SIMULATING TURN-BASED RPG USING SYSTEM DYNAMICS AS A BALANCING TOOL: AUTOMATING PLAY TESTING

[1]ERVINA WIJAYA, [2]ARYA ANTAPUTRA, [3]YUSUF ISLAM

Human Computer Interaction Surya University Tangerang, Indonesia
E-mail: [1]ervina.wijaya.su@gmail.com

Abstract- Designing game balancing by game designers are tedious work where they need to design the statistics using spreadsheet, and playtest them manually in many iterations. Finished balancing will be the best state of the game to deliver as intended by its game designer. In order to reach the said state, game designer must spend a good deal of time doing playtesting. Therefore, this paper proposes a game balancing tool which functions to automate playtesting by simulating model of battle RPG using utilized system dynamics. The purpose is to help game designer cuts playtesting time to collect insights on how their current balancing works. This paper explores the possibility of creating and using system dynamics to model RPG, especially battles, to be used as game balancing tool to help building strategies in design direction.

Keywords- playtesting, simulation, system dynamics model, turn-based RPG, game balancing tool

I. INTRODUCTION

Role playing games (RPG) incorporates various numbers inside their game mechanics. The relationships between players and enemies are connected by battle mechanism. As game designers need to design balancing to those numbers to make every game from scratch. Ideally, a game is considered balanced when it does drive most of target players to quit before the game reaches its final. The core aim of game design is to create a fun entertainment/game for players [1].
The complexity of considering how the numbers affect each other often hinders game development. Currently there exist several enemy/monster statistics generators to help the designing, but they have limitations. Once being merged into a whole game, playtesting session is still needed to test the balancing. A research about game balancing tool found several feature requirements by asking game designers. One of them was the ability to do live-edit playtesting that concerned to avoid time wasting by stopping and editing back and forth [2].

In the other hand, there is a method that is used to model, simulate, analyze complex real-world systems called system dynamics [3]. As game is also being able to be perceived as a system, this paper's objective is to fill the gap by using system dynamics as the backbone and the demand of playtesting automation. This research starts with literature review of RPG balancing includingnumerical relationship, progression, and system dynamics. Then, continuing conceptualization of simulation model, which finally the simulation will be run based on a scenario in imaginary case study.

II. LITERATURE REVIEW

A. Numerical Relationships

Game is consisted of numbers and mechanic that governs how each number relates to the others. RPG is one of the oldest genre of game, examples of RPGs are Chrono Trigger, Final Fantasy, Pokemon, etc. The key elements of RPGs are where the player as main character who is either a hero or a pokemon trainer, going on an adventure to complete quests. During the adventure, player will be faced by enemies and battles where player needs to beat the enemies before getting beaten, win the battle, and then gradually get experience points to gain more levels as they progress through the game. The relationship between experience points needed each level is called progression relationship. The other is tradeoff relationship where number of certain item is traded with another item. The simple example is trading gold with potion items. These progression and tradeoff relationships affect on how balanced a game is when player is interacting with game components [4].

The basic numbers that are part of most RPGs are: (a) level, (b) exp (experience point), (c) attack, (d) defend, (e) randomness. The relationships between two number elements can be represented as curve. There are several types of curve: identity, linear, exponential, logarithmic, triangular, or even custom crafted curves.

B. Randomness

Random is needed in not only RPG, but most game in order to create a realistic point of view. For example, in reality during a battle, damage produced by a character cannot be always constant; even robots don't produce 100% same output all time. Constant relationships will make the game feels static and everything becomes predictable; which will become boring for players. That is why game designers like to implement randomness in their game, but in such a way to avoid wild randomness.
Several ways to implement randomness in calculation is by using dice roll or using normal/uniform distribution. When designing randomness in game,

designer needs to understand the probabilities of winning and losing the game [5].

C. Enemy Random Encounter v.s. StaticEncounter

Single player RPGs usually need bot monsters called mob to fight in forest/dungeon to help gain level, unlike multiplayer RPGs. Currently, there are two possibilities of designing enemy encounter type for a game [6]:

1. **Random encounter:** when player is walking through a dungeon, they will encounter invisible enemy which are spawned by counting how many steps are taken.
2. **Static encounter:** Another one usually this type of encounter consists of visible enemies where player can see the enemy position in the map. Battle will be engaged when player touches the enemy, while also able to walk past the enemy by avoiding contact. This method will require designer to position enemy over the map.

D. RPG Maker

RPG Maker is game engine specific for RPG development. RPG Maker currently has several series for witch the popular versions now are RPG Maker VX, RPG Maker VX Ace, and RPG Maker MV. RPG Maker has created hundreds of games; both traditional RPG and nontraditional such as visual novel horror RPG genre [7].

RPG Maker has features which are designed specifically toward the ease of RPG development, and that makes it a good source to reflect on how RPG structures are usually and generally built.

RPG Maker implements elements of RPG into:

1. **Maps:** filled with items, NPCs, and enemies in a specific position in a space.
2. **Actors:** actors are basically playable characters. They have statistic template consisted of 9 elements: EXP curve, Max HP, Max MP, Attack, Defend, M.Attack, M.Defend, Agility, and Luck. These elements are saved as element-to-level curve, which there's a feature to generate the curve based on growth speed.
3. **Enemies:** a database of enemy troops. Each enemy also has 9 elements actors have, in addition to EXP Drop and behaviors. These behaviors represent enemy's actions and its probability to act.
4. **Items:**a database of items accessible during battle. Items includes potions, having amount of restoration point.
5. **Skills:** a database of skills, including physical attack. RPG maker uses a default damage formula:

$$\text{Damage} = \text{Base_Value} + a.atk * (0.04 * \text{Physical}) + a.mat * (0.04 * \text{Magical}) - b.def * (0.02 * \text{Physical}) - b.mdf * (0.02 * \text{Magical}) \quad (1)$$

III. METHODOLOGY

A. Case Study

The case study used to design this tool is using a scenario of persona of a game designer for RPG who usually works on games using RPG Maker.

At early game development, a game designer wants to design her RPG to have this difficulty of having 2-3 turn to take out enemy at first forest, then raised until 5-7 turn for entering new dungeon of more powerful enemy. As player travels through dungeon and gaining more level, when they are out of dungeon (after beating 20 enemies), player can re-enter dungeon and fight with same set of monsters which will be faster; decreasing turn count to 3-5.

Game designer looks for feedback data of whether the enemies are balanced enough for player in each state battle to make further amendments for enemies' statistics to reach the balancing she wants.

B. System Dynamics

System dynamics is a methodology to analyze structure and how a system behaves developed by MIT [8]. By dividing a system into variables and model their relationships, the audience will be able to understand the behavior of certain variables. Relationships are modelled by using mathematical equation involving related variables by analyzing the referenced system itself.

System dynamics models' purposes is to: (a) teach audience how a system works and learning its behaviors, (b) creating a predictive simulation to test out policy makings. System dynamics could be represented as qualitative and quantitative modeling. Qualitative modeling takes form of causal loop diagram, while quantitative modeling uses stock-flow diagram [9]. During development, it is recommended to use stock-flow diagram in the formulation stage [8].

System dynamics is used to model RPG battle simulation in this paper because game is also a system that has related variables.

IV. IMPLEMENTATION AND EXPERIMENTATION

After literature studies to understand balancing in RPG, developing the system dynamics model was divided into 4 stages. There was also studies on how a turn-based RPG is usually made in order to create a balancing tool that could work with wide variety of turn-based RPG. The studies and observations collected will help designing game rules and elements' relationship. Followed stage is developing the system dynamics model using framework consisted of: (a) problem identification (b) conceptualization (c) formulation (d) testing. All stages are done in chronological order using iteration development process. Each iteration's feedbacks were

used to improve the model in the next iteration. Few conceptualization points will be explained in model's discussion.

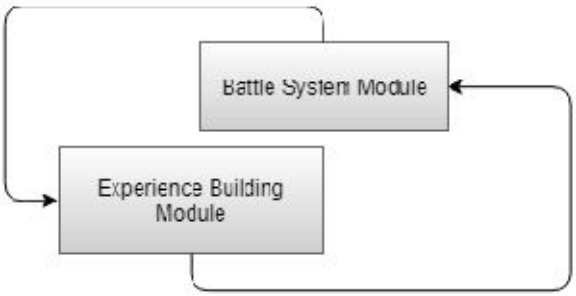


Fig. 1. Model framework.

A. System Dynamics Model

1) Model Framework

System dynamics model was developed using stock and flow diagram developed using Vensim PLE by Ventana Software. As shown in Fig. 1the result model is composed of two main modules: battle system module and experience building module.

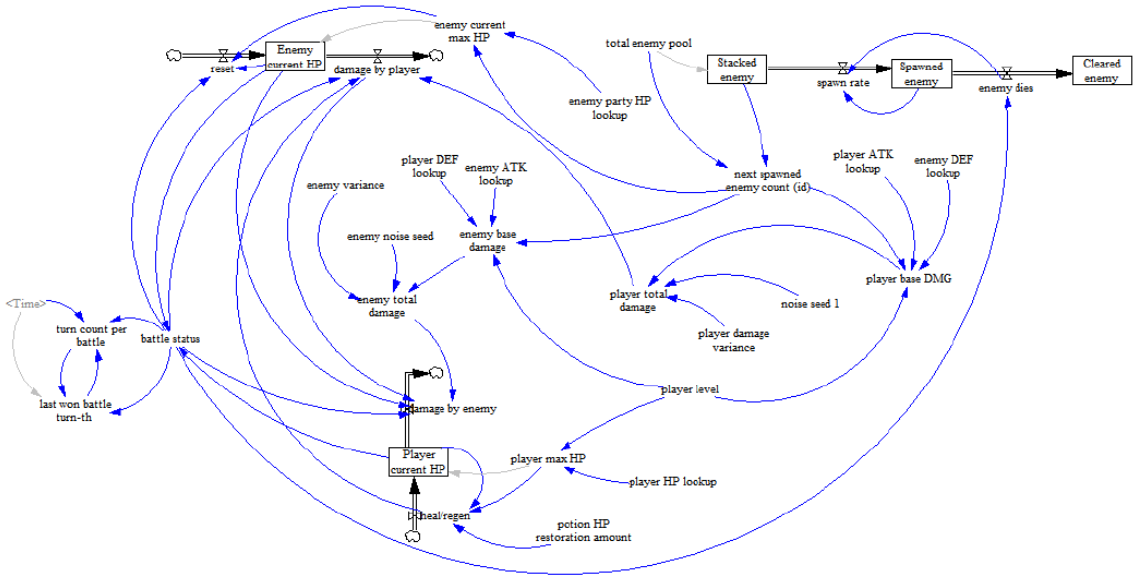


Fig. 2. Battle system dynamics model.

This simulation model uses "turn" as referenced time unit in order to generate variables dependent to a situational turn.Considering that attacks are done within time window of turn, it is more intuitive to immediately use turn as time unit rather than using second or minute of time unit. In addition, turn count also indicates level of challenge/difficulty [10]. By using Vensim PLE, the simulation result for every level and auxiliary variables are able to be observed by graph and table window.

2) Battle System Module

Battle module's purpose is to simulate battle process through a set of determined enemies into turns of action. Whether is the battle won by player or enemy, depends by enemy current HP and player current HP. Player and enemies attack each other, dealing a number of damage. The battle proceeds first by player's turn and then enemy's turn, which then repeats until one side loses. After that new battle starts with the next enemy in list as opponent. There are several assumptions made in the model design:

Assumption 1: The player's turn action is chosen between attacking or healing using potion by referring to current player HP. Low HP below certain percentage point will trigger potion usage in the turn. This model assumes potion to always restore a same fixed value of health points. On the other hand, enemy's turn action is always attacking the player.

Assumption 2: Delivered damage per normal attack action is calculated using random normal of ATK, DEF, and variance. This calculation refers to RPG Maker formula of attack.

Assumption 3: A queue of monsters that player will face sequentially in the game is needed to be inputted into the simulation. This model assumes that the designed RPG uses static encounter type.

Assumption 4: The simulation will continue creating new battle every time player wins a battle. However, when player loses, simulation will stop generating new battle simulation as player doesn't recover anymore.

Assumption 5: difficulty curve represented in this model mainly stems from the number of turn count in defeating the enemies. The resulted graph was meant to show designer if the statistics are balanced visually. Variable graph can be shown after each simulation by selecting the variable, then choose *graph* quick output buttons on the left-hand toolbar. For showing table to examine detailed value of a variable through time, choose *table*[11].

The main equation used in calculating total damage in battle system module is shown by function (2).

$$Total\ damage = INTEGER\ (RANDOM\ NORMAL(damage_{base} - damage_{base} * variance, damage_{base} + damage_{base} * variance, damage_{base}, variance, noise\ seed) + 1$$

(2)

The major input variables in this simulation model for game designer to use are both enemy and player statistics, which are inputted into lookup variables listed in Table I.

Table I.Lookup Variables

Variable	Type	Unit
Player Exp	Lookup	
Player HP	Lookup	HP
Player ATK	Lookup	
Player DEF	Lookup	
Enemy Exp Drop	Lookup	
Enemy HP	Lookup	HP
Enemy ATK	Lookup	
Enemy DEF	Lookup	

3) Experience Building Module

Experience building module's focus is to simulate character's experience growth based on when and how much experience points dropped by defeated enemy. Experience points will build up in a level variable called Player Exp. The variable affects the player level and indirectly affects other player statistics lookup variables.Balancing of player's power will be determined by the level-exp curve designed.

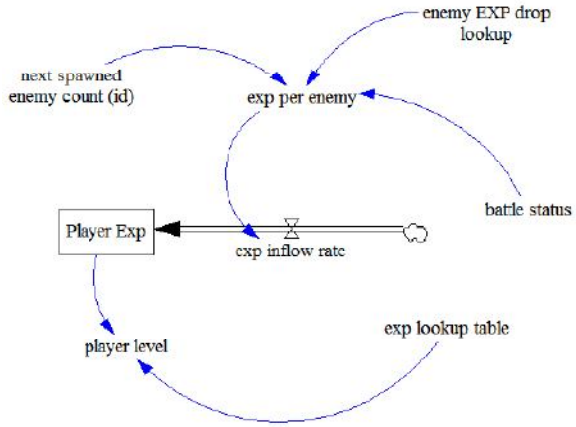


Fig. 3. Experience building system dynamics model.

B. Initial Model Simulation

Based on the purpose of model, several initial values of the simulation can be inputted as game designer's designed variables and statistics. Because the model is simplified has not yet able to represent a real complex RPG system, actors' statistics used are HP, attack, defend, and experience point. For player experience point is the number of experience needed to reach next level, while experience point of enemies are number of experience given to player after player's success to defeat the enemy. In addition to retrieved values from game designer, there rest of initial values are assumed from model design andconceptualization are shown on Table II.

Table II. Initial Values

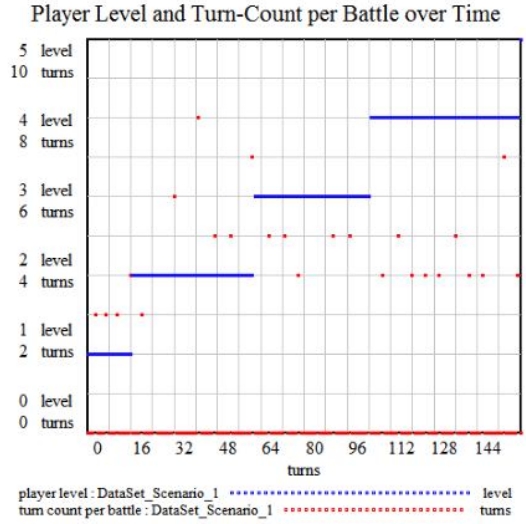
Variable	Type	Value	Unit
Spawned Enemy	Level	0	
Player Exp	Level	0	
Total enemy pool	Constant	30	
Potion HP restoration	Constant	25	HP

C. Simulation

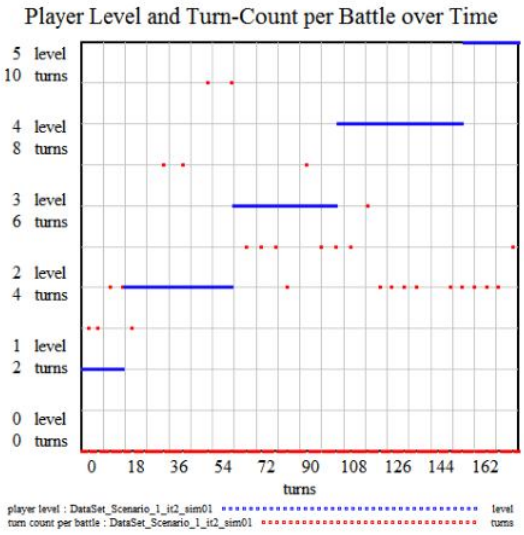
After collecting statistics data for both player and enemies, simulation will use the data to run battle simulation to give results of battle statistics. Collected data consist of character statistics and enemy set statistics. Enemy sequence of dungeon starts from 6th to 29th with dungeon boss placed on 26th.Results will include character level growth over time and also how many turns player performs in each battle. Turn counts information will give insight to designer about the current balancing to player. By skipping manual playtesting, this results information will play role as game designer's reference to make changes to actor statistics.

According to case study scenario, entering a new dungeon and fight a new monster can be a sudden jump of difficulty [10]. Similar jump can be observed from Fig.4a-c after fifth battle. Not only that, fighting dungeon later after 20 fights still did not reduce turn counts. Several observations were made and statistics were improved using iteration development and then simulated again to compare results.

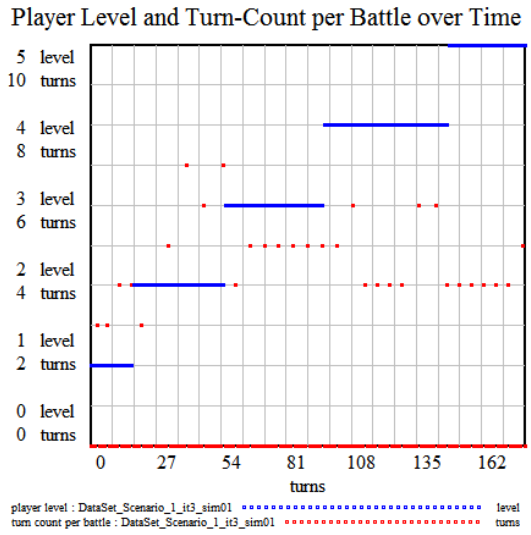
In first iteration, player's level at the end of simulation was level four as shown in Fig. 4a. After increasing the experience dropped from enemies and switching the enemy position, player ended up a level higher at the end of simulation.Fig. 4b showsturn count overshoots at dungeon startwith value two times previous enemies. This apparent imbalance jump could be improved by lowering the enemy power. We can also observe that dungeon boss was faced by player in level 4 instead level 3 of iteration 1.



(a)



(b)



(c)

Fig. 4. a. Player level and turn-count prebattle over time using iteration 1 enemy statistics. b. Using iteration 2 enemy statistics. c. Using iteration 3 enemy statistics

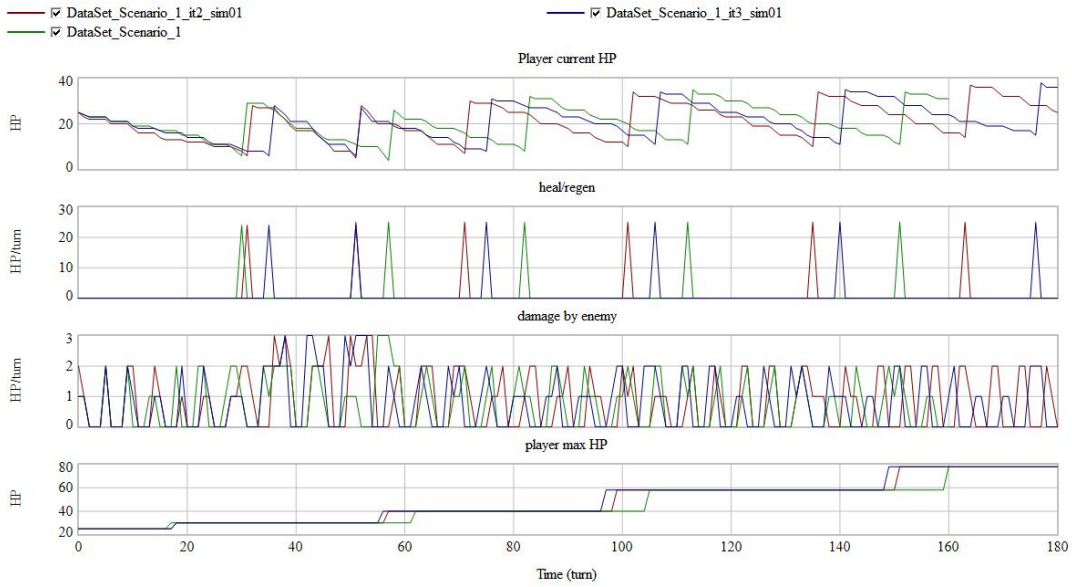


Fig. 5. Player current HP level variable over time graph.

So to improve the overshoots observed previously, another enemy rearrangement was done to introduce the weaker dungeon monster to the earliest part of dungeon. By looking at Fig. 4c we can see the reduction of overshooting. Last iteration balancing brought player to level 5 after 23 fights. If we look at Fig. 5, we can also see the player HP graph and find out how many potions used during this scenario. In addition, almost other variables like enemy HP can also be graphed out and compared to player HP to understand how the battles' balance.

V. RESULT AND DISCUSSION

Based on results of the simulations above, we are able to observe the turn-based RPG system dynamics model simulation's characteristics:

(1) Simulation's behaviors are discrete which is caused by the nature of all discrete variables used. Unlike the usual implementations of system dynamics that use continuous variables and continuous time unit.

(2) Putting strong enemies at the beginning can result in quicker time to gain level especially in early start of game. This will make player become easily used to fight the similar enemies and feel stronger for finishing them more quickly.

CONCLUSION AND LIMITATIONS

By shown results, apparently it is possible to simulate RPG playtesting using system dynamics model, even though the current model lacks complexities of a full

featured RPG. Playtesting simulation tool to help game balancing is barely available to designers, since it is difficult for computer to define fun for human. The current finding opens possibility of creating a tool mainly to give insights to game designer in order to decide what measures to take to improve the numbers. Despite being simplified, this tool could be useful in early development when complexity is still low. In the future, several versions of more complex model could be achieved and be used to analyze other components that are helpful to understand how balanced a game is.

ACKNOWLEDGEMENT

We thank Ian Schreiber for supporting us with his book material and insights about game balancing. We also thank Nelson Gustav Wisana, senior game designer of Agate International, for his insights, thoughts and viewsas a practitioner. We are also very thankful to our colleagues whom we had discussed with and had insights from.

REFERENCES

[1] Fullerton, T. (2008). Game Design Workshop: A Playcentric Approach to Creating Innovative Games (2nd ed.). USA: Elsevier Inc.

[2] Eriksson, J. (2014). The Making of a Conceptual Design for a Balancing Tool. Retrieved December 2017, from http://sh.diva-portal.org

[3] Homer, J. B., & Hirsch, G. B. (2006). System Dynamics Modeling for Public Health: Background and Opportunities. *American Journal of Public Health,96*(3), 452-458. doi:10.2105/ajph.2005.062059

[4] Brathwaite, Brenda and Scheiber, Ian (2018). You did 250 Damage! (or,Numerical Relationships). Manuscript in preparation.

[5] Brathwaite, Brenda and Scheiber, Ian (2018). Probability and Randomness. Manuscript in preparation.

[6] Perez, D. (2016). Beginning RPG Maker MV (pp. 114-121). New York, US: Springer.

[7] Perez, D. (2016). Beginning RPG Maker MV. New York, US: Springer.

[8] M. P., & Čerić, V. (n.d.). DEVELOPING SYSTEM DYNAMICS MODELS WITH "STEP-BY-STEP" APPROACH. *Journal of Information and Organizational Sciences; Vol 31, No 1 (2007)*.

[9] Pruyt, E. (2012). *Small system dynamics models for big issues: Hop, step and jump towards real-wold dynamic complexity: SPM2313*(pp. 33-40).

[10] Brathwaite, Brenda and Scheiber, Ian (2018). Progression in PvE Games. Manuscript in preparation.

[11] Hillman, D. (n.d.). *Systems Dynamics Using Vensim Personal Learning Edition (PLE)* [PDF].

★★★