisfy our needs and whims so that we profit the most out of it.

If one focus on the digital games industry, we find that it has benefited from this, and according to the Entertainment Software Association (ESA), an organization formed by multiple game producers, the industry profited twenty-three and a half billion dollars. ESA also states that there is, in average, 1.7 gamers in every house in the USA. But although the industry seems to thrive, there's an obvious lack of development guidelines and models, to support the creation of games [1]. Due to this obstacle a new approach was needed.

Digital games are a form of software, since software is regarded as any collection of instructions that allows us to interact with hardware. In this case, digital games are an entertainment-focus software [2]. As such one might think that using a software development model is enough to successfully develop a game, but digital games are not a product of pure engineering and therefore cannot be developed by a systematic and strict process of engineering. Nevertheless, digital games are not pure art either, they are a combination of interleaving disciplines, from art, music, programming, acting and the management or integration of all these [3] - [6]. For that reason, game development requires a specific distinct development process. Nevertheless, software development models still pose as a great resource of

*tomfc@uninova.pt

guidelines, ideas and methodologies, but lack on the specifics of game development.

Moreover, the creation/development of games as a mean of improving the life of people with disabilities is not a new idea, but is a "noble" one that has, so far, counted with some efforts following an ad-hoc approach instead of a systematic one guided by a recognized and well documented creation and development process

The Social Tech Booster (http://stb.uninova.pt) is an example initiative that has been developing distinct "serious" games and systems towards this cause, by teaming up institutions with engineering students in their final year. The students are offered an opportunity to positively impact the life of numerous individuals through their master thesis, while the institutions are granted additional resources, tailored to their needs, effectively changing the life of the ones that really need it [7].

The problem lays on a fact stated previously, the absence of a development model, particularly, one capable of conducting successful design and development of a game for this purpose [3]. Furthermore, the way the STB initiative works provides even more obstacles to the development, since the limitations of both parties involved in the process. Factors like, lack of time from the students who have only 9 months to develop the game, lack of resources from the institutions and the knowledge gap between the parties constitute an issue to overcome.

The goal of this paper is to propose a game development model capable of successfully deliver a good quality game, taking the STB initiative as the baseline.

2. Related Work

2.1. Software Development Models

Software development models are composed by processes, methods and tools that are used to develop software. These models try to describe how to successfully develop software, each with a different approach.

Code-and-Fix Model

This model is the simplest one. For this reason, it is also the fastest and most common one, particularly among new programmers/developers. It consists of three simple tasks:

- Code, where the software is coded.
- Test, related to a stage of tests, usually performed by the development team.
- Fix, where every error found on the previous stage is emended and redone until the client is satisfied.

These three tasks are done by order until the result is satisfactory. There isn't any regard for design nor planning [8].

Waterfall Model (WFM)

The waterfall model was developed in the fifties, as a result of the experience gained during the development of an aerial defense system called SAGE (Semi-Automated Ground Environment) [8]. This model has multiple version but all versions are linear, broke down into stages and take special attention to documentation.

The difference between each version lays on the stages and their amount. The models usually have a requirement stage, a design stage, a development stage, a test stage and a deployment stage, but there can be more or less stages, depending on the type of software to be developed, like a maintenance or evaluation stage [2], [9].

Rapid Prototyping Model (RPM)

The rapid prototyping model appeared in the sixties and "shifted the paradigm". Instead of following a strict, with heavy documentation, staged-based approach like the waterfall model, it focuses more in interacting with the client. To do so, it makes constant use of prototypes. These are a result of a quick planning and design, and are extremely fast to develop, since some features don't have to be fully functional, they just have to send the message across [2], [10]. The prototypes allow the client a wider understanding of the project, therefore enabling him or her to know exactly what it required, what is wrong and how to achieve it and correct it.

Each time a prototype is delivered to the client, the developer waits for feedback. Based on the feedback a new prototype is created and the cycle repeats itself until the client is satisfied with the prototype. The last prototype is then used to create a definitive version of the program [2], [9].

Iterative and Incremental Model (IIM)

The iterative and incremental was created around the same time as the previous one and consequently, shares some of its ideology. The focus remains in interacting with the client, but without the usage of prototypes. Instead the project is divided into smaller sub projects, each consisting of at least a feature or requirement and is treated as a separated project. Since each sub project is regarded as a separated project and the client is required to actively participate in each one, the interaction between developer and client is greater when compared with the waterfall model.

These sub projects are implemented one at a time, by order of relevancy and importance. Each time one subproject is completed, it is added to the remaining finished projects, this way there's always a functional program. For each iteration, the project is incremented with new features and therefore, closer to completion [2], [9].

Spiral Model (SpM)

The spiral model was proposed by Barry W. Boehm and is a combination of the previous two models, the rapid prototyping and the iterative and incremental models, with the waterfall model. From the iterative and incremental model

comes the iterative nature, while from the rapid prototyping model comes the usage of prototypes. That being said, each iteration corresponds to a small waterfall project, during the early iterations a prototype is generated, only on later iteration it's generated a more definitive version of the program. Since each iteration is a waterfall model, we get a well-documented development while still benefiting from the interactions and feedbacks from the clients, due to the usage of prototypes and multiple iterations [8].

Agile Processes Model (APM)

The agile model is the most recent of all models presented. This model follows five basic principles: communication, simplicity, feedback, courage and respect, and is generally divided into four basic activities, planning, design, code and test [12].

During the planning activity, the client shares experiences, stories and wishes, describing the software requirements. Then, the developers and client rate each story and decide which ones have priority. This way a document is generated with all requirements, their development time, release date and development order. This document or plan is flexible so, if during any activity, an alteration to the plan is needed, it is immediately executed and dealt with [11].

The next activity is design, where simplicity is key. In this activity, both developers and clients figure out how to develop the program the simplest way possible and how to test the program [11].

The following activity is code. The first action is to generate the set of scenarios or test units required for the next activity. The next action is to start programming the software in groups, so some can code while the others focus on ensuring the code's quality and refine some design details. All the code and design adjustments must be documented and integrated daily [11]. The last activity is test. This activity consists in executing the set of test units generated previously.

These activities are redone until the client is satisfied or runs out of stories [11].

2.2. Existing Models Comparison

Each model has different characteristic, philosophies and therefore, different pros and cons. In order to properly analyse and compare all software development models, some key parameters must be set in place. These were defined after the study of the same models to be analysed and attempt to illustrate their weak and strong points. The key parameters are:

- Documentation – the attention put into documenting all actions,
- Agility –ability to quickly deal with unexpected obstacles and difficulties,
- Application – easiness in following the model's guidelines and stages
- Interaction – the attention given to the client, so that it can be involved actively in the project,
- Quality – quality of the generated code,
- Speed –time required until achieving a functional or completed project.

From this study and analysis some conclusion where drawn out:

(i) The code-and-fix model simple approach poses huge drawbacks, there's little to no planning, meaning it's almost certain to let details unnoticed or misunderstand what the client wants, although its development times are the shortest of all models.

(ii) The waterfall model sacrifices the simplicity and the speed of the code-and-fix model, in order to obtain a higher quality code and better documentation and therefore, insure an also higher rate of client satisfaction.

(iii) On the other hand, the rapid prototyping model approach focus on interacting with the client and less in documentation, and with the usage of prototypes and multiple iterations it achieves a higher agility without sacrificing so much speed nor simplicity.

(iv) The iterative and incremental approach provides an even higher interaction with the client than the previous model, to insure client satisfaction, compensating the extra wasted time on documentation with a higher quality code.

(v) The Barry Boehm's spiral model increases the documentation generated but sacrifices too much the simplicity, and the interaction with the client.

(vi) Finally, the agile model comes close to perfection, but lacks speed and reduces the quality of the code in order to simplify the whole model. The fact that it doesn't make use of prototypes and relieves heavily on the client, might pose some problems towards finding gaps and error in earlier stages but its agility should solve most of the setbacks.

A rating for each model, product of this analysis can be viewed in the table below (Table 1). With zero being the lowest possible value and five the highest, Table 1 illustrates the performance of the models in each of the key parameters.

	CFM	WFM	RPM	IIM	SpM	APM
Documentation	0	5	3	4	5	4
Interaction	1	2	3	4	3	5
Agility	0	0	4	3	3	5
Speed	5	3	4	3	3	4
Quality	0	3	3	5	5	4
Application	5	0	4	4	3	5
OVERALL	1	3	3	4	4	4.5

3. Challenges in Game Development

According to Pretillo[5] in a recent data gathering, only 16% of project are completed on time and on budget. It's also possible to observe from the data gathered that the problems

with greater occurrences (over 50%), are due to bad project management and poor requirement gathering.

The challenges present in game development are plenty and the results are disastrous, but many have been solved already by the software industry. To solve problems, as with most issues, these must be recognized and understood so that a solution can be found. The main challengers, as stated by Christopher M. Kanode and Hisham M. Haddad, [4] are:

- Diverse Assets – Games are a result of integrating many different expertise. Handling all these poses a challenge as the project grows.
- Diverse Assets – Games are a result of integrating many different expertise. Handling all these poses a challenge as the project grows.
- Scope – Lack of a plausible, viable design and planning, means that the project is constantly increasing as features are added. Evermore the addition of feature without a care thought, may lead to the addition of unrealistic features.
- Publishing – Bringing the game to the industry can be a challenge due to the lack of investment or outdating technologies, since the game industry is a very competitive and fast-paced industry.
- Management – Dealing with so many assets and keeping the project on the right track requires great communication between all members and an excellent oversight.
- New Technologies/Third party – The constant competition of the gaming industry leads to a never-ending development of new exciting technologies. Coping with this can prove hard if the wrong technology is chosen.
- Team Organization – Keeping all team members in check, thinking the same and working to the same common goal is a challenging task.
- Development Process – Choosing the right process can determine the success of the project. Understanding the process is also crucial.

These problems can also be present on the STB initiative and must be taken into account when developing this new model. Although Christopher M. Kanode and Hisham M. Haddad have come up with some solution, those don't fit the STB initiative. However, knowing the existence and importance of these problems is enough to draw conclusions and prepare the model to deal with them.

4. Digital Games' Development Model

This article proposes a new game development model, capable of overcoming the STB above mentioned issues. In other words, developing successfully quality games, through a systematic process instead of an ad-hoc approach, aimed at improving the life of individuals with disabilities, through the partnership of child-care institutions and undergraduate students.

Using the knowledge acquired from the software development models and the common problems with games development while keeping in mind the entities involved in the STB initiative, the following game development models composed of 5 stages is proposed (Image 1).

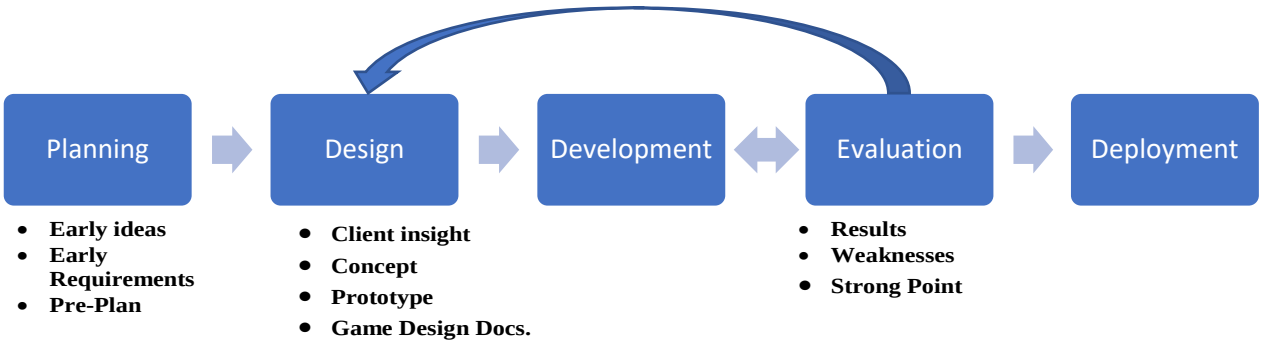

Figure 2. This is a legend. Caption to go below figure

4.1. Planning Stage

The goal of this stage is to prepare for the development of the game. The first step is to create the concept of the game. To do so three main activities are suggested:

(i) Brainstorm, the first activity, is related to the development of the concept of the game, where all the ideas for features must be compiled and a market research done, towards eventually finding similar or related already existing games.

(ii) Meetup, the second activity, is related to meeting the "client". In this case the child care institution. The objective is to allow the student and institution to present all ideas, hopes and features. Documenting all aspects of this meetup is vital

towards the remaining of the game development process.

(iii) Set-up, the last planning activity, relates to the refinement of the game according to the results gathered from the previous activities and the preparation to start the development of the game. Therefore, all features that will be implemented to the game must be thoroughly documented and the schedule of the development specified, starting with the core features. This documentation should be kept in a portfolio to allow for a clear supervision of the whole project.

4.2. Design Stage

The Design stage involves not only the creation and revision of the game design but also the creation of the first prototype. Similar to the previous stage it consists of three main activities:

(iv) Appointment, the first activity, tackles the gap between institution and students, diminishing the knowledge gap and preventing communication failures while also enabling the student to gather vital information. It consists of one or more go-along visits to the institution, where students follow the members of the institution on their normal day and gather information concerning the resources available to the institutions and patients, and understand the limitations of their target audience.

(v) Prototype, the second activity, consists in the development the initial prototype, including all instances of the game and adjustment of the planning made on the previous stage to take into account the addition or removal of features.

(vi) Re-evaluation, the final activity, consists in the creation of documentation with the final design (Game Design Documents), including all features, planning and the initial prototype. It is important to note that all generated documentation from all stages should also be kept in a portfolio.

4.3 Development Stage

The Development stage is the core process which revolves around the creation of the game. This stage applies an iterative approach to game development. In each iteration, a new feature is added, tested and evaluated.

First the feature is developed and added to the already implemented project, then it's tested. The tests, at this stage, are performed by the development team and should only reflect the functional aspect, not the fun aspect.

After performing the tests, the already implemented project is subject to an inspection. The inspection is performed, primarily by a professor and then by the institution. The first inspection serves to determine if the institution is required to clarify some details and either confirm or refute the decisions made along the development, determining if the game is still on "good tracks".

4.4 Evaluation

The evaluation stage aims at assessing if the game performs as designed, both in the functional aspects as in the fun context. This stage consists on a series of simple tests, where both functional and fun aspects are evaluated. The tests are made in three phases, each resulting on evaluation sheets.

The first involves only the development team and members that know all the aspects of the project, the second phase involves members that hardly know or don't know the project at all and lastly, the third phase involves people that fit inside the desired target, members of the child-care institution that don't know the project and other professionals.

4.5 Deployment

This stage is responsible for the "deployment to the masses" and the institution. The development team must opt by a method of deployment easily accessed by the institution taking into account the resources available to them. The deployment should be made in two phases, first targeting the institution and secondly the general public.

5. Validation

The proposed model was used in the development of two games from the STB initiative for validation purposes. Comparing the success in deploying these games with previous games it turned possible to determine the validity of the proposed model.

The games are "Bê-à-Bá" and "Falar pelo Cotovelos". The development of these games lasted 9 months. "Falar pelos Cotovelos" went through only one full cycle, while "Bê-à-Bá" went through two.

"Falar pelos Cotovelos" is a game targeted to children who struggle to say a few words, often because they don't know its meaning (semantics), leading to difficulties in socialization, this condition is usually called autism. During the planning stage, the development team drafted a simple game with extra features besides the main core activity, aimed at turning the game more stimulating and enticing. However, on the appointment activity during the design stage, the child-care institution made it clear that the game should be as simple as possible, with no extra features other than the core gameplay. The idea was to replace and provide a new way of performing a specific therapy. In this case, the simplicity of the game was a must so that this children with already difficulties in understanding basic instruction don't get lost in actions other than the ones that will improve their status, deteriorating all the work done. The result was a very

simplistic prototype which was approved by both child-care institution and professor. The development stage consisted of four iterations. In the first iteration, the core gameplay was developed with only one level of difficulty and few words. The following iterations increased the amount of difficulties and words, all according to specifications provided by the child-care institution and their therapists during evaluations and previous meetups. The evaluations were made, firstly by the professor and students within the STB initiative, then by members of the child-care institution and only after that by patients. The results showed a very simple game, with little to no depth but it suited the institution. The deployment only targeted the institutions since it was a very focused game, but some changes are planned in order to make the game accessible to the public.

"Bê-à-Bá" is a game aimed at aiding in the early stages of learning how to read, targeting children within 3 to 6 years of age. In the planning stage, a direct, simple and enjoyable game was drafted and planned. The prototype created in the design stage appealed, both child-care institutions and professor. The development stage was very similar to the game discussed previously, with four iterations, but during the latter iterations the evaluations made during the meetups showed some signs associated to a lackluster experience. The evaluation stage clearly pointed it out, making it obvious that a new cycle was required. Therefore, the game went through a new design, development and evaluation stages. The design stage needed to tackle the blankness of the game. The problem laid in the lack of meaning and rewards for a good performance, so a new feature was added. This feature made it possible to unlock content every time the player beats a new level. The idea was simple but ideal when dealing with children. To develop this feature, it was necessary to iterate two times during the development stage, to insure the final result was functional and worked as intended. This time, the evaluation stage provided much more satisfying results and the game is currently in deployment both to public as to the child-care institution.
The application of the proposed model to "Bê-à-Bá" and "Falar pelo Cotovelos" showed that it can successfully provide and create good quality games towards the STB initiative, namely according to the specialist of the involved child-care institution.

References

[1] ESA, Essential facts about computer and video game industry, vol. 2016, pp. 1–3 (2016).
[2] R. S. Pressman, Software Engineering A Practitioner's Approach 7th Ed - Roger S. Pressman (2009)
[3] Ramadan, R., Widyan. Y., Game Development Life Cycle Guidelines, (2013)
[4] Kanode, Christopher M., Haddad, Hisham M., Software Engineering Challenges in Game Development (2009)
[5] Petrillo, F., Pimenta, M., Trindade, F., What Went Wrong? A Survey of Problems in Game Development (2009)
[6] Adams, E., Fundamentals of Game Design, 2nd Ed. (2010)
[7] Cardoso, T., Santos, V., Santos, C., Barata, J., Transferência de Tecnologia para Causas Sociais através de VideoJogos (2015)
[8] Boehm, B., A Spiral Model of Software Development, IEEE Comput., no. 1, (1988)
[9] Ghezzi, C., Jazayeri, M., Mandrioli, D., Fundamentals of Software Engineering, vol. 5961 (2002)
[10] Tripp, S. D., Bichelmeyer, B., Rapid prototyping: An alternative instructional design strategy, Educ. Technol. Res. Dev., vol. 38, no. 1, p. 31-44 (1990).
[11] Beck, K., Extreme Programming Explained, 2nd Edition (2004)
[12] Martin, R. C., "Agile Software Development, Principles, Patterns, and Practices," Book. p. 529 (2002)
[13] Cardoso, T., Santos, V., Santos, C., Barata, J., Games Social Tech Booster, International Conference on Serious Games, Interaction, and Simulation, pp. 119-126 (2015)