International Journal of Advance Research in Science and Engineering
Vol. No.6, Issue No. 06, June 2017
www.ijarse.com

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

GDLC: A Software Engineering Approach in Game Development

Raj Patel[1], Prof. Kruti Lavingia[2], Prof. Rushabh Shah[3], Prof. Rutvi Shah[4]

[1,2,3]Computer Engineering Department,

Institute of Technology Nirma University, Ahmedabad

[4]Shri Chimanbhai Patel Institute of Computer Applications, Ahmedabad

ABSTRACT

Software Engineering is the field of computer science which is applied during software product development and game development is not an exception. Game development is highly multidisciplinary field of Information technology because many developers (Graphics Artists, Music Artists, Back end programmers, Game play analyst etc.) are involved in different stages of game development life cycle. However, it is seen from the past experiences that applying traditional software development life cycle (SDLC) does not yields fruitful results in the end product. To resolve this issue a more domain specific approach is followed called game development life cycle (GDLC). This paper briefs about game development life cycle and addresses its pros and cons.

Keywords: *Console Games, Game Development, Game Design, Game Play, GDLC*

I. INTRODUCTION

Technological advancement in the past decade has revolutionized each and every domain of Information technology industry and has made a great impact on the Game industry specifically. Now it is no more true that video games are played only on game consoles, but with the advent of internet, mobile devices, high end graphics cards, powerful AI it is possible to develop high end games for each and every platform with a great ease. Today each and every age group of people tend to play games of differing genres, so it is extremely important to keep the user's domain, age, geographical location, expertise etc. in mind while designing/developing games. As games generally include Art, Music, Gestures, Control System, Specific Hardware, Logic Programming, User interface (UI) and User experience (UX) and Robust Artificial Intelligence (AI) so it requires developers from different domains to work together in order deliver an efficient and user centric games. Though software development life cycle (SDLC) provides a systematic approach to software product development in general, but the diverse nature of game development restricts us to use software development life cycle as an underlying model. Game development life cycle comes at rescue in this condition and it is well suited for both Independent (indie) game developers as well as for well-known game development companies like Nintendo, EA-Sports and many more [1].

II. KEY TERMS AND DEFINITIONS

A. Game Theory

Game theory in general a field of mathematics which involves the concept of decision making by a player to secure the best end result in his favor, Concept of Zero-sum games is applied which means gain to player 1 results a total loss to player 2. Game theory is involved in almost all the games ranging from action games, role playing games (RPG), strategic games like chess, checkers and many more. It is the combined task of game theorist and back end programmers to provide an efficient implementation to these algorithms and AI logics in order to get an active response in the game play without any unwanted delay [2].

B. UI/UX

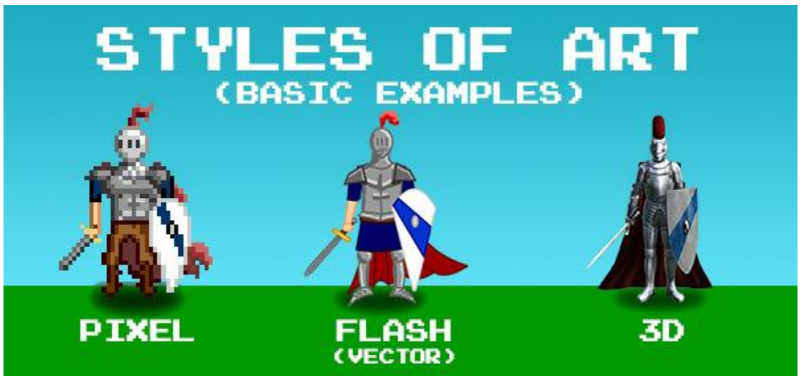

Figure 1: Example of Vector Image.

User Interface Design (UI) and User Experience Design (UX) is the first and foremost thing that the end user will interact with while playing the game, so it is extremely important to keep it user focused. Different people involved in UI/UX design are Game design artist, sound and music artist, front end developers, script writers and animation artists. They all need to coordinate and deliver a good design which must be fast enough to render on various devices with differing screen resolution and graphics capabilities and must also synchronize with the back end code implementation for smooth performance. It is advised to use vectors (Figure 1) to make sprits, so that they can be magnified to a great extent without having any blurring.

III. GAME DEVELOPMENT LIFE CYCLE

There are various game development life cycles followed by different companies to meet their requirements, but in general GDLC are iterative in nature and they focus on the quality criteria as their prime focus. There is a great area of intersection between SDLC and GDLC, so the benefits (planning, timeliness, deliverables, testing etc.) provided by SDLC are implicitly inherited by GDLC models.

The described GDLC is divided into six different phases:

A. Initiation

It is the process which involves drafting a very high level overview of the game concept. Following are the major sub processes involved:

1. Building a Story and Script

A general theme around which the game will revolve is defined here. The theme is targeted to audience of a specific age group. Then leading and side characters are decided after that a plot is defined for every state of the game. Here the script writers plays major role. The plots are then sub divided to fit into the script and it is made sure that there is a continuous of the game from one level to another.

2. Feasibility Study

Feasibility study is carried out to check whether it is possible to develop the defined game above within the given constraints. Here all the requirements are gathered such as scope, platforms, interfaces, profitability etc. Then the requirement analysis is done to find out the pricing terms, technical support, legal issues and cultural barriers etc. Capabilities of the development team and other artists are also considered before drafting the feasibility report. The feasibility report consists of technical specification and management summary. The final decision whether to undertake the project is taken thereafter in a general meeting.

B. Pre-production Phase

Pre-production is the very first phase of the production cycle. It involves mainly game prototyping and game design. A working prototype of the game is created using the design concept which involves the core features of the game such as fun elements, layouts, texture, and animation. Various tools used to create design elements are Adobe Photoshop, Adobe after effects, Coral Draw etc.

Following are the design and sound aspects:

1. Character Designing

It is important to design a strong and convincing character that is suitable for the given context and looks visually appealing. Expressions and posture should also match with the theme of the game especially role playing games.

Figure 2: Character design software

2. Set and Props designing

Set and Props design includes the material other than the character which should be displayed on the screen at a particular instant. These should match the time and place around which the game story is modeled. E.g. one should not show electronic fans in a room if the story of the game is depicting a situation in 16th century.

3. Background designing

It is important to keep background as simple as possible so that the attention of the player remains on the game play without any distraction. A good background design will convey the context very easily and may sometimes show a 3D effect on a 2D screen.

4. Texturing

It is the process of adding minute details and providing the surface texture to the characters, props and sets so that they look realistic.

5. Animation

Animation is the process of giving a sequence of action to the characters of the game. Since the flow of the game is not linear mesh methodology is used to give the animation, with the help of mesh animation we can show the deformations like facial expressions, rib movements, ripple effects etc. Animation is further classified as motion capture animation and key frame animation.

6. Music and Sound

Apart from animation, music and tones play a great role in enhancing the overall game play. Sound is used to exaggerate the effects shown in the motion. Examples include punch sound, bullet firing sound, background music etc. In some games like first person shooter games footsteps of the enemy helps in making good decisions. Examples of sound making software are FMOD, Audio kinetic and Fabric etc.

Figure 3: Wall texture design

Pre-production phase is itself an iterative process and after its each iteration its elements are documented in the Game Design Document (GDD). When the pre-production phase ends a promotional demo is carried out in which used to attract the potential customers, since the complete game is not ready so videos are made to assist the game play shown in the prototype.

C. Production Phase

Production phase comes when the prototype is already designed and approved. It is the process where the actual back end as well as front ends programming is done, game assets are created and are merged together. Here the previously tested interfaces are used (reusability) and if they are not available then they created and added to the existing repository of the interfaces. Games are actually programmed in a Game Engine which is general purpose game development tools to assist rapid game development. The development team first decides which programming language to use in order to implement the code, as efficiency is the main concern at the backend side so C++ is preferred at backend development. A suitable game engine is chosen from a pool of game engines like unreal game engine, Unity 3D, Cry Engine, Cube etc. One more issue addressed here is of the requirement of the internet connectivity while the game is being played. If it is the case then proper cloud infrastructure is developed to store the user’s game data so that it can be retrieved later. Database is also designed to store the user’s data locally and the concepts of normalization are also applied here. Production phase is also an iterative process in which after each iteration the formal details like game balancing is refined by adding new features, resolving bugs and improving overall performance. Apart from that game is constantly polished to make it more challenging and fun elements are enhanced.

D. Testing

Testing is an integral part of the software development life cycle and it becomes of utmost importance when it comes to game development. As it has shown from past experiences that games are the only software's in which major bug fixes are reported. Due to complex nature of games a logical error may sometime look legitimate as the game would run perfectly but when played with intense care it would affect the game play. Game testing is performed in a well-structured manner for each and every games ranging from small games like packman to a multiplayer games like counter-strike.

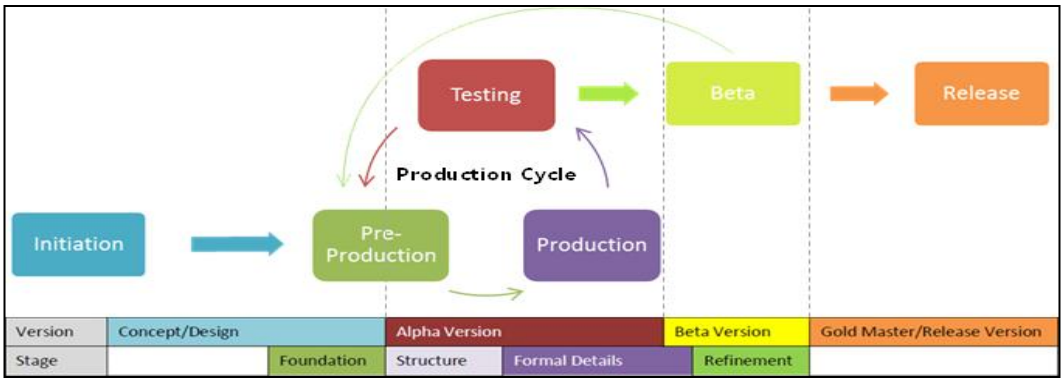

Figure 4: Game Development Life Cycle

Basic Game testing structure:

1. Pre-production: It is true that there is no actual testing when the game development begins but assets, scopes, and games design are continuously iterated in the pre-production phase which requires testing and needs to be corrected at the early stage itself. It is seen that proper documentation testing (game plan, game design and test plan) can help in solving the problems in the early stages itself, if these tasks are delayed then it may sometime lead to project failure itself.
2. Alpha testing: Alpha testing is generally carried out by the employees of the company itself. It is done to identify the bugs before releasing it to the end users. During the process of alpha testing the software developed by the team is merged and the game play is further improved. Here the game is played from start to end and if a bug is found it is resolved and the tester again starts to play from the beginning to make sure the previously resolved bug have not introduced any other bug in the game.
3. Beta testing: Beta testing is carried out after the end of alpha testing. It is usually performed by the tester outside the development team, a few group of end users are given a feature complete game for beta testing and the feedback is noted to resolve the bugs. The main feature of beta testing is that the tester can exercise every part of the game as the game play, music, AI are all completely built.
4. Gold testing: After the beta testing ends the game becomes almost ready to be shipped to the actual market, but as an extra preventive measure a final rigorous testing is carried out on the beta build to find out any leftover bugs like system hangs, function failures and crashes. This testing also reveals if there are any performance issues still remaining. If any bugs are found in this testing then the final release may be delayed as resolving bugs in the final product is very time consuming. Passing this phase ensures that 90% bugs are resolved.
5. Post-release: In many situations due to some hard deadlines it is not possible to exhaustively test the game. So final version of the game is sometimes released which may contain one or two minor bugs. So the

development team then solves the bugs and releases the bug fixes, updates and patches for the game that are installed by the end users. Each new patch must be compatible with the older version of the game.

Game testing techniques [4]:

Test Flow diagrams: They are the models which depicts the game behavior from the user's perspective. Here all the paths of test flow diagrams are explored in order to find out the unexpected behaviors. Since they are graphical in nature it is easy to test, analyze and provide the feedback to the developers.

1. Play testing: This technique is used to find out the effectiveness of the nonfunctional features like fun factor, addictiveness, difficulty level, game play and performance. The results of this testing reveals that whether the game would "hit the market well or not".
2. Combinatorial testing: Exhaustively testing for all the combination of game inputs, game variables, events, configurations and options will surely detect all the bugs which are present in the game but at the same time it would create a very huge set of test cases and it would be impractical and economically infeasible to test for all. The main idea in combinatorial testing is that not all of the game parameters interact with each other so we can save time by considering only those pairs of parameters which interact with each other and check if there is any bug created by their interaction. The tests here can be of two types homogenous and heterogeneous. So this increases the test execution efficiency to a great extent and reveals the bugs in the early stage itself.

IV. CONCLUSION

As game development is a highly multidisciplinary field of Information Technology so software engineering plays a crucial role in assisting its development. Game development involves five to seven teams each of size ten with a varying background so it is very important to adhere to Game Development Life Cycle in order to deliver the product on time. The most time consuming process of game development is game testing because a game involves interaction of many objects in an interleaved fashion. Currently manual and traditional testing methods are applied to test for the bugs. Research is going on in the field of testing automation and it is still not fully matured.

REFERENCES

[1] Yani Widyani, Rido Ramadan "Game Development Life Cycle Guidelines" ICACSIS 2013

[2] Avinash Dixit and Barry Nalebuff "Game Theory" CEE 2nd Edition 2008

[3] K.Subhash Babu and R.Maruthi "Lifecycle for Game Development to Ensure Enhanced Productivity" IJIRCCE Vol.1, Issue 8, October 2013

[4] Claudio Redavid, Adil Farid "An Overview of Game Testing Techniques"