Adaptation Atomic Design Method for Rapid Game Development Model

Rickman Roedavan 1,*, Agus Pratondo 1, Bambang Pudjoatmodjo 2, Yahdi Siradj 1

1 Multimedia Engineering Technology, School of Applied Science, Telkom University, Indonesia
2 Centre for Advanced Computing Technology (C-ACT), Faculty of Information and Communication Technology, Universiti Teknikal Malaysia

rikman@telkomuniversity.ac.id, agus@telkomuniversity.ac.id, p031810011@student.utem.edu.my, yahdininformatika@telkomuniversity.ac.id

ARTICLE INFO
Received February 16th, 2021
Revised March 16th, 2021
Accepted March 17th, 2021
Available online May 8th, 2021

Abstract
Prototyping a game is a process to build the core game mechanics for use in the final game. This process is essential because it can save cost, time and reduce the potential for errors. There is a lot of study and research on the Game Development Life Cycle (GDLC). Some existing GDLC’s were developed based on the waterfall model, while others were developed based on prototyping models. However, most of these models are not considered the cost and time factor. This research proposes a new Rapid Game Development (RGD) model, adapted from our previous work in developing the Game Mechanic Framework for Unity Game Engine. The experiment result showed that this method could be used to create a game faster while keeping the budget in mind during the development process.

* Corresponding author at:
School of Applied Science, Telkom University
Jl. Telekomunikasi No. 1, Terusan Buah Batu, Bandung, 40257
Indonesia
rikman@telkomuniversity.ac.id

ORCID ID:
• First Author: 0000-0003-3169-2663
• Second Author: 0000-0002-6976-7459

https://doi.org/10.25124/ijait.v4i02.3658
Paper_reg_number IJAIT000040210 2020 © The Authors. Published by School of Applied Science, Telkom University. This is an open access article under the CC BY-NC 4.0 license (https://creativecommons.org/licenses/by-nc/4.0/)
1. Introduction

Games are interactive multimedia products that are built by combining various disciplines. In addition to entertainment, games can also be developed as simulation products, military training, education, and medical training tools [1][2][3]. As a product that looks easy and fun, the game development process is complex [4].

There are many research and studies on the Game Development Life Cycle (GDLC) [5][6][7]. Most GDLCs developed based on the Waterfall model in the Software Development Life Cycle (SDLC). Most models only focus on developing the game without considering the cost, time factor, and error potential in the development process.

In reality, the waterfall model of development is slow, rigid, and difficult to adapt to change. Statistics show that 80% of games released on the market by indie studios fail [8]. Reducing failure can be done by detecting errors earlier in the development process using rapid prototyping methods.

Rapid Application Development (RAD) is a method in software development [9][10], which was popular in the 1990s [11][12]. However, this method is still relevant to us today. This method can produce digital products quickly, speed up the evaluation process by the user, and reduce errors that arise during the development [13][14].

This research continues our previous research results, namely developing the Game Mechanic Framework on the Unity game engine [15]. This research aims to produce a new Rapid Game Development (RGD) model, speeding up the game development process. Most importantly, this model can guide developers to consider the costs required during the development process.

2. Development Model Analysis

2.1. Game Development Life Cycle

Most of the GDLC proposed by practitioners and researchers have a similar development pattern [5][6]. Figure 1 shows that, in general, all GDLC models can be divided into three main phases: pre-production, production, and post-production. When viewed from the production process, GDLC can be divided into Linear GDLC and Iterative GDLC. In Linear GDLC, each phase is resolved sequentially for each phase. While in Iterative GDLC, the development phase is carried out through an iteration process before entering the post-production phase.

![Figure 1 Game Development Life Cycle Pattern](image-url)
Linear GDLC was adapted from the Waterfall method [7], while the Iterative GDLC was adapted from the prototyping method. Iterative GDLC is better than Linear GDLC because it can build products quickly and reduce errors in the development process.

The most popular GDLC among academics is the Rido Ramadan version, as shown in Figure 2. This GDLC has timeline elements and its output version but not considering cost elements. While in real game development, the cost is an important element that should not be ignored in the development process.

Some indie developers or game studios tend to develop their versions of GDLC to solve a cost problem. One example is the Agate Studio version shown in Figure 3. This GDLC adds evaluation and monitoring elements by relevant stakeholders at each stage of its development. [6]. This element is added to prevent funds to prospective games that are deemed to have no potential for success.

At a glance, this element is perfect because it can control the budget and prevent loss of game funding. But on the other hand, this model has a high probability of termination of the game candidate in the middle of the development process.

This causes the development process can be very long and immeasurable. However, this problem can be overcome by using a generic asset or game template in development. Using a reusable object in the game development process is key to create a prototype faster and adapted to meet stakeholder needs.

2.2. Rapid Application Development

RAD method is best used for making project-based applications with well-defined requirements. Figure 4 shows the most popular version of RAD, which is divided into six main stages. Namely, Analysis and Quick Design, Develop,
Demonstrate Refine, Testing, and Development. The Develop, Demonstrate, and Refine phases become the core that makes up the prototyping iteration.

The RAD implementation is very diverse in the real development process; some software company combines its method to create a hybrid development model. Figure 5 shows that the RAD pattern consists of three main parts: Modeling, Prototyping, and Deployment.

RAD purposes are developing high-quality software systems more quickly. There are at least five essential success factors to achieve this, namely determining Application Purposes, understanding of Technology knowledge, easy access to Asset / Materials, ability to translate Product Complexity, and Estimation of cost and time [16][17][18].

### 2.3. Atomic Design Method

Atomic Design is a method for combining different stages of application development hierarchically [19]. This model was originally developed to build a website using the atomic theory approach in natural science. The atomic design has five stages of development: Atom, Molecule, Organism, Template, and Pages, as shown in Figure 6.

Atoms are explained as the smallest part of website development, like HTML tags. Molecules are a collection of HTML tags that will form a user interface group with one specific function. Organisms mean larger collections of molecules with
more complex functions. Header, Footer, and Content are terms that are the most suitable analogy for Organisms. The collection of Organisms into templates can accelerate the development process to create a web page as a final result.

3. The Proposed Model

The model we propose combines the Ramadan, Agate, and other popular GDLC models [5][6][7]. We use the Atomic Design Method's advantages to map all game development processes according to their classification. We also add key success factors for the RAD method and some important game development process elements.

Table 1 Game Development Atom

<table>
<thead>
<tr>
<th>Development Phase</th>
<th>Pre-Production</th>
<th>Production</th>
<th>Post-Production</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development Activity</td>
<td>Initiation</td>
<td>Prototype</td>
<td>Creation</td>
</tr>
<tr>
<td>Pre-Production</td>
<td>Pre-Production</td>
<td>Pre-Production</td>
<td>Develop</td>
</tr>
<tr>
<td>Production</td>
<td>Production</td>
<td>Production</td>
<td>Evaluate</td>
</tr>
<tr>
<td>Testing</td>
<td>Beta</td>
<td>Product-Launch</td>
<td>Test</td>
</tr>
<tr>
<td>Beta</td>
<td>Live</td>
<td>Live-Ops</td>
<td>Review Release</td>
</tr>
<tr>
<td>Release</td>
<td>-</td>
<td>-</td>
<td>Release</td>
</tr>
</tbody>
</table>

Development Consideration

<table>
<thead>
<tr>
<th>Consideration 1</th>
<th>Consideration 2</th>
<th>Consideration 3</th>
<th>Consideration 4</th>
<th>Consideration 5</th>
<th>Consideration 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project-Based</td>
<td>Linear</td>
<td>User Research</td>
<td>Generic Assets</td>
<td>Application Purposes</td>
<td>Testing</td>
</tr>
<tr>
<td>Product-Based</td>
<td>Iterative</td>
<td>Feasibility Study</td>
<td>Genuine Assets</td>
<td>Technology Knowledge</td>
<td>Deployment</td>
</tr>
<tr>
<td>Cost Estimation</td>
<td>RAD</td>
<td>Pitching</td>
<td>Prefabrication</td>
<td>Asset/Material Access</td>
<td>Evaluation</td>
</tr>
<tr>
<td>Time Estimation</td>
<td>Prototype</td>
<td>Approval</td>
<td>Blueprint</td>
<td>Product Complexity</td>
<td>User Feedback</td>
</tr>
</tbody>
</table>

Development Output

|---------------------|-----------|---------------|--------------|-------------|-----------------|--------------|

Table 1 contains the results of solving the complexity of game development and several supporting factors. The results consist of 4 groups of organisms, namely the development phase, development activity, development consideration, and development output.

Each organism consists of several molecules derived from pre-existing models. And each molecule consists of a collection of cells, which refers to the main activity of game development or game terminology. We combine all the cells obtained to produce a new Rapid Game Development (RGD) Model, as shown in Figure 7.

In general, the RGD model consists of 3 main phases. Each phase consists of several activities that are carried out by considering the success factors in the rapid prototyping model and cost development.
1. **Pre-Production.** This phase has one main activity, namely Quick Ideation, which results in the game's initial design concept. This activity must also be able to determine whether the game to be built is product-based or project-based. Understanding this will significantly affect the research time and engineering costs that will be used in further development.

2. **Production.** This phase consists of three activities, which are the main activities in the entire rapid development process, namely Prefabrication Development, Blueprint Integration, and Prototype Testing.
   
   a. **Prefabrication Development.** This activity aims to build a game foundation using templates, frameworks, or independent assets that already have basic functions. Some examples of a basic common function are character movement, finger-gesture detection, and an inventory system. Each game engine has a different workflow. So it is crucial to understand the technology used and the functions in it. This activity also needs to determine which parts of the game require original assets and which parts can use generic assets to speed up the development process.

   b. **Blueprint Integration.** This activity aims to modify and integrate all existing assets into a functioning game mechanic. Using various types of generic assets, script, or game element prefabrication has its problems in practice. The development team must understand how each asset works and get them to communicate with each other. This process is time-consuming and costly. But it's still a lot shorter than creating the entire asset from scratch.

   c. **Prototype Testing.** This activity aims to produce a game prototype that functions mechanically and already has an interesting initial visualization.

Based on indie game developers' experience, game prototypes should be produced within 1-2 Weeks [20]. Testing the game prototype can be used...
as evaluation material for improvements in the next development stage iteration.


   a. Beta Version Release. This activity produces beta game products that are fully functional. This beta product is released for testing to players or tested with clients for suggestions and feedback on a project-based basis. It depends on the complexity of the game being built, but in general, the feedback needed for this phase is essential. So if needed, independent user testing must be carried out, and that requires a budget.

   b. Final Version Release. This activity is the last activity whose output is the final product game. For online-based games or requiring an operating server, some costs must be considered. Even though it has entered the final phase, it does not mean that the released game will be free from bugs. So it takes a reserve of overhead funds for making game patches if needed.

4. Experimental Results

We tested this model's use in a Telkom University Multimedia Labs research team consisting of only 3-4 people. This team must work on three-game projects in parallel, which came from 3 different clients. Games that must be made have varying deadlines between 3-6 months, and limited development costs range from $2.000 - $5.000.

The first game that must be made is a 3D cooking simulation game intended as a learning medium in the hospitality study program. The second game is an educational game for elementary school chemistry learning. And the third game is a 2D mobile game for introducing a positive culture for a private communication company.

The result was that the team managed to complete all three games in parallel with limited time and budget. There is a lot of potential for improvement, but in general, all clients love the games they make. Table 2 shows some screenshots of the games that have been created.

<table>
<thead>
<tr>
<th>Table 2 Screenshot of The Game</th>
</tr>
</thead>
<tbody>
<tr>
<td>Game ScreenShot</td>
</tr>
<tr>
<td><img src="image1" alt="Game ScreenShot 1" /> <img src="image2" alt="Game ScreenShot 2" /></td>
</tr>
</tbody>
</table>

1

2
Measurement of user satisfaction results is tested by 30-40 respondents using the User Experience Questionnaire (UEQ), which consists of a 26 item questionnaire [21][22]. Figure 8-10 shows the overview UEQ results for all the games.
UEQ encapsulates all user experience points into six major groups: Attractiveness, Perspicuity, Efficiency, Dependability, Stimulation, and Novelty. The UEQ score scale ranges between -2 (horribly bad) and +2 (extremely good). In general, all the games that have been developed have scores ranging above 0.5 and 2. The highest score lies in the points of Attractiveness and Stimulation. This is because the game built has a good visual aspect and is equipped with a game mechanic that is exciting, interesting, and challenging for players.

5. Conclusions

The proposed RGD consists of 3 main phases consisting of 6 main activities. Several factors are considered for carrying out each activity, especially those related to development time and cost.

This model results from a combination of the GDLC model from academics and practitioners’ points of view. In general, this model will depend on the mastery of technology or the game engine used. The availability of various generic or prefabricated assets is also needed to speed up the development process.

The test results show that this method’s use can significantly accelerate the game development process, even within a limited time and budget. However, further research is still needed to verify this model’s validation to create a more complex game and variety mechanic.

Bibliography


