UQAC
UNIVERSITÉ DU QUÉBEC
À CHICOUTIMI

MÉMOIRE

PRÉSENTÉ À

L'UNIVERSITÉ DU QUÉBEC À CHICOUTIMI

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INFORMATIQUE

PAR

TREMBLAY JONATHAN

B.Sc.A.

A NEW APPROACH TO DYNAMIC DIFFICULTY ADJUSTMENT IN VIDEO GAMES

FÉVRIER 2011

SUMMARY

Most video games suffer from system inflexibility, which is responsible for the player to give up on the game. As the players are expecting fun produced by different experiences and gaming sessions, the game should be able to adapt itself to the player and meet their expectations. It is important for the player to experience fun as it is what the game industry relies on to keep the player consume their products. Fun is not trivial to define and create, in order to understand fun in game, researchers have been using flow theory that provides a strong understanding of an emotion state that is linked to fun.

It is undeniable that every game should provide the possibility for the player to experience flow, which means it has to provide an understanding of the player' skills so it can adapt the game proposed challenge to their specific abilities. The goal of this research is to address this issue by proposing a new adaptive model for dynamically adjusting, in real-time, the difficulty level of a game in order to enhance the player's experience. This model has been implemented for validation in the form of a simple calculation/combat serious game called *Number to Number*. An experiment has been conducted with this prototype where 150 playing sessions have been completed by 32 players. Each player had to answer a detailed questionnaire on their playing experience. The results of this experiment were very promising, showing the value of the proposed approach and giving us clues for improving the model.

RÉSUMÉ

L'arriver de nouvelles consoles de jeux vidéo tel que la *Wii* de *Nintendo* ont ouvert l'industrie du jeux vidéo aux « joueurs casuals ». Dans cette nouvelle réalité, joueurs expérimentés et joueurs inexpirémentés évoluent dans le même environnement. Ils cherchent a s'amuser par le biais de différentes expériences et de sessions de jeu, c'est ici que le jeu devrait se plier aux exigences du joueur. C'est important pour le joueur de s'amuser en jouant, l'industrie du jeu vidéo se repose sur cette facette afin de porter les joueurs à consommer leurs produits. Cependant, l'amusement est difficile à définir et encore plus à créer dans les jeux. Afin de comprendre l'amusement dans les jeux vidéo, les chercheurs utilisent la définition de la théorie du "flow" qui se repose fortement sur la compréhension forte d'un état émotionnel qui est lié à l'amusement.

C'est incontestable que les jeux doivent assurer que le joueur puisse expérimenter une forme de "flow", dans un tel cas, le jeu doit comprendre le niveau d'habilité du joueur afin de pouvoir offrir un défi qui est à la hauteur des habilités qui sont spécifique au joueur. Le but de cette recherche est de répondre à cette problématique en proposant un modèle adaptatif d'ajustement dynamique (DDA), en temps réel, du niveau de difficulté afin d'améliorer l'expérience de jeu pour le joueur. Ce modèle a été implémenté afin de le valider sous la forme d'un petit jeu sérieux (combat/mathématique). Grâce à ce prototype, 32 personnes ont testé et répondu à un questionnaire portant sur leur expérience de jeu. Les résultats de cette expérience sont très prometteurs, démontrant la valeur du modèle proposé et pointant vers des indices pour des améliorations futures.

ACKNOWLEDGEMENTS

This master's project is part of the research activities of the Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activités (LIARA) at the Université du Québec à Chicoutimi (UQAC). I would like to use this opportunity to thank those who helped me complete this master. First I would like to express my sincere appreciation to one of my director, Professor, Dr. Bruno Bouchard, for his supervision, guidance, support and patience, as well as his generous help during my MSc studies. I would like to express my gratitude to my director, Professor, Dr. Abdenour Bouzouane, for his guidance and help during my project. I would like to express my appreciation to all of the students on campus who played my game and answered the questionnaire, without you this project would have been impossible. I would like to show my deepest appreciations to all my friends and collaborators who helped and encouraged me during my work.

Financial (in the form of scholarship) and in kind support received from the Ministry of Education of Québec (MEQ), the Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activités (LIARA), the Université du Québec à Chicoutimi (UQAC), the Département d'Informatique et Mathématique (DIM) and The Fond de Campagne de développement de l'UQAC.

Finally, I would like to express my warmest thanks to my life partner Lauren McGann, whose indescribable patience, constant correction and encouraging words were essential to the completion of this project. I would express also my sincere thanks for the encouragement of my parents and my family.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The video game industry is currently the commercial sector with the fastest growing rate among all entertainment medias worldwide (Ho and Huang 2009). Video games are becoming a bigger part of the youth, an average student spend in approximately 2 hours per day playing video games (Gilleade and Dix 2004). In the United States between 2005 and 2008, the value of the video game industry increased by 17% and it is now evaluated at $4,9 billion (Siwer 2010). This new reality is mainly caused by the arrival of new ways to interact with games. The Nintendo Wii now allows the player to use the controller movements to control their avatar in game (Natapov, Castellucci et al. 2009), the player can use a pen on a tactile screen on the Nintendo DS (Claypool 2005), they can now control the game using their whole body with Kinect from Microsoft for XBOX 360 (Bleiweiss, Eshar et al. 2010) or they can use their fingers on the Apple iPhone screen to interact differently with video games (Grissom 2008). The action of playing video games is simple, although defining this experience is not trivial. The following is an attempt to define and describe the video game experience in order to complete understand the problematic and thematic of this master.

When the player interacts with the game, this interaction exists in a structure that makes sense in an applied context (Salen and Zimmerman 2003). Although video games are not just applied interactions, according to Salen and Zimmerman (Salen and Zimmerman 2003), a video game is: "a system in which the players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome". The conflict in which the player is engaged is related to a problem that needs to be resolved, an certain action has to

be made in order to move to the next problem. The problem represents a challenge that needs to be addressed. In order to resolve the problem, the game needs rules to limit the player's actions and the player needs to know if the problem is resolved or not following their actions. The rules guide and limits the player's behaviour, meanwhile the game gives feedback to all their actions. Nevertheless, this definition is incomplete because it does not include any form of pleasure/fun. According to Crawford (Crawford 2003), a video game has to be fun and produces a form of enjoyment in the same structure defined by Salen and Zimmerman. Clearly, a video game has to be fun.

Even though the fun in game is directly related to the their parts. It is an emergent action that makes the game fun; adding the color green to the game does not necessarily make it fun. This reality of indirect linking between fun and game structure has lead researchers to develop a better comprehension of video games. This research addresses several topics, such as defining the semantics associated with video games (Schell 2008). The definition of tokens in game and their roles (Walz 2010). The need of letting the player fails sometimes (Juul 2009). As it is been seen, the game is limited by rules, but what does the rules are? More precisely the rules are consider game mechanics. Game mechanics are defined as a "series of interesting choices" by Sid Meier (Rolling and Adams 2003), the creator of the popular series Civilization (Meier). This definition is incomplete, it does not include any structure, conflict that evolves in time and any rules. This definition also lacks of regularity on how to define game mechanics, it refers to a certain instinct in order to find them in game (Salen and Zimmerman 2003; Hunicke and Chapman 2004). Although, Sid

Meier definition gives us a good start, as Rolling and Adams (Rolling and Adams 2003) pointed out, game mechanics is "one or more causally linked series of challenges in a simulated environment". This definition takes into account a certain structure that lies in the definition of games. In other words, game mechanics are actions the player do in order to solve a problems. Therefore, every action that the player does has to be related to the game challenge (conflict inside the game).

The challenge inside the game is define by the conflict which is determined by rules, in order to resolve the structure of the game we need to understand its rules. According to Salen and Zimmerman (Salen and Zimmerman 2003), rules limit the player's actions, are explicit and unambiguous, are shared by all players, are fixed, are binding and are repeatable. It is important that the game provides the player with a clear understanding of its rules. Using this knowledge the player can then construct their strategy and enter in what is called the magic circle.

The game evolves inside an abstract magic circles, the game only exist inside it and it exists only if the player respects the rules (Salen and Zimmerman 2003). It is because of the magic circle that during a 100 meters race, all the racers respect their line of race and do not interfere with the other ones. Its inside the magic circle that the rules make sense to players, without the magic circle the game just does not exist. As the player accepts the limit of the game, they can experience the emergent pleasure of the game.

1.1 DEFINING THE NOTION OF FUN IN GAMES

Emergence is the key to fully understand pleasure in games. Campbell (Campbell 1982) defined emergence as "a modest number of rules applied again and again to a limited collection of objects [that] leads to variety, novelty, and surprise. One can describe all the rules but not the set of all whole numbers." One of the classical application of this definition is the game of life from John Conway (Gardner 1970). This application constitute of cells that are either occupied (black) or free (white) and there are rules that govern the comportment of the cases, as we can see in Figure 0.1.

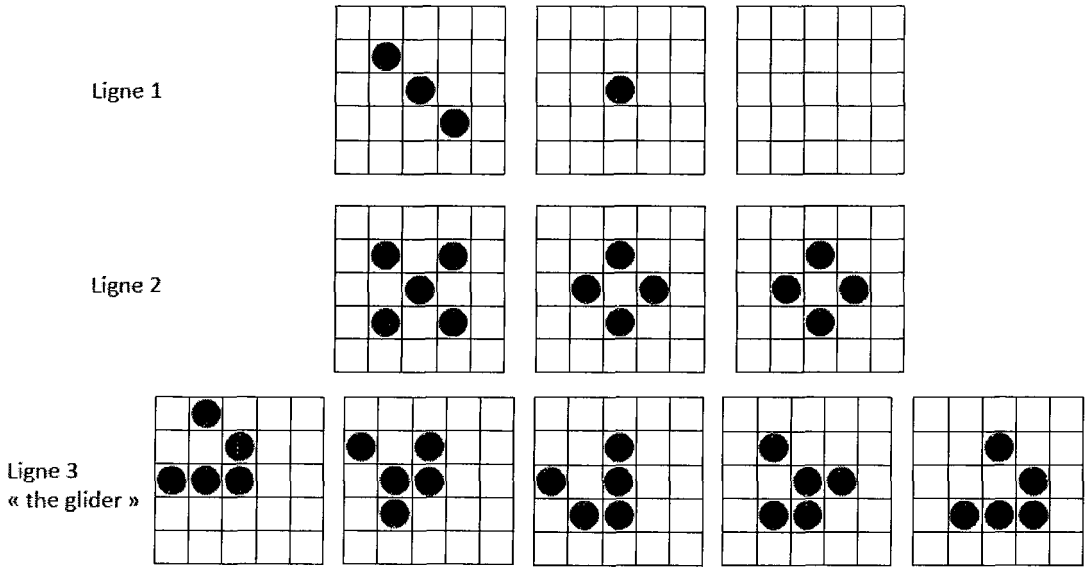

Figure 0.1 : John Conway's Game of life

In this figure, every column represents iteration of game of life and every line is read left to right. The game works as follow, when a cell has only one neighbour (an adjacent cell that is occupied) or has no neighbour, this particular cell will die at the next iteration from loneliness, the cell is then unoccupied. In the line 1 at the second iteration, there is only one cell occupied, as it is been said, because the cell is alone, it dies from loneliness in the following iteration. When a cell has more than three neighbours, it dies because of over population. In line 2, the cell in the center has four neighbours, in the second iteration, the cell in the middle dies because of the over population. The last rule says, every cell that has two or three neighbours survive or are born. In the first line, the cell in the middle has two neighbours, in the second iteration it survives. Using those three simple rules, models like the glider can emerge, in line 3. The glider is a simple form that moves in a diagonal line in the world of game of life. The emergence in video games is the result of the underling system (Salen and Zimmerman 2003). In other words, the rules that defines the game and the magic circle allow the fun to emerge.

Fun in game is intrinsic and the simple fact that the magic circle constraint the player's activities leads to fun (Salen and Zimmerman 2003). According to Koster (Koster 2004), fun in games is a result from the learning platform that it provides to the players. Every action posed by the player inside the game leads to a better understanding of the games mechanics of the game; they are mastering the game by playing it. The action of learning procures great amusement amount the player. LeBlanc (LeBlanc 2000) defined eight kind of form of happiness for a player to experiment:

1. Sensation: game as sense-pleasure
2. Fantasy : game as make-believe
3. Narrative: game as unfolding story
4. Challenge: game as obstacle course
5. Fellowship: game as social framework
6. Discovery: game as uncharted territory
7. Expression: game as soap box
8. Submission: game as mindless pastime

Within this structure it is possible to understand the different kind of pleasure a player can experiment. For example, in an adventure game like Baldur's Gates (Norton and Avery 1999), the pleasure of the game lies into the narrative, sensation, discovery and challenge the game propose. By understanding what kind of pleasure the game offers to the player, it leads to a better knowledge of the teaching faculty of the game and how the player interacts with it.

In the modern literature, fun in game relies in a more abstract (based on player's implicit experience) definition, the theory of flow defined by Csikszentmihalyi (Csikszentmihalyi 1990; Salen and Zimmerman 2003; Sweetser and Wyeth 2005; Chen 2006; Dormann and Biddle 2008; Nacke and Lindley 2008). The player experience flow when they feel a sense of control over the game, a disconnection with everyday life, the sense of self disappearing and time being altered (Sweetser and Wyeth 2005). Flow is not

trivial to introduce and it is hard to keep the player in a flow state during the entirety of their game experience (Chen 2006). In order to reach that flow state, the game has to provide a task that can be completed, give the possibility for the player to focus on this task, define clear goals, and provide constant feedback to the result of the player's actions (Sweetser and Wyeth 2005). Experimenting flow for the player is so gratifying that the player is ready to accomplish all sort of activity even though they could be in danger (Csikszentmihalyi 1990). Flow also requires that the challenge should not be too hard or too easy for the player.

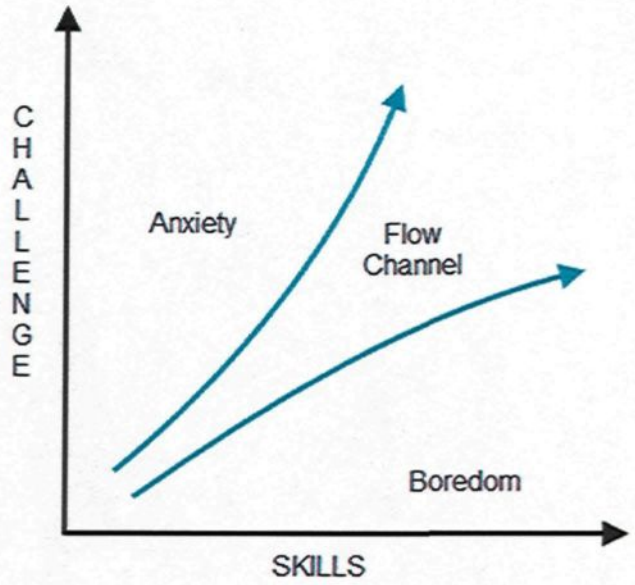

Figure 0.2 : flow model (Chen 2006)

In the Figure 0.2, the challenge provides by games is in relationship with the player's skills level. In order to let the player experience a constant flow state, the game has

to be balanced between skills and challenges. If the challenge is too hard for the player's skills, the player gets anxious (Koster 2004), on the order hand if the player is too skilled for the in game challenge, the player will get bored because the challenge is too easy. Overall, if the player's skills and the game challenge are too low, the player will be absent from the game and will not pursue playing the game (Chen 2006).

1.2 THE NEED OF DIFFICULTY ADJUSTMENT

As the player evolves playing the game and develops better skills at the game, the game should provide the player a challenge that suits their evolving skills (Csikszentmihalyi 1990). While playing, the game and the players evolve differently, as flow should always be provided (Chen 2006), the game should understand the players' skill level (Tremblay, Bouchard et al. 2010). Most current games do not provide any intelligent system to understand the players' experience as a result of not being able to provide flow to the player during the whole game experience. The classic approach to address this issue is to introduce multiple difficulty levels, for instance casual, normal and hardcore (Gilleade and Dix 2004). Using this approach, player is most likely to choose the right difficulty level for their experience at the beginning of the game. However, this solution is very limited and only partially addresses the problem. First, the player can underestimate or overestimate their own capabilities. Secondly, the player is constantly evolving in the game. So, they could select "casual" at the beginning, become more experienced while playing and get bored from there because the challenge is now too easy for them and should have been

readjusted to "normal". In the end, using an unique difficulty level for the entire game session can be very problematic because player can be exceptional in achieving one type of challenge and at same time be completely inexperienced in another type of challenge. This problem is even more significant when designing serious games (Tremblay, Bouchard et al. 2010), which are often targeting casual gamers with heterogeneous abilities, such as the elderly. In serious gaming, it is imperative for the player to experience fun and stay hooked to the game. In the case the player does not want to play because the fun is not there, their experience and learning faculty from the game are altered until its inexistence. In any expression of games, the player seeks recognition of patterns to study (Koster 2004). It is important to give the player the right amount of difficulty so they can seek and understand specific designed patterns such as medical procedures, military tactics, mining for oil, etc.

Nevertheless, a great game should be able to provide flow to all its players, even if they have very different skill sets. This can only be achieved by introducing an alternative approach in the game that is able to dynamically adjust the difficulty level to the players' profile learned through their performance (Tremblay, Bouchard et al. 2010). Moreover, a game should be able to recognize when the players are in need, offer them help and adapt challenge to their skills (Lindley and Sennersten 2008). In summary, the players are expecting games to be fun throughout, not too hard or too easy, and to adapt the game world to their experience and recognize when they are in need. The game should adapt itself to the player, not the other way around.

1.3 RELATED WORKS ON ADAPTIVE GAME MECHANICS

In the last few years, in order to address this issue, researchers have proposed different computational, algorithmic and design approaches helping to enhance the player's experience. Video games of all sort have been developed using those kind of models: Max Payne from Remedy Entertainment (2001), Final Fantasy VIII from Square Enix (1999), God Hand from Clover Studio (2006), etc. Although those games are fill with information about the impact of dynamic adjustment on the player's experience, they do not offer an understanding and development of the structures and algorithms used to adapt the game to the player. Furthermore, the following researchers give enough information about their models for us to understand and reproduce them. Chen (Chen 2006) proposes an purely design approach to make sure the player is always in a flow state. This design approach uses simple game mechanics that are imbedded in the game experience. By letting the player controls the difficulty levels from their action, it leads them to a better experience. Hunicke (Hunicke and Chapman 2004) proposes a computational approach, it uses algorithms and states to understand the player's experience. The system process the data the player can produce inside the game, with these data, it is possible to determine the player's expectation about the difficulty. Also the system determine the different player's states they could experience in the game, the system is, then, able to change the difficulty of the game in order to enhance their experience. Lindley and Sennersten (Lindley and Sennersten 2008) present a model that recognizes the game mechanics algorithm. In other word every action in the game can be represents by an accomplishment algorithm. All game mechanics

inside a games have a structure than can be understand by a computer language. In that case, if the player does not follow how the game mechanics completion is defined, it means they do not fully understand the game and the game should adapt. The adaptation is straight forwards as it helps the player to complete any tasks. Gilleade and Dix (Gilleade and Dix 2004) propose a model that reacts to the player's level of frustration. When the player is experiencing frustration, it means that they are not able to complete a game mechanics or they did not fully understand the puzzle. The level of frustration can be determine by the pressure that the player exerts on the controller buttons. When the player is getting to a certain point of frustration, it is possible to adapt the difficulty of the game, it leads the player to a better experience. In general, the idea is to let the game adapt to the player not the other way around by giving the game different adaptive tools.

1.4 CONTRIBUTION OF THIS THESIS : A NEW MODEL OF DYNAMIC DIFFICULTY ADJUSTMENT

The aim of this research is precisely to address this important issue that we raised in Section 1.2. We address this issue by proposing a new adaptive computational model for dynamically adjusting, in real-time, the difficulty level of a game according to the players' performance and rank them in order to enhance their gaming experience.

The first goal of this research is to design a flexible adaptive model that can, in real-time, adapt to the player's expectation. The foundation of this model relies on a computational approach for Dynamic Difficulty Adjustment (DDA) (Hunicke and

Chapman 2004) and it can be seen as an extension of our previous model proposed in (Tremblay, Bouchard et al. 2010). The model is composed of two distinct layers. The first is called the *tactical layer*, which is unstable in the long-term but is useful to evaluate the players' skills for quick actions. The second part is called the *strategic layer*, which is used to adjust the difficulty in the long run by exploiting the theory behind the well-known ranking system ELO (Coulom 2010). The calculated rank is used to understand the player's performance over a long period of time and to adjust the game according to their skills. The model , then, feeds important information to the parts game (AI, generated content, etc.) based on the ranking value. When using the *tactical* and the *strategic* layers together in a video game, it is possible to build a flexible game experience that better suits different types of players.

The second goal of this research is to enhance the player's gaming experience by introducing DDA system. Our new model uses rank system that understands the player's level of skill by challenging them using rank on every challenges. This allow us to change the difficulty in real-time and predict the player's challenges results. To achieve this objective, the model is implemented for validation in the form of simple calculation/combat game called *Number to Number*. An experiment has been conducted using this prototype where 150 playing sessions have been completed by 32 players. Each player has to answer a detailed questionnaire on their playing experience. This experiment leads to conclusion about the success of this second goal. So far, the results are very promising, showing the value of our new model.

This work can be seen as a small part of a vast project conducted by the LIARA laboratory at UQAC (Bouchard, Bouzouane et al. 2007). It consists of the development of a game platform that is specifically adapted to people with brain diseases, e.g. Alzheimer's. This new game platform will be used as an accessible and cheap tool enabling the patients to train their brains several times a week in order to help slowing the degenerative process of their mental functions (Srinivasan, Butler-Purry et al. 2008). By designing games that suit the elderly with cognitive impairments, helping them at the right moment and adapting the difficulty to their skills, they should be able to experience fun and flow playing the game. This will encourage the patients to play more and to use their cognitive functions by having a deeper investment in the game (Srinivasan, Butler-Purry et al. 2008).

1.5 RESEARCH METHODOLOGY

This research project followed a three steps methodology. The first step of the project was to acquire knowledge targeting video games and furthermore adaptive game mechanics. There are lots a well written related work on the subject of video games: (Crawford 2003; Koster 2004; Lazzaro and Keeker 2004; Sweetser and Wyeth 2005; Schell 2008; Juul 2009; Siwer 2010). These works were really important to understand the underlying structure of video game, the impact of games on player, how the game matters for them and the comprehension of the game industry. A formal study of the classic approach on DDA was made: (Salen and Zimmerman 2003; Gilleade and Dix 2004; Hunicke and Chapman 2004; Hunicke 2005; Chen 2006; Lindley and Sennersten 2008;

Tremblay, Bouchard et al. 2010) in order to understand every model and how they can be implemented to enhance the player experience.

The second step had for goal to define the basic theory needed for a new model to address the problematic presented in Section 1.2 using the related work in Section 1.3. In order to achieve the first objective, the design of the propose model is a complementary design approach to (Chen 2006) and a reciprocal approach to (Hunicke 2005). The model used by Hunicke is well defined and introduces important concept for our implementation. We used a model that is based on a clear DDA system using ranking system to determine the player's difficulty level. By doing that it is possible to understand the player's skill level in a clear and computational way.

The final step of the project had for goal the validation of the new DDA model implemented in a prototype called *NtN* and conclude on the second objective. As our second objective is enhance the player's experience based using DDA, the model was implemented in a cross platform language (AC3) using Flash Player which is widely used for web based video games. The Flash platform allows any developer to build and publish a game for a worldwide audience. It allowed different tester from around the world to play and test the game online in their browser without downloading any component. In order to obtain result on the game and the model for validation, player was asked to answer a questionnaire about their game experience and the game; 32 players answered the questionnaire and the results are really promising to validate our project.

It is important to notice that parts of this new DDA model have been published in the proceedings of the International Conference on Computer Supported Education (CEDU): session "Gaming platforms for education and reeducation" in Valencia, Spain in 2010. This paper was well received by the community and this recognition constitutes a clear sign of the pertinence of the solution proposed in the thesis. An extended version of this previous paper is under review for publication in the Journal of Computer Entertainment, Elsevier.

1.6 THESIS ORGANIZATION

The rest of this document is organized into four chapters and follow the chronological of the research events.

Chapter 1 is an introduction to the problematic of the thesis and serves as a context for the development of the thesis. It presents an introduction to different concepts underlying in video game. It describes the problematic relying in the relationship of the player and game. In the end, it presents the basic comprehension model of our new model implemented for this research.

Chapter 2 discusses the previous work related to our research topic. It introduces the basic information about flow in games and how it is use to define fun in them. Different adaptive game model are then study to fully understand the problematic and their implementation. The following models are introduces: DDA system using algorithm

approach (Hunicke and Chapman 2004), Adaptive game flow using a purely design approach (Chen 2006), Gameplay schemas using algorithm to define the game mechanics from a semantic point of view (Lindley and Sennersten 2008) and frustration adaptation based on the player's level of frustration (Srinivasan, Butler-Purry et al. 2008).

Chapter 3 is the contribution from this research. It presents the formal foundations of our new DDA model, its conception and its implementation in the form of a simple calculation/fighting game called *Number to Number*.

Chapter 4 presents the results of a preliminary experiment conducted with the prototype (*NtN*) where 150 playing sessions have been completed by 32 players. It also discusses the limitations of our DDA system and the expectation about DDA in games from a player point of view.

Finally, Chapter 5 concludes this research by summarizing our contribution compared to the related work. It presents future perspectives of this work. The chapter ends on a more personal perspective about further work.

CHAPTER 2

RELATED WORKS ON DYNAMIC DIFFICULTY ADJUSTEMENT

Following the explosion of the industry with casual gamers, it is getting harder for game designers to foresee the adequate difficulty level for different type of players (Tremblay, Bouchard et al. 2010). To address this issue several researchers (Gilleade and Dix 2004; Hunicke and Chapman 2004; Chen 2006; Lindley and Sennersten 2008) have explored different avenues to implement adaptive game mechanics based on flow theory. First, flow is introduce for a better understanding of how flow is generate by video games. From the scholars work, we can circumscribe four main approaches that lead to difficulty adjustment in real-time. The first, (Hunicke 2005) is simply called Dynamic Difficulty Adjustment (DDA) and consists in a purely computational model. The second, (Chen 2006), named Adaptive Flow Model, is based on purely design approach that add adativity to game mechanics. The third, (Lindley and Sennersten 2008), is called Gameplay schema model and uses an algorithmic approach to understand the player's actions. The last, (Gilleade and Dix 2004), is based on the players' level of emotion (arousal) which is described as an adaptive model based on frustration feedback. In this section, we will present them and review the advantages and limitations of all these previous approaches.

1.7 NOTION OF FLOW

According to Chen (Chen 2006), the methodology used to help game designer to understand flow in video games is not enough developed. Scholars (Salen and Zimmerman 2003; Hunicke and Chapman 2004; Koster 2004; Hunicke 2005; Sweetser and Wyeth 2005;

Chen 2006; Dormann and Biddle 2008; Nacke and Lindley 2008; Tremblay, Bouchard et al. 2010) have recently been using flow to understand fun and amusement in games.

Flow is an emotional and psychological state of focused and engaged happiness, when a person feels a sense of achievement and accomplishment, and a greater sense of self (Salen and Zimmerman 2003). Csikzenmihalyi used professional gamers to describe the mental state of flow. He wanted to understand how the flow state is created, he came up with 8 prerequisites for it :

> First, the experience usually occurs when we confront tasks we have a chance of completing. Second, we must be able to concentrate on what we are doing. Third and fourth, the concentration is usually possible because the task undertaken has clear goals and provides immediate feedback. Fifth, one acts with a deep but effortless involvement that removes from awareness the worries and frustrations of everyday life. Sixth, enjoyable experiences allow people to exercise a sense of control over their actions. Seventh, concern for the self disappears, yet paradoxically the sense of self emerges stronger after the flow experience is over. Finally, the sense of the duration of time is altered; hours pass by like minutes, and minutes can stretch out to seem like hours. (Csikszentmihalyi 1990)

As seen, it is pretty striking how Csikszentmihalyi's definition could be use and apply to video games. Based how flow theory (Sweetser and Wyeth 2005) propose a model that allows designer to evaluate the fun factor in their games. They determine how flow can

be apply to a game and understand its repercussion over the player's experience. The Table 0.1 introduces an evaluation spreadsheet to determine if the game has all the prerequisites to let the player experiences flow state or in other words fun and enjoyment.

Table 0.1 : Gameflow Criteria for player enjoyment in games

Element	Criteria
Concentration Games should require concentration and the player should be able to concentrate on the game	• games should provide a lot of stimuli from different sources • games must provide stimuli that are worth attending to • games should quickly grab the players' attention and maintain their focus throughout the game • games should have a high workload, while still being appropriate for the players' perceptual, cognitive, and memory limits • players should not be distracted from tasks that they want or need to concentrate on
Challenge Games should be sufficiently challenging and match the player's skill level	• Challenges in games must match the players' skill levels • games should provide different levels of challenge for different players • the level of challenge should increase as the player progresses through the game and increases their skill level • games should provide new challenges at and appropriate pace
Player skills	• players should be able to start playing the game

Games must support player skill development and mastery	without reading the manual • learning the game should not be boring, but be part of the fun • game should include online help so the players do not need to exit the game • players should be taught to play the game through tutorials or initial levels that feel like playing the game • games should increase the players' skills at an appropriate pace as they progress through the game • players should be rewarded appropriately for their effort and skill development • game interfaces and mechanics should be easy to learn and use
Control Players should feel a sense of control over their actions in the game	• players should feel a sense of control over their characters or units and their movements and interactions in the game world • players should feel a sense of control over the game interface and input devices • players should feel a sense of control over the game shell (starting, stopping, saving, etc.) • players should not be able to make errors that are detrimental to the game and should be supported in recovering from errors • players should feel a sense of control and impact onto the game world (like their actions matter and they shaping the game world) • players should feel a sense of control over the actions that they take the strategies that they use and that they are free to play the game the way that they want (not simply discovering actions and strategies planned by the game developers)

Clear goals Games should provide the player with clear goals at appropriate times	• overriding goals should be clear an presented early • intermediate goals should be clear presented at appropriate times.
Feedback Players must receive appropriate feedback at appropriate times	• player should receive feedback on progress toward their goals • players should receive immediate feedback on their actions • players should always know their status or score
Immersion Players should experience deep but effortless involvement in the game	• players should become less aware of their surrounding • players should become less self-aware and less worried about everyday life or self • players should experience an altered sense of time • players should feel emotionally involved in the game • players should feel viscerally involved in the game
Social interaction Games should support and create opportunities for social interaction	• games should support competition and cooperation between players (chat, leaderboard, challenge, etc) • games should support social communities inside and outside the game

In first hand, it is really important to let the player concentrate on the game. When all of a person's relevant skills are needed to cope with the challenges of a situation, that person's attention is completely absorbed by the activity, and no excess energy is left over to process anything other than the activity (Csikszentmihalyi 1990). In that order the game

should always keep the attention of the player on the game; at 10 seconds, 10 minutes or 10 hours of play time, this is suitable by increasing the difficulty of the game at the pace the player is mastering the game mechanics.

In second, the challenge is often considered the most important designed part of a game (Sweetser and Wyeth 2005). In order to let the player experience a flow state it is important to offer them a challenge that suit their ability as express in Figure 0.2 (Johnson and Wiles 2003). The challenge should not be too hard or too easy for the player's skills (Tremblay, Bouchard et al. 2010). Games create enjoyment by challenging the player, often taxing the limits of their memory and performance (Lazzaro and Keeker 2004).

In third, for a game to be fun and pleasant to play, it needs to support the development and the mastering of the in game player's skill. The game should let the player learn its game mechanics from simple tutorial. As the player is learning it is important that the game rewards them for their accomplishments (Koster 2004).

In fourth, the game should let the player have control over their actions. The game should let the player learn the controls and the graphic user interface (GUI) using tutorials that is pleased and not too complicated. If the player is overload by the amount of mechanics; they can learn and are most likely to leave the game behind (Sweetser and Wyeth 2005). It is important that the game respect the player's controls schema, if not the magic circle will be broken. The game has to give the perfect control of itself to the player including the world, narrative, characters, avatar, etc.

In fifth, the player has to accomplish clear and precise tasks. All along the play session the player has to be aware of their goals (Salen and Zimmerman 2003). Although all the levels should have different objectives for the player to accomplish to prevent repetitive game content.

In sixth, the game should always provide the player with clear feedback of their actions. During flow, the player can progress because they always know how close they are to succeed. Therefore, the game has to provide constant feedback about the state of the goal and/or the task (Sweetser and Wyeth 2005). Evermore, the game has to constantly give the player feedbacks about their little actions, for example when the avatar jumps; the game plays the jump sound.

In seventh, the player has to experience a deep effortless engagement with the game. A player that is fully immersed in the game is less self-aware, their problems are gone and the notion of time is altered (Salen and Zimmerman 2003; Sweetser and Wyeth 2005; Nacke and Lindley 2008; Schell 2008).

Overall, looking at each point of the Table 0.1 leads to a better understanding of the flow in game. Although Sweetser et Wyeth went a little further and used the criteria introduced in Table 0.1 to evaluate if a video game is fun or not. By comparing a design to this table, it is then possible to determine if the game will be fun or not. Although, it is not an assumption, it is based on the flow theory and not on the nature of fun itself, some game

might not respect every point and still be extremely fun and the comprehension and evaluation of flow is subjective rather than being objective.

1.7.1 Applying Gameflow

By applying gameflow criteria to a game like Warcraft 3 from Blizzard 2002, it is possible to understand and predict the game popularity. Warcraft 3 is a Real-Time Strategy (RTS) game where the player controls a force of unit (e.g. soldiers, tanks, wizards, etc.), which they must build, train and upgrade in order to defeat the enemy (Rolling and Adams 2003). The player has a god like view over the game map where they use mouse to select units and give orders. The player has access to a GUI that allows them to build buildings, control units, check their resources and view a mini-map of the game. The strategy is to master units, resources and building at the same time. RTS games often consist of campaigns, in which the player is lead through many maps or levels. Also the player can play skirmishes against other human player or computer-based players.

1.7.1.1 Concentration

Warcraft 3, see Figure 0.1, meets the concentration criteria by providing a multitude of high-quality stimuli. The world, units, buildings, and characters of the game are all intricately detailed, with unique animation, sound, speech, and appearance. As the stimulus have to be in multiform (sound, graphics, animation, speech); Warcraft 3 provides the player with stimulus that are unique and meaningful according to the race they are playing.

Each race (Human Alliance, Orcish Horde, Night Elf Sentinels, Undead Scourge) has a different theme, which affects the appearance of the interface, terrain, units and buildings. The player quickly understand how the world and races are separated. The player is also really intrigued at their first play session, the game starts with a spectacular visualisation of the world and context. The player wants to know more as they start playing. The game keeps the player's attention with the growing dramatic tension, interesting and varied goals, good-sized missions and rewarding cut-scenes that leaves the player wanting more.

In order to let the player concentrate on more interesting and important tasks, the game alleviated unimportant one such as micro managing the workers, casting spells, finding a path to move units, etc. The missions and goals feel important and are central to the storyline. As the background the player is given by the game in the introduction and through the narrative makes the missions and actions meaningful.

The workload of Warcraft 3 is high because of all the stuff the player has to monitor (resource, collection, population limit, unit activities, progress of fights, research, building and production, number of units, upkeep) as well as many tasks to perform (upgrading, building, defending, exploring, maintaining, researching). Even though, the game simplified it all so the player is not overburdened. For example, there is a population cap so that player can have only a limited number of units, which they must take care of and use widely. There are also a limited number of buildings, unit types and upgrades that reduces the cognitive load of the player. The simplicity and design of the GUI also makes it easier

for the player to express what they want to do, so the workload comes from playing the game and not from using and understand the software.

Figure 0.1 : Warcraft 3 screenshot

1.7.1.2 Challenge

Warcraft 3 easily meets the challenge criteria in the campaign mode, but is less successful in the skirmish mode. It is impossible to change the opponent AI settings in that mode and it is quite difficult. It is possible to beat it with for a professional but certainly impossible for a novice player. Although as the campaign progresses, the difficulty

increases through the missions and player learns how to masters all four races. When a switch of race occurs, new game mechanics are introduce which leads to a new playing style. The game is well-paced, with optional side quests that players can complete if they want more to do or more breadth to the game.

1.7.1.3 Player skills

Warcraft 3 is exceptional for developing and accommodating player skills. As describe in Table 0.1, player is allowed to play the game without reading any manual even thought RTS are complex games with a steep learning curve. The GUI is well labelled, the game follows RTS conventions, such as using right click to give basic commands. In need the player has access to plenty of help; tool tips, online help, tips, and tutorial. In the campaign, one race is introduced at a time and new units and buildings are incorporated gradually, so that the player can become familiar with each one in turn. In order to help the player developing their skills, the game rewards them with items, experience, and skills for their heroes. The game interface and mechanics are easy to use and learn, as there is plenty of help.

1.7.1.4 Control

Warcraft 3 helps the player to feel in control of every aspect of the game. The player feels a greater control over their units because of the low population caps. It is easy to move the units around as the path finding algorithm is really good and the native mouse control is based on point and click. Player can feel in control of the GUI as it is well

designed, including a bottom heavy menu system, hot keys, detailed tool tips, shallow menus and clear icons, see Figure 0.1. The game shell is also easy to use, as it is simple to start, stop, pause, save, or join a game. It is also well integrated in the game style and looks real in the game world. Feeling a sense of impact and control of the game world is criterion that Warcraft 3 does not really respect. The linear story in the campaign means that the player experiences no real freedom or consequences on the story. The payer does, however, have freedom and control of their actions and strategies that they use during play.

1.7.1.5 Clear Goals

Warcraft 3 clearly presents both overriding and intermediate goals. At the beginning of the game, the game introduces the goal and background story for the player to understand their in game actions. The goals of the game are presented and questions arises for the player to answer by playing the game. Continuous in game cut-scenes clearly present goals and further the story. Between campaigns for each race, a movie is presented that provides the background and goals for the following campaign.

1.7.1.6 Feedback

Warcraft 3 provides continuous feedback to the player on their goals, actions and status. The player is notified when they reached a mission objective. The player can also check the status of the mission goals, sub goals and completed goals. The mission objective icon flashes when there is something new for the player to check and certain area of the map would flash to guide the player to complete their next goal. At the end of each mission,

player receives their scores, broken down into heroes, units and resources. Additionally, every action that the player performs and the progress of every task provide feedback, usually in multiple forms (speech, sound, animation, text, etc.)

1.7.1.7 Immersion

It is easy to become engrossed in Warcraft 3, as there is so much to concentrate on, so many tasks to perform and things to monitor, as well as graphics, sound, animation and intricate details to be admired. The game causes player to feel tension, excitement, anger at enemies and feel a personal connection to their units and heroes. The immersion level is not comparable to a FPS because the player is not incarnating their heroes, the level of abstractness of RTS allows player to feel in contact with their actions and strategies.

1.7.1.8 Social Interaction

Warcraft 3 supports competition and cooperation between players, as well as social communities inside and outside the game. It is possible play online using Battle.net or using a lan party connection. The included game editor allows player to create and share new game content. Player can chat to other players in the game via text, and although this is fairly limited, it is the current standard in RTS games.

In general gameflow is used to understand the criteria necessary for a game to provide a state of flow to the player. Game designer could uses it as different points a game should have to be fun. Sweester et Wyeth (Sweetser and Wyeth 2005) prove that it could

also be use to determine how much fun the game could be and if it is going to be popular or not.

1.8 DYNAMIC DIFFICULTY ADJUSTMENT (DDA)

The player creates meaning by interacting with the game's internal systems. One of the first approach to adaptive game mechanics apply to video game is called DDA. The game Xévions (1982) for Atari is one the oldest game using DDA system. It is a top scrolling shooter game, the game uses adaptive logic that kept the game challenging (Seitz 1997). The prior is to ultimately let DDA takes control away from the designer and put it in the hand of the code. So DDA offers alternative-modulating in-game systems to respond to particular players' abilities over the course of a game session. DDA is based on the mathematical analysis of structures and relationships within a game system (Hunicke 2005) and on the players' flow experience. DDA uses the flow principle in order to keep the game intriguing and enjoyable, the designer wants to keep the player in the flow channel as we can see on Figure 0.2. As the game balancing and tuning can't be complete automated, directed mathematical analysis can reveal deeper structures and relationships within the game system. Using the right DDA structure, everything from narrative structure to procedural art assets can be adjusted to the difficulty level (Mateas 2002). It is important to completely understand the design and how the system could interact with the game in order to challenge the player.

1.8.1 Hamlet

In general, the DDA system has to monitor the game statistics according to designed metrics, defining adjustment actions and policies, executing those actions and policies, displaying data and system control settings and generating play session traces. An example of such a system is the Hamlet system (Hunicke and Chapman 2004) use for First Person Shooter (FPS), it uses statistical metrics to monitor incoming game data. It estimates the player's future state from this data. When an undesirable but avoidable state is predicted, the system intervenes and adjusts the game setting as necessary. Basically, it is trying to find when the player is experiencing a state of play where the current means can no longer accomplish necessary and immediate ends. Hamlet wants to help the player progress through the game.

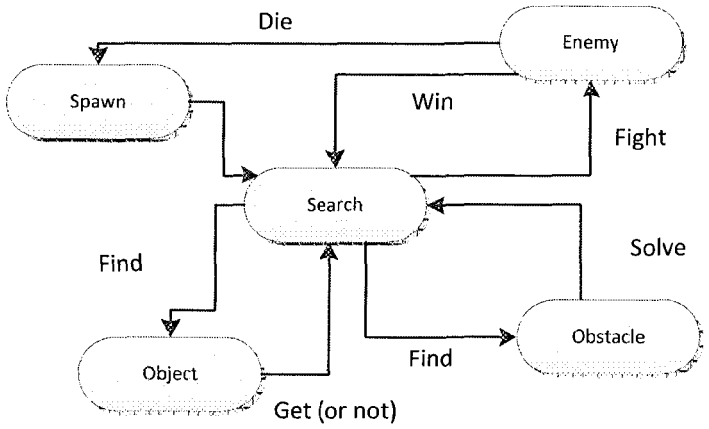

Figure 0.2 : FPS simplified state transition diagram

As Hamlet is designed to keep the player in the flow state it encourages certain states, and discouraging other. In Figure 0.2, the general states are represent for a FPS, the player seeks challenge when they meet enemies, obstacles and objects, although the state of search could stay on for a very long period of time. It is important that the player does not get bored from the state of search, that is where a DDA system should step in. The goal of the system is to keep the player in an engaging interaction loops, for the most appropriate period of time, given their level of overall skill and game-specific experience.

Hamlet uses the abstract game mechanics a player accomplish in a FPS. Player engage in loops of searching, retrieving, solving and fighting. With each new level, new enemies and obstacles are introduced. Overall, difficulty increases with time, as does skill acquisition. It is important that the system assess when adjustment is necessary, determine which changes should be made and execute changes as seamlessly as possible. In a FPS it is possible to determine is the player is looping in any states, see Figure 0.2, by noticing repeated inventory shortfalls; when the player's available resources fail to meet the immediate demands. Hamlet looks at the player's health, more precisely the damage a player takes over time.

$$F(d) = 1/\left[\sigma\sqrt{2\pi}\right]\int_{-\infty}^{d} e^{(x-u)^2/(2\sigma^2)}dx \qquad \textbf{0.1}$$

The Equation 0.1 represents the normal distribution of damage taken by the player. *F(d)* is the probability of taking any damage at a given time. This distribution leads to a

better understanding of the player's state. A same normal distribution function is defined in Hamlet to determine if a shortfall happened in the inventory. Using these distribution, it is possible to determine if the player is in need or not.

The adjustment needed take from on two different actions: reactive and proactive. The reactive actions adjusts elements that are "in play" (i.e. entities that have noticed the player and are attacking). This includes directly manipulating the accuracy or damage of attacks, strength of weapons, level of health, etc. The proactive actions will adjust "off play" elements (i.e. entities that are spawned but inactive or waiting to spawn). This includes changing the type of enemy, spawning order, health, accuracy, damage and packing properties of entities that have yet to appear on screen. Overall, reactive changes run the risk of disrupting the player's sense of disbelief. They can make it difficult to interpret the immediate behaviour or in-game obstacles, causing the game to appear schizophrenic (Sengers 1998). Proactive changes, because they happen away from player's eyes, are less likely to interrupt the player's suspension of belief. Hamlet is a good example of how DDA works and how it could be apply in a video game.

1.8.2 DDA principle

In a general way, the DDA system works as expressed in Figure 0.3. The player produces data by playing the game. The data is then analyzed by the system and sent to the game part of the system. The analyse consist in looking at the player's game data, based on

the game mechanics. After evaluating the player's skill level, the system can determine if the game should be adapted or not.

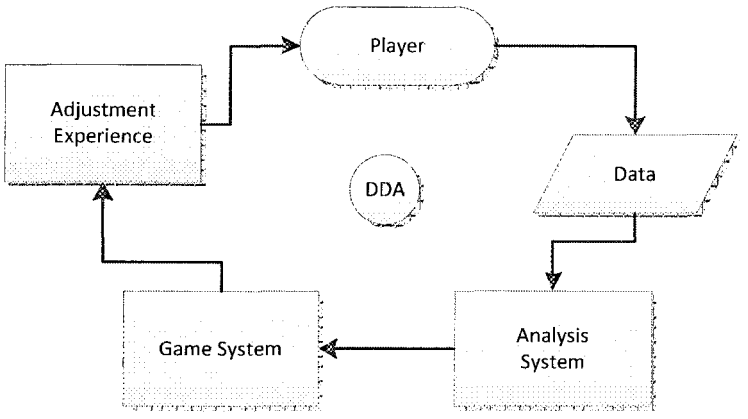

Figure 0.3 : DDA system

The game part will make some adjustments to the game mechanics according to the analysis results which leads to a real-time adjustment of the player's experience. DDA uses a system that changes the game mechanics without the players knowing it. These changes are made in order to keep the players challenged and interested (Hunicke 2005). First, the system computes the player's data; player's position, player's health, player's ammo, etc. Following the system assessment, the system chooses the data that reflects the player's state of flow. The system analyses the players' state of flow and notifies the game of any changes. Lastly, the game applies the modifications (Chen 2006).

For instance, imagine a player playing Half Life 1 see Figure 0.4, they just reached the second engagement of *Case Control*. There are four enemies, and the player has 45% of

health. A DDA system observes their inventory stats, to see if they are flailing; in this situation, it represents the player dying repeatedly before completing the engagement, often with a majority of enemies standing.

Figure 0.4 : Half Life 1

If the player is flailing, the immediate action would be to donate a health pack somewhere in the scene, upgrade the player's strength or ammo or reduce the accuracy or the strength of enemy attacks. Depending on the success of these individual actions, the system could intervene again. If, after time, the player requires significant adjustments to the initial levels set by the game designer, the system could drastically chance the difficulty of the level by changing the enemy stats, the number of spawned enemies, the type of enemy, available ammo and health, etc. The key here is that the system will intervene

iteratively, allowing for trial and error. The game will gradually change to accommodate the current player.

The system analyses the player's data based on the player's flow experience. However, one of the major problems with DDA is that the system bases its decisions on the player's flow state using only raw data. The raw data used represents the performance of the player, which is objective, while flow is subjective (Chen 2006). On the other hand, DDA is straight forward to implement and understand (Hunicke and Chapman 2004).

1.9 ADAPTIVE FLOW MODEL

The second approach is based on Chen's (Chen 2006) work and introduces flow as a design process. Based on the assumption that the player's flow experience is subjective, (Chen 2006) proposes giving the players control over the game difficulty in the form of multiple choices design throughout the game. Instead of letting the control of the difficulty to a computational algorithm that will determine the player's skill level subjectively. Control is a requirement for the flow experience, the players must feel in control over their actions in order to experience flow (Salen and Zimmerman 2003). The sense of control comes from the sense of progression and positive feedback (Nacke and Lindley 2008). Letting the player navigating their own flow experience leads to wider control over the game. In the design of the game, the designer should let the player controls their level of difficulty. In order to design such a game, the designer needs to include a wider spectrum of game mechanics for a variety of levels of difficulty and tastes as it is a game design

approach. Based on the player's tastes, each individual will choose different choices and work at a different pace to navigate them.

In the Figure 0.5, by introducing meaningful choices to the player, they can even have a wider control over the game. The player is now playing a game with a wider spectrum of design with different difficulties and flavour.

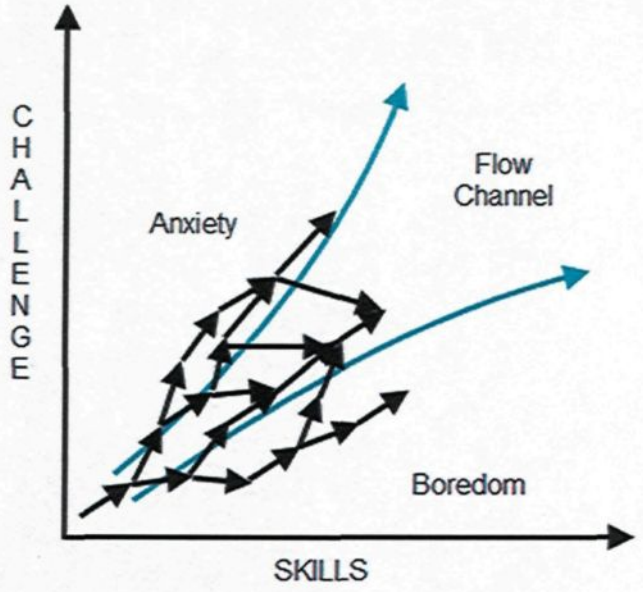

Figure 0.5 : Dynamic flow

Once a network of choices is applied, the flow experience is very much customizable by the player. If they start feeling bored, they can choose to play harder, vice versa. The game should provide a player-oriented active DDA to allow different players to play at their own pace. It consist on designing game mechanics system that allow the player, without noticing it, controlling the difficulty of the game.

In order to adjust flow experience dynamically and to reduce flow noises, the choices have to appear in a relatively high frequency. Although, if these choices appear too often, it might become potential interruptions for the players who are in the flow state. In order to avoid that, the choice of the system must be embedded inside the game core mechanics and let the player makes their own choices through play (Chen 2006). For instance, in role playing style game, the player could experience intense challenges by going right to the boss or choose to explore the environment and power up their avatar which will attenuate the challenge's intensity.

1.9.1 Adaptive flow example

Chen (Chen 2006) applied his model to a game called flOw. In flOw, the player uses the mouse cursor to navigate an organism through a surreal biosphere where it consumes other organisms, evolves, and advances into the abyss as we can see in Figure 0.6. The game mechanics are quite simple, the only action the player can perform is to swim around and eat other organisms in front of its mouth. When eating an organism, the player's avatar gets bigger and then they can eat bigger organisms. The game mechanics are left simple to let the game open to casual and new players.

FlOw is divided into 20 levels. Each level introduces new creatures with new challenges. Instead of forcing the player to have complete the level in order to progress to the next level, flOw offers the power to the player to control their progression. By choosing

different food to eat, the player can advance to a more difficult level and return to the easier level at any time. Also, the player can choose to avoid the challenge, skip the level, and come back later when their avatar is bigger. In flOw, the player can customize the game difficulty naturally through the core game mechanics, swimming and eating. By swimming closer too or farther away from other organisms, and eating different types of food, the player can balance the difficulty of the game. The game had a huge success, after two weeks of online release, it has more then 350 000 downloads and was well received by critics.

Figure 0.6 : Gameplay screenshot of flOw

Overall, to include flow and adaptive flow in game, the game should be intrinsically rewarding and the player is up to play the game. The game offers the right amount of challenges that matches with the player's ability, which allows them to delve deeply into the

game. The player needs to feel a sense of personal control over the game activity. In general matter in order to enhance the experience quality for the player, designer should expand the flow coverage by including a wide spectrum of design with different difficulties and flavours see Figure 0.5. Create a player-oriented DDA system to allow different players to play in their own paces. Embed DDA choices into the core game mechanics and let the player makes their own choices through play. Although, designing a game with embedded meaningful choices regarding the difficulty of play is not a simple task. It is important, from the beginning of the design, to think of the game from a flow adaptive point of view. If the game already exists, it is almost impossible to include adaptive flow. Therefore, this approach is complex to design and limited in its possible application to existing games.

1.10 GAMEPLAY SCHEMA MODEL

The third approach (Lindley and Sennersten 2008) introduces game mechanics based on schema concept. Learning how to play a game is divided into three distinct section: (1) learning the *interaction mechanics*, that is, the basic motor operations required to operate, i.e. the keyboard and mouse. (2) Learning *interaction semantics*, that is, the simple associative mappings from keyboard and mouse operations to in game actions. (3) learning game mechanics competence, that is, how to select and perform in game actions in the context of a current game state in a way that supports progress within a game. Interaction semantics can often be carried across different games within a genre and even across different genre i.e. "w", "a", "s" and "d" to move an avatar; forward, left, back and

right. Learning the interaction semantics represents a form of game challenge by itself (Rolling and Adams 2003). When player learned the basic competence, the focus of learning then shifts to the development of game mechanics competence. Game mechanics competence involves the ability to (1) decode the audiovisual sensory and perceptual information delivered by the game into apprehension of a local situation within the synthesized game world; (2) evaluate this understanding in terms of objectives of play, goals, tasks, the player's state (potions, health and other statistics) and various rewards; (3) make the decisions about which in game tactics and actions to perform next, based on their observations and evaluations; (4) perform the selected action based on competence in interaction mechanics and semantics. In general the player bases their actions on the following logic : sense > model > evaluate > plan > act sequence. The player senses the environment, they model the world so they can evaluate all the actions they can process, they choose one action to plan and then act. That sequence is the basic logistic uses by players, as it is impossible to monitor on a computer, it leads to an understanding of the player's play actions.

In order for the player to pose the right action they need to understand the cognitive structure underlying in the game mechanics. The structure relies on the schema model (Mandler 1984), schemas refer to the declarative knowledge and taxonomical types with their features and relationships, and integrate these with decision processes. More precisely, game mechanics schema is understood as a cognitive structure for orchestrating the various cognitive resources required to generate motor outputs of game play in response to the

ongoing perception of an unfolding game (Lindley, Nacke et al. 2008). It is therefore, the structure and algorithm determining the management of attentional and other cognitive, perceptual, and motor resources required to realize the tasks involved in game play. In general, schema is the semantic representation of knowledge integrated into the decision process.

Schemas can be regarded as mechanisms or algorithms that, among other functions, determine the allocation of attention to cognitive tasks. In the context of game play, attention and the operation of game play schemas are driven by hierarchical goals that set tasks for the player. Schema has the potential to provide both an explanation of the decision and operational processes underlying in game play and an explanation of the detailed reward and motivation factors behind play. In order to process the schema model, statistical patterns of play interaction (mouse movements, key strokes and eye movements) may be use to correlate the presence and execution of specific game play pattern.

To process the schema model, a deep analysis has to be made of the game design, play test levels and player's actions. It could lead to different play modes and abstract hypothetical underlying game play mode. A complete schema description must include all possible subschema and include a way of representing the operation of simultaneous parallel schemas, their relative priority and the principles for switching from one schema to another. When the schema is developed based on game mechanics, it becomes an algorithm

that represents the semantic knowledge needed to perform an action within the game. Therefore, the players' actions can be reproduced as a model algorithm.

Example of a schema game mechanics algorithm for typical adventure game:

```
if(player.sick)
      player.getHeal();
else
      player.Attack();
```

This algorithm is typical in adventure games, before attacking the player checks if the avatar is sick. Based on the data, the player's action consists of either attacking or healing. Using this algorithm a system could find where players make mistakes during play. In this case, if the players never heal their avatars; this could indicate that the players do not understand the meaning of being sick. With this mistake found, the system could intercept it in order to help the players understanding the game mechanics. This help could be meaningful for casual players, such as Alzheimer's patients, it would give them assistance when needed.

In other situations, a schema game play could be used to adapt the game to the player's style (Lindley and Sennersten 2008). The system, by determining what type of players are playing, can introduce elements that the players enjoy based on their repetitive actions such as how the players defeat Non Playable Characters (NPC), how the players drive their car, how the players interact with the menu, etc. It could be used to modify the game's goal according to the type of learners (Magerko, Heeter et al. 2008).

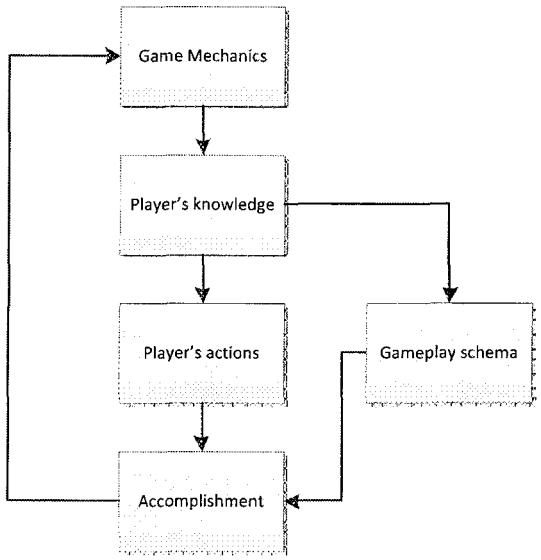

Figure 0.7 : Gameplay Schema abstract application

In general, in order to apply the schema model to a video game it is important to understand the game mechanics. In Figure 0.7, the game mechanics give us the understanding for the player's knowledge to pose their actions. The player's knowledge turning into actions is exactly what the Gameplay Schemas are about. By looking at the player's action in the game and the result of the game, it is possible to find where the player has any problems. Because the Gameplay Schemas do not reach the accomplishment state of the game mechanics, there is a problem. The following pseudo code gives us hint on how an algorithm could adapt and give help to the player when in need.

Pseudo code mechanics algorithm to recognize mistake:

```
If (NOT GameMechanics.certainAction.Need != player.items)

      return problem ("Player is missing the basic game
            mechanics needs")

Else If (player.Actions != GameMechanics.certainAction.
      result)

      return problem("Player does not fully understand
            the actions required")
```

The pseudo code takes into consideration that the game mechanics are well understood. In the pseudo code, *GameMechcanics* refers to structure that defines all the game mechanics inside the game, *certainAction* refers to an action i.e. unlocking a door. the last variable in the structure is *Need*, it refers to the prerequisites the player needs; a key to unlock the door. First of all, it is important to check if the player has the prerequisite to accomplish the action. In the case they do not have all the requirements to fulfil the game mechanic, the algorithm as to point out to the player that something is missing In the pseudo code, it sends a message that the player is missing something, it could say the key is missing; it could be into the form of an hint or tip. By doing that, the player can then fully enjoy their actions and understand what they are missing. Some game mechanics have prerequisites and others do not, so it is important to fully understand the repercussion of the game mechanics. In the second case, if player possesses everything needed to process the action, we check if the player succeed to accomplish or not the action. In the case they do not, it is important to understand where the player has issues. The system can then give hints to the player on how to succeed or change the form of challenge to an easier one.

A major issue with the game play schemas' model is that it has never been implemented nor tested (Lindley and Sennersten 2008). Moreover, it seems that it would be very time consuming to implement. In order to understand the players' interactions with the game, the designer would need a lot of case studies and introducing mistakes in that process could be easy. Although, this system could be really useful to narrow down the difficulty, but show a lot of limitation to the case the difficulty is too easy. Also, it is a case by case study, it cannot be seen as a general approach to adaptive game mechanics, that can be use in a generic matter.

1.11 ADAPTIVE MODEL BASED ON FRUSTRATION FEED

The last approach uses frustration in a game, it is something that every player has experienced at least once. Playing video games can sometimes be somewhat frustrating. For example, repeatedly failing to defeat a particular enemy or wandering aimlessly around a maze-like dungeon looking for exit. Frustration arises when the in-game progress towards achievement is impeded (Gilleade and Dix 2004). It is a negative emotion and if monitored for can be used to indicate when an user is in need of assistance. Video games often require the player to inputs time-sensitive command sequences in order to gain a tactical advantage over their opponents, for example the command in Figure 0.8, when succeeded, produces a fireball in the game Street Fighter 2.

When the players are unable to complete a command; they become frustrated as they are not able to progress. Furthermore, the players become frustrated when they cannot

complete a certain challenge in the game due to a misunderstanding of the game challenge. For example, in *Metal Gear Solid,* a sniper rifle is required to defeat Sniper Wolf. The player is told to search for this weapon, but is not told its location upfront. There is no means to resolve this objective unless the player wishes to search the entire game world for the rifle. Exploration such as this can lead to frustration if not resolved within a reasonable time frame.

Figure 0.8 : Fireball manoeuvre in Street Fighter 2

Using DDA, based on the player's frustration level, the system could change the game difficulty or provide help to the players when they are experiencing frustration. For example, the game system could change the width of a hole the players have fallen into several times and is starting to experience frustration, so at their next attempt at jumping over the hole, which is now narrower, they would be successful and the players would be less frustrated.

The idea when using frustration is to make sure we do not let the player get too frustrated which could lead the player giving up the game. In Figure 0.9, if it is possible to read the frustration level of the player, when they reach the first level of frustration the system starts giving the player some hints on how to accomplish the next step of the game or game mechanics.

If the hint given to the player does not work, it may cause the player to be more aggressive. The following step would be to help the player by showing them the path to take for example. The idea is to help the player by giving them not just hints but to take part in their game actions and to make sure they can find they way around. If giving help does not help to calm them, the last step would be to adapt the difficulty or to make the game easier so the player can relax and enjoy the game better.

The major issue with this model is the detection of frustration. Frustration can be measured using blood pressure, heart rate and conductivity (Gilleade and Dix 2004). These measurements are related to the level of arousal. From a commercial point of view, using those measurements is almost impossible; the only existing direct connection between the players and the game is the gamepad including mouse and keyboard. Some research (Sykes and Brown 2003) indicates that it is possible to measure the players' level of arousal by monitoring button pressure on the gamepad. However, this idea has never really been officially used as an adaptive game mechanic. The model as shown only works one way, the system can only decrease the difficulty of the game, it cannot increase it. In the case the

player is looking for a challenging game, a system base on frustration would only decrease the challenge. The player might need that frustration to fully enjoy the challenge and the rewarding for accomplishing a hard challenge.

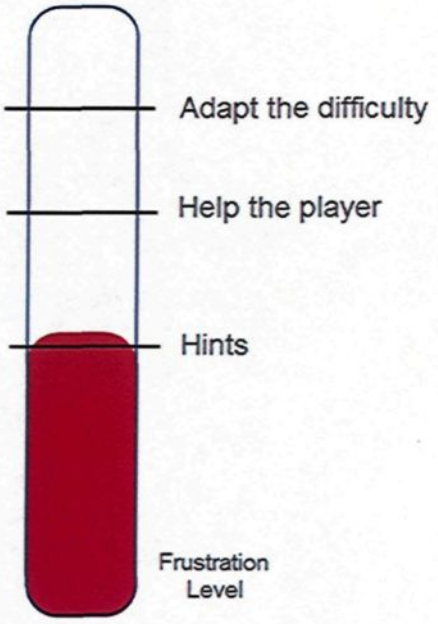

Figure 0.9 : Using frustration in game

1.12 Assessment and Conclusion

This chapter had for goal to introduce the reader to the problematic about adaptation in games and to present the related work about the major approaches on video games adaptation. This research tend to address the following question: is it possible to enhance the player's experience in game using DDA system in real-time? As seen, this issue had been address by several researchers (Gilleade and Dix 2004; Hunicke 2005; Chen 2006;

Lindley, Nacke et al. 2008; Magerko, Heeter et al. 2008; Tremblay, Bouchard et al. 2010). The model from Hunicke presented the base of any computable DDA system, the system as to understand the player's flow state in order to change the difficulty. Chen presented a general design approach that gives the player more control over the game and its difficulty. Lindley uses the game play schema to represents the player action into an algorithm that when broken represents where the player needs the game to adapt. Gilleade and Dix uses the frustration level of the player to understand when the game needs to be adapt to the player. All of those researches presented certain limitation, the model being too strict, impossible to add to certain existent games or never been implemented. To encounter those limitations, this research uses some idea presented by (Gilleade and Dix 2004) by simplifying the model so it could be use by a wider style of video games. This research also applied the basic idea of adaptive flow presented by (Chen 2006), although this research model does rely on it as it limits the design process. The propose model in this research applies the game play schema model (Lindley and Sennersten 2008) in order to find deadlock in the design of the game, although integrating the model into our research would have been interesting. Even more, The idea of repetition leading to frustration (Gilleade and Dix 2004) is also taken into consideration, as it showed some limitation it is not a big part of the propose model. The following chapter introduces our contribution: a computational model for DDA using rating system. The propose model is a complementary design approach to (Chen 2006) and a reciprocal approach to (Hunicke 2005).

CHAPTER 3

AN NEW MODEL OF DYNAMIC DIFFICULTY ADJUSTMENT (DDA)

From the player's perspective, a game relies on the fact that it is fun and relaxing. Is it possible to enhance this sentiment of relaxation and fun using a DDA system? This proposal is inspired by the previous works presented in the last section and consists of redefining a new computational DDA model while introducing different layers that can be threaded independently in order to enhance the different control the players have over the game, as Chen (Chen 2006) points out regarding the design of flow in games. To represent the players' experience, the players' tactics and strategies in the game were taken into account. The word tactics refers, in military science, to a short term calculated decision, for instance an action, a move or a manoeuvre against an enemy. On the other hand, the word strategy refers to a particular long-term plan for success. Using these words to define the two model layers demonstrates an effort to represent and respond to both the players' short term behavior or actions (tactics), and to the overall gaming experience (strategy). To portray these two specifications, we designed two algorithmic layers that can be used independently and/or congruently. The first is called the tactical layer and the second the strategic layer. The tactical layer uses small feedback variables that are easy to adjust that follow the players' fast actions with ease. It allows us to quickly respond to the players' immediate situation. The strategic layer is used to understand the player's skills in the long run. Therefore, putting these two layers together allows us to have a better understanding of the players' experience over both a short and a long period of time in the game, which provides us with a dynamic adjustment approach that works on two different layers.

The following section is organized as follows. In section 3.1 we present the tactical part of the model, followed in section 3.2 by the strategic portion. Thereafter, we present in section 3.3 the implementation of the model taking the form of a small calculation/combat game, named Number to Number. This section clearly portrays, with some examples, how we implemented the model in the game.

1.13 TACTICAL LAYER

As previously mentioned, the tactical layer is used to understand a small and/or short action that the players perform in the game and to adapt to the results. The tactical layer uses an abstract level of difficulty representing the players' skills. The players' experience evolves during play, for a second they may be on top of their game and the second after be less than average. We need a way to represent how the players evolve during game play. So the tactical layer exploits two main concepts. First, the players have a level of intensity, in other words the players have a level of difficulty that matches their skills best for a particular moment. During play, everything evolves, moves and changes, so the player could be at a certain intensity one second and completely different one the next. Therefore, the second main concept refers to the use of a certain desired level. We could represent it as something mapping the players' experience. Since their experiences and skills are not necessary constant, the players' desired level does not evolve in a linear manner.

On the first hand, we need a way to evaluate the players' experience. Their level of difficulty; it's an abstract representation of the intensity of the game. Let us call this abstract level *levelIntensity*, based on the idea that we could translate it as a variable, a certain variable that represents the players' intensity should evolve in a way that understands and represents the players' experience. So the game needs to do something for every value that *levelIntensity* could encompass. When the *levelIntensity* increases, the game should be harder in some way based on that idea that we need to plan a chart for every level of intensity.

Table 0.1 : *LevelIntensity* chart

levelIntensity	**Difficulty**
1	low
2	medium
3	normal
4	high
5	intense

As we can see in Table 0.1, according to the value of *levelIntensity*, the experience is different for the players. These difficulty levels are just an example, but they speak from themselves, depending on the variable value, the difficulty changes for the player. Some games can use anywhere from three to twenty levels according to the game design. In the tactical layer, we have to make sure to implement every level of intensity. There are two possibilities; first we make the difficulty procedural, which can be done by implementing the difficulty using variables. The level of intensity changes the variable of the

implemented difficulty. On the other hand, sometimes it is impossible to make the difficulty procedural. The second possibility would entail every level of intensity being implemented manually, case by case. There is a limit between every level of difficulty that is distinct. It is important to play test all along with both techniques. See section 1.15.2 for an in depth example.

1.13.1 Level Intensity adjustment

How do we control *levelIntensity*? The value of that variable is adjusted according to the players' skills and performance in game. The way to keep track of the players' performance is by analyzing the game mechanics, what the players really do in the game. For example, in a calculation game; they answer math questions, in a racing game; they drive cars to win a race, in puzzle games, they solve problems, etc. We can always find the players' basic interaction with the game. It is important to base the value of *levelIntensity* on the players' interactions with the game.

The players learn from the game mechanics, not from random experiences in the game. With the mechanics in mind, we need a way to evaluate the players' performance. It is time to introduce a new concept, *nowExperience*, which is used to evaluate the players experience. When the players are really good at the game, we increase the value of the *nowExperience*, when it hits a certain value, 10, for instance; we increase the value of *levelIntensity* by one. On the other hand, when the players are having a hard time playing

the game, *nowExperience* will decrease, when it hits a certain value for instance -10; we decrease the value of *levelIntensity* by one. So as we can see, *nowExperience* is meant to be updated quite quickly and *levelIntensity* depends on its value.

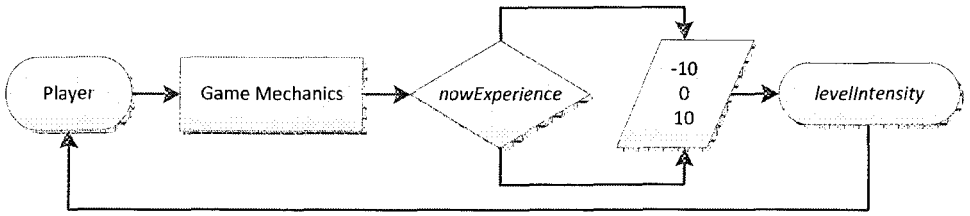

Figure 0.1 : Tactical layer flow chart

Figure 0.1 represents how the model works in a general way. The players play with the game mechanics, according to the result of the game mechanics, *nowExperience* will change. If the action result is positive we increase *nowExperience* and decrease it when it is negative. When the value of *nowExperience* is 10 or -10, it will have an increasing or decreasing influence on *levelIntensity*; that would have an impact on the game mechanics. A way to express how *nowExperience* should vary is to base our analysis of the players' data for every instant of the game. Imagine the players are playing a fast puzzle game, their *levelIntensity* is 2, so we know they are doing well in the game. They are solving all the puzzles very quickly, for example 2.4 seconds per puzzle. We know, from analysis and observation that at that level of intensity a puzzle takes 3 seconds in general to be solved. Since the players are succeeding in the puzzle, we need to increase *nowExperience*. The following formula expresses the *nowExperience* behavior:

$$nowExperience \mathrel{+}= succeed\ (+)\ or\ (-)\ averageTime\ /\ timePlayer \qquad \textbf{0.1}$$

By dividing 3 by 2.4, we obtain 1.25 and increase *nowExperience* by that number. Therefore, when the players have a negative answer, we decrease the *nowExperience* by one as seen in formula 0.1. So that way, we can follow the players' experience and adapt to it. When *nowExperience* is over 10, we will increase *levelIntensity* by one, making its value three. As the *levelIntensity* changed the value of *nowExperience* is 0. According to the players' performance, *nowExperience* will be changing again. Although, it is possible to base our evaluation of *nowExperience* on other variables, for instance, the number of gold collected, the period of time without being hit by an enemy, the amount of ammo available, etc. the important thing is to base the evaluation of *nowExperience* on the game mechanics.

This layer of our approach can be applied to several parts of the game. According to what the designers want to emphasize, they could make some parts adaptive and others not and design them in a linear way. For instance, imagine a shooter game, the way the players get ammo could be based on the tactical layer, so if the *levelIntensity* is 2 and it stays at two it would take 4-5 shots for the players to defeat an enemy. If the players can do this with only two shots, *nowExperience* would increase until it hits 10, then their *levelintensity* would be 3. The way the ammo is distributed on the map and the bonus ammo pack follows the *levelIntensity* value. At the value of 3, the system is expecting 3-4 shots to take down an enemy and the ammo and health packs would be rare. This way, the players will experience a game that is harder and more challenging for their skill level based on flow theory. If the

level of intensity increases quickly, the players could experience anxiety and failure. Although failure is really important in game, as Juul (Juul 2009) pointed out : "failure is central to the experience of depth in game, to the experience of improving skills." Therefore in order to master a game, its content and mechanics, it is important to experience failure at some point.

1.14 STRATEGIC LAYER

The tactical layer is really useful for analyzing small sections or actions in the game, though it is not very useful in understanding the players' whole experience in a general way. The strategic layer is not so different than the tactical layer, but it leads to different results. It is especially powerful to evaluate the players' overall experience. The strategic layer exploits, in fact, a rating system. The main idea is to rank the player, like we rank tennis, football and chess players. By dynamically ranking the players we know how experienced they are in the game and how they are evolving. Their ranks could be used to control the difficulty in the game. For example, we could base the difficulty level of the artificial intelligence according to their rank.

First, it is important to determine how the ranking system works. The ranking system should be dynamic and viable. In order to complete this task, it is important to look at different approaches to rating players. Instead of designing a new approach to the rating system, it is possible to use one that already exists (Coulom 2010): ELO, Harkness, Glicko,

etc. They are widely used in the competition world of chess and football, for instance. In our system we decided to use ELO for its simplicity and its ease to implement.

1.14.1 ELO SYSTEM (RANKING THE PLAYER)

How do we predict the result of a match based on statistics? Arpad Elo designed a way to rank players in 1939 to answer that question (Coulom 2010). He proposed to use the normal distribution function, *FN*, to predict the result of a match. With two players ranked *R1* and *R2*, the expectation value is given by *FN(R1-R2)*. The greater the difference is between the players, the higher the expectation value for the highest ranked player. A game can have three value as a result win = 1, draw = 0,5 and loose = 0. These values are given to the player that wins or loses or both in the case of a draw. The expectation value expresses that number, for example, a player with a ranking of 1400 playing against a player ranked 1600 has an expectation value of 0.18 for *R1*. Based on that, after the match the ranks of the players are incremented or decremented according to the result. If the players have a better result than expected, their rank will increase and if their result is worse than expected, their rank will decrease. The following formula expresses the rating variation:

$$RA = RA + K \, x \, [SA - E(A)] \qquad \textbf{0.2}$$

As we can see in formula 0.2, *RA* is the rank of the player *A*, *RB* is the rank of the player *B*, *SA* is the score obtained for player *A* (1 = victory, 0,5 = draw, 0 = lose). *E(A)* is

the expectation value of player A; $E(A) = Fn(CA\text{-}CB)$ where Fn is the normal distribution function. K is the update coefficient. For example, K could be function of the number of games during the career of player A, $k = 30$ if the players have less than 30 games, $k = 15$ if $CA < 2400$ and $k = 10$ if $CA > 2400$. K is the volatility of the player.

The ELO system is still used today as the rating system by the World Chess Federation. ELO is used in our project for its ease to be implemented, its simplicity and the rating possibility. We could have used Bayesian based algorithms that are more reliable to find the expectation value of a match based on the rank (Bolstad 2007). A Bayesian approach is harder to implement and control than the ELO algorithm and it required more process time and resources, therefore, it was not implemented.

1.14.2 USING THE PLAYER' RANK

Based on the ranking of the players, we can design our AI input. Using ranking system we know that normal players will be around 1400, very good players around 1800 and weak players around 1200. So it is important to construct the AI ranks based on the players ranking. As the players want to fight AI around their difficulty level, it is important to give them what they are expecting. Therefore we can obtain the AI ranking level based on the player's ranking level.

$$RankAI = rank +/- 200 \times rand(0,1) \qquad \textbf{0.3}$$

Using that formula 0.3, it possible to have sometimes a stronger, weaker or normal AI. By controlling the AI ranking level we have control over the difficulty; it is possible to control the AI difficulty based on game mechanics too. AI difficulty is based on the game mechanics, they offer a challenge to the players by introducing conflict, like in a racing game, the faster and quicker the AI is, the harder the game is or in a FPS a strong AI will have a fast response to the players' actions, better accuracy, more health, etc. The idea is to find the layers of the difficulty and implemented those dynamically using the players' rank. Imagine we are building a RPG, we can give an adaptive challenge to the players using the AI ranking. Imagine a certain monster has an attack of 10 as normal, but the players have a really high ranking, so they are stronger than usual. To make the game challenging to the player, we could add a bonus to the attack of the monster.

$$Bonus = rankAI/1000 \times 2 + rand(0, rankAI/1000 + 1) \qquad \textbf{0.4}$$

As we can see on formula 0.4, a monster gains a bonus according to its ranking that is based on the players' rank. So imagine the players have a rank of 1400 the AI 1450, the bonus for the attack will be between 2.9 and 5.35, with a basic attack of 10 that could end up being a really interesting challenge.

The flow chart seen in Figure 0.2 leads us to a better understanding of how the strategic model works. Every time the players perform an action that matters to the ranking, the result will influence the ranking system as well as the players' rank influences the

intensity of the game mechanics. This model is really useful in understanding the players' experience. Therefore it could be used in any kind of game. Even the strategic rating system would be useful for online gaming where you could have players of the same skill levels playing against each other. In general, it is a strong and easy approach to implement that leads to dynamic difficulty in games.

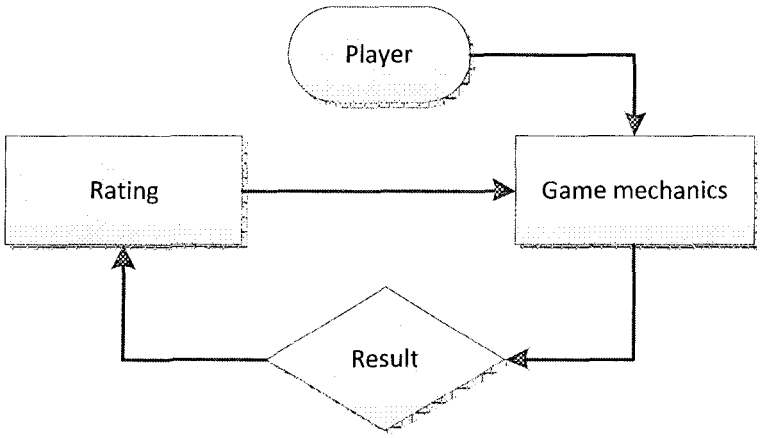

Figure 0.2 : Strategic layer

1.15 IMPLEMENTATION OF THE MODEL: NUMBER TO NUMBER

We implemented the new adaptive model we just presented in the form of a simple calculation/fighting game named Number to Number. This game is in fact a version 2.0 of the former Number to Number Combat game, presented in [8], which implemented an earlier version of our adaptive model. It received good comments from the players for its design in the testing phase we did with the first model so, by choosing to keep the same basic game concept, we made sure to put more effort into the players' experience based on

the adaptive game mechanics and not on the design approach for enhancing their experience. It was really important for us to start with a design that was already interesting for the players.

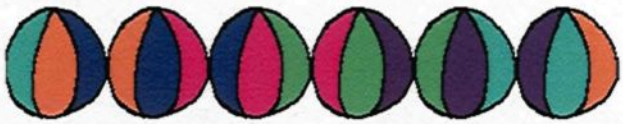

Figure 0.3 : Spritesheet example, minion level 3

We made the game using the open source Integrated Development Environment (IDE) *FlashDevelop* who is base on the MIT license. This editor allowed us to use the Software Development Kit (SDK) *Flex* released by *Adobe Systems* for the development and deployment of cross-platform rich internet applications based on the *Adobe Flash* platform. Flex was developed for programmers that found it challenging to adapt to the animation metaphor upon which the *Flash Platform* was originally designed. It allowed us to use websites that could utilizes streaming video, audio and a whole new set of user interactivity for online gaming. The IDE is an easy to use editor with syntax highlighting, bookmarks and tasks. It also includes a comprehensive find and replace dialog. The interface is intuitive and with very flexible panel. It is possible to add open plug-in based architecture. The IDE supports ActionScript 2 and 3, MXML, HaXe, XML and HTML. In general *FlashDevelop* is a very handy IDE that has been a great tool in order to complete this project.

The art assets were made using *Adobe Photoshop CS5*. *Photoshop* is a graphics editing program that uses image layers system. A home script program has been develop, in order to compile the layered animations into spreadsheet. As we process all frames of the animation in separate layers, our script program was able to take all layers and distributed it as seen in Figure 0.3. Inside the video game, we just had to load the graphic. The first frame is then read by the reader (a rectangle the size of a frame that just show a part of the graphics), after a certain amount of time the reader will read the following frame until the end of the graphics. At the end of the graphics, the reader will come back at the first frame which is going to give the impression of an animated object to the player because the time between frames is quite short. In general, the tools uses were really handy and easy to use.

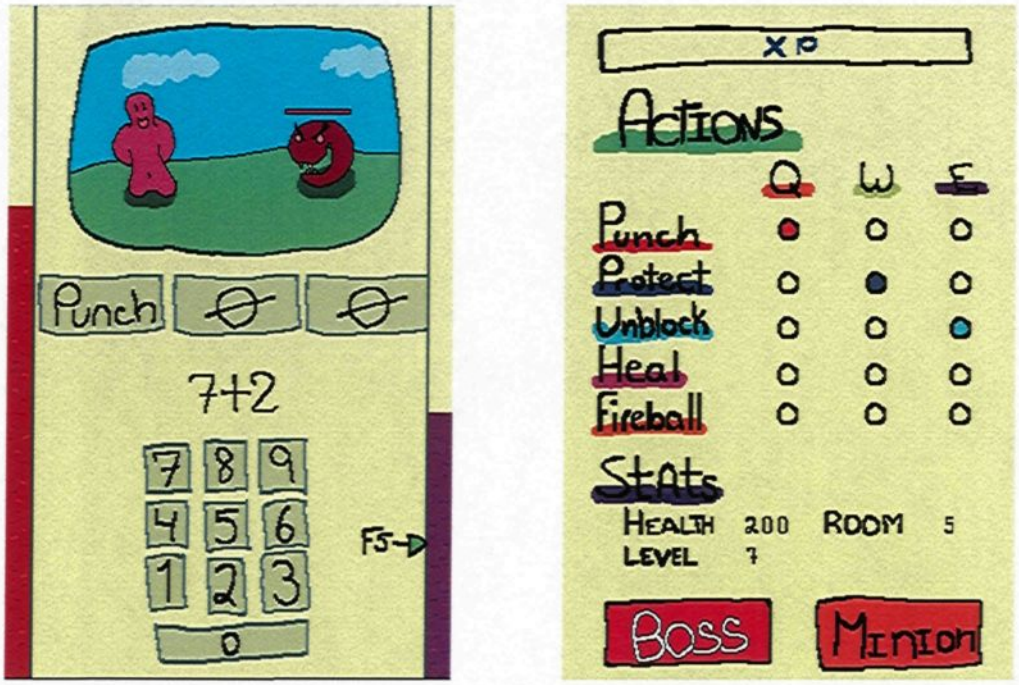

Figure 0.4 : (a) NTN game play (b) After combat screen in NtN

Figure 0.4 shows the interface of Number to Number (NtN). The game is quite simple, but some twists have been added to make sure the players do not get bored. We can see the game play screen on Figure 0.4 (a), the goal of the game is to answer arithmetic questions in order to perform actions and win the combat against the opponent. Every correct answer fills up a bar on the right hand side on the screen see Figure 0.4 (a).

The purple bar is the players' power, as you can see there is an arrow pointing to the power bar, that arrow is followed by F5. In the game F5, F6 and F7 correspond to the actions the players can perform; in this case F5 refers to "Punch". It means that if the players want to perform that particular action, the level of power should be above that particular arrow. Those actions are situated over the arithmetic question, in this example, the first one is "Punch" and the two others are empty. These actions could have been "Heal", "Fireball", "Protect", etc. Therefore the players have perfect control over the actions that they want to perform. The basic idea of the game is for the players to beat the AI before it beats them. The players' health bar is the one on the left hand side of the screen and its color is red. The monster will be attacking and using magic that will have different effects on the players' avatar. The players lose when their health bar is empty and win when the AI's health bar is empty; it is located over the head of the little red monster on Figure 0.4 (a). After every positive combat for the players, they will gain some experience. When the players have enough experience points, they can level up. When levelling up the players will be able to choose from a variety of new actions or upgrade the actions they already have.

Figure 0.4 (b) represents the screen after a combat. The players can choose the actions they want to use for the next combat. The punch and fireball are used to attack the enemy. Protect and heal are use to protect themselves against the enemy. In order to move to the next room, the players must fight against the boss. The game is over when the players fought the calculator; the last boss in room 5.

1.15.1 INTEGRATION OF ADAPTIVE FLOW

The game is separated into 5 rooms; each room has a minion and a boss to fight. The minion is always easier to vanquish than the boss. Every fight gives the player experience points, at every level the points needed to reach the next level gets higher. When the player gain a level, they have access to a wider sceptre of magic and strategies. Getting a magic at the time helps the player to understand and master the game, they do not feel overcome by the game mechanics they have to learn. They start the game and they have to understand the mechanics of answering math question in order to increase their power and when they have enough power, they can attack their foe. Learning how to heal and protect themselves at the same time of learning the basic mechanics would have lead the game to fail entertained the player.

Leaving the choice to the player to fight between a minion or a boss left a bigger expression to the game. The players can choose between fighting the boss directly for a greater challenge or preparing for their fight against the boss with a minion by levelling up their character. As you can see in Figure 0.5, when the game starts, the player have the

choice to build up their character by fighting the minion (1) or they could go fight the boss right away to have a bigger challenge (2). If they choose to build up their character (1) the fight at (3) is going to be less challenging then the one in (2). As you can see the player have access to a colourful expression of difficulty. All the fight in the game leads to more experience to the player and then levelling up (4).

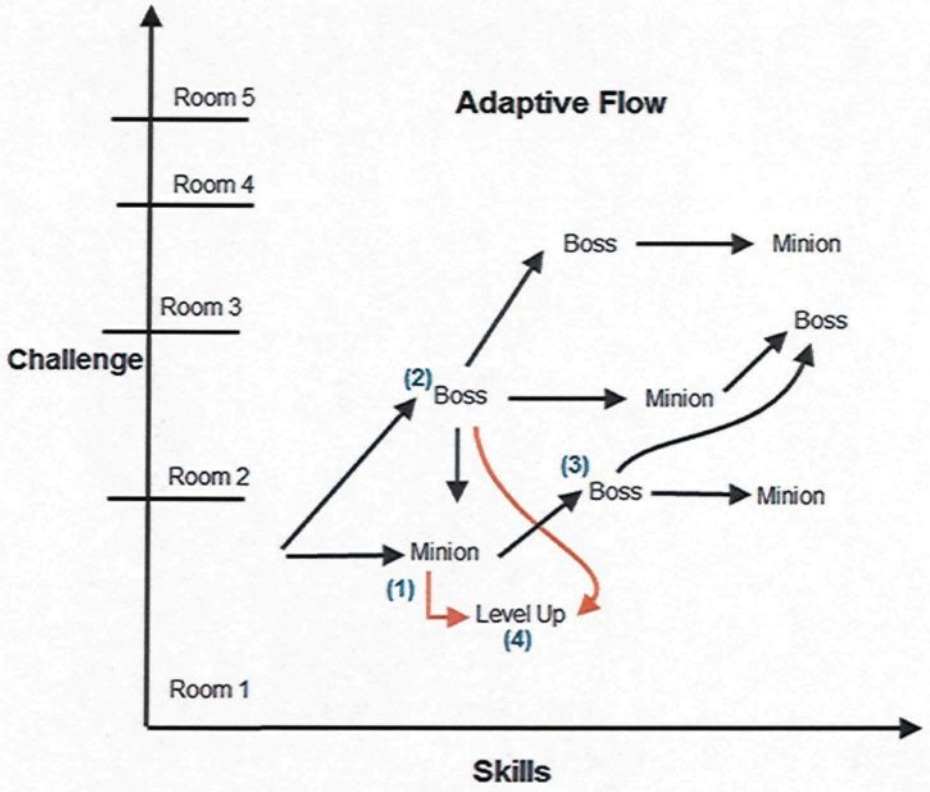

Figure 0.5 : Adaptive flow apply to NtN

Although the DDA system makes several little changes in the challenge of the game, the basic idea still is the same, some enemy are design to be fought with a certain

magic, if the player does not have it, it is still possible for them to succeed even though it increases drastically the challenge of the enemy. In general the possibility for the player to explore the game difficulty as they want give them a better control over their experience.

1.15.2 CORE COMBAT MECHANICS

As explained previously, we needed to implement our adaptive mechanics in accordance with the game interactions. NtN is a combat game using arithmetic questions, so the main game mechanic is the arithmetic questions. Based on that, we implemented 5 levels of difficulty, so 5 different levels of question difficulty, though the answers for every level are within the range of 0-9 on the keyboard.

As a part of the review we received regarding the first version of Number to Number Combat, players were expecting harder questions from the game. Table 0.2 represents the different intensity the arithmetic questions can attain.

Table 0.2 : Tactics layer apply to NtN

levelIntensity	*Difficulty*	*Question*
1	low	1 + 3
2	medium	3 x 2
3	normal	3 + 10 - 4
4	high	(20 + 2) / 11
5	intense	3 x 10 - 23

To respond to this issue, we implemented harder questions imbedded in the adaptive game mechanics. As the players start a new combat, they have a *levelIntensity* of one. At this point, the questions are pretty simple. As the players answer questions, the *nowExperience* value follows the players' behaviour according to the time took to solve the questions. For every question we have an average time to compare the players' time to.

Table 0.3 : Timing for questions

levelIntensity	**Time (*s*)**
1	1.2
2	1.8
3	2.4
4	4
5	4.5

Table 0.3 expresses the average time required to solve the level question. For instance, if the players have a *nowEperience* equal to 8.4 and a *levelIntensisty* of 3, they answer the question 4 + 9 – 5 in 1.8 seconds, then *nowExperience* will increase by 2.4/1.8 = 1.3. If they succeed in answering the next question, then the level of intensity will increase. As the questions get harder, it takes longer for the players to answer them, therefore the challenge becomes harder.

1.15.3 DESIGN OF THE ARTIFICIAL INTELLIGENCE (AI) OF NTN

The AI of NtN is designed based on the dynamic adaptive model proposed in this research. First, we needed to find what makes a difficult AI. The difficulty should feed off

the players' rank. In NtN, the player is fighting a monster in a RPG way. Therefore, the AI is defined by: the timing, rate, variety and power of the monster's attack.

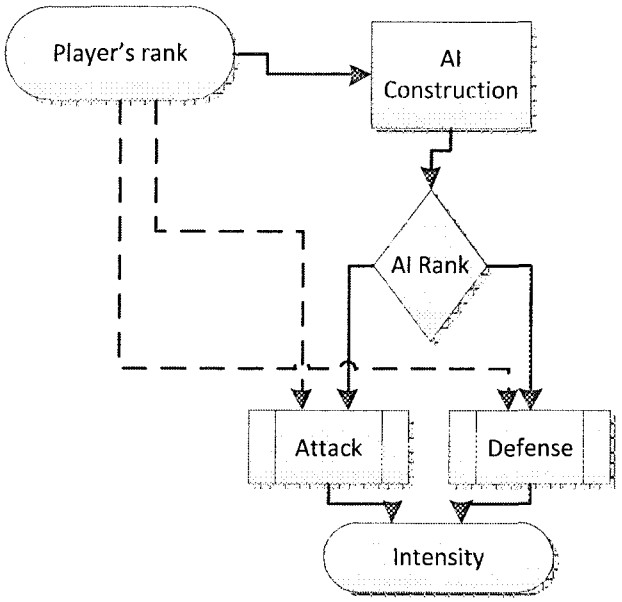

Figure 0.6 : Working AI model

Figure 0.6 represents the basic idea behind the AI design, based on a ranking system. The AI is built based on its rank and the player's rank. The AI's actions are derived from the player's rank too. By using both players and AI rank, we make sure the AI can rapidly adapt to the player's level of intensity.

All opponents are designed differently; the basic monster that we are going to use, for example, is the red monster in Figure 0.6 (a). This opponent only attacks the players; it waits for a period of time, according to different variables and will hit the players. Its attack power is based on the player's data; maximum health, level, rank and number of victories.

First, it is important to determine the rank of the AI, the rank is based on the following formula 0.5 based on the formula 0.3. The formula 0.3 gives us an major idea as each game is different, each AI development will be as well. The constants use in the formulas are from tweaking the system and play testing.

$$RankAI = playerRank - T + rand(\ 0\ ,\ 100\ )\ x\ (nv + 1) \qquad \textbf{0.5}$$

Where, T (0.6) is a random function based on the number of the player's victories in a row.

$$T = rand(\ 0\ ,\ 30\ )\ x\ nv \qquad \textbf{0.6}$$

In formula 0.5 and 0.6, *nv* is the players' number of victories in a row. The number of victories is used as positive feedback (Salen and Zimmerman 2003), where the new monster encountered is going to be a lot harder every time the players win a match. However, it is sometimes important that the players, even if they are getting stronger, play against weaker enemies, so they can build confidence. Then, the enemies could be weaker one third of the time. The rank is based on the following formula 0.7.

$$rankAI = playerRank - T \qquad \textbf{0.7}$$

The players will encounter weaker enemies according to the formula 0.7. When the AI has ranking, it is now time to find its power and how long the time lapse between its attacks will be. The power of the attack is given by the following formula 0.8.

$$Power = rand(0,1) + rand(0, rankAI/600) + min(nv/5, 5) + levelPlayer \quad \textbf{0.8}$$

RankAI is the monster rank determined by the formula 0.5, *nv* is the number of victories in a row and *levelPlayer* is the rank the player is currently at. We add the *levelPlayer* to the formula because its attack is based on their level. The other part of the AI faculty is time between attacks and the behaviour is given by the following 0.9.

$$timeCheck = rand(0,T) + 3 - min\ (nv/3,3) + timeAverageMath \quad \textbf{0.9}$$

The variable *T* could be anywhere between one and three if the AI has a higher rank than the player. The value of *nv* is the number of victories the players have and *timeAverageMath* is the average time for the math question level the player is at, expressed in table 3. To make the attack more dynamic, if the players have energy over 15 and have more than three victories in a row. *TimeCheck* is given by the following 0.10:

$$timeCheck = rand(0, 2.5) + 1 \quad \textbf{0.10}$$

The time between attacks is very important 0.10; it changes the player's experience a lot by changing the difficulty of the game, a faster monster is a much harder monster to fight. By putting all those formulas together we constructed a basic AI design for our game. This design process was then used to build different AI characters with different actions, for instance, the calculator can hide from the players.

Generally speaking, the game is pretty easy to play and all those adaptive mechanics are hidden to the players so they cannot understand how it works and change the way the game evolves. NtN had pretty good review during the testing phase, the players really enjoyed the new twists added to the mechanics and the adaptive mechanics gave the players some real epic play sessions.

CHAPTER 4

EXPERIMENTATION, RESULTS AND DISCUSSION

Our primary goal with this game was to enhance the overall players' experience. However, simply implementing a game using our adaptive approach does not show if our model is working or not. The model needs different play testing with different players. Therefore, we decided to design an experiment to put our model to the test. The experimental protocol was quite simple. First, we asked our relatives and friends to try our game and we received immediate feedbacks, that were used to correct bugs and improve the general experience of the game. When the prototype was free of bugs, we sent an email to all the computer students on campus, they were asked to go on a website where they read all instructions about how to play the game and to fill up a questionnaire. We ended recruiting a group of 32 participants for the test, mainly students. The participants were solicited to go online and play the game. At the end of their playing session, they were asked to complete an online survey on their gaming experience. This survey was carefully designed to give us the needed clues about the performance of our model and the evolution of the difficulty level. Finally, the survey had, at the end, an open section where participants could write notes, comments and suggestions about the game experience. We conducted approximately 150 play sessions involving 32 players. Each of them answered the survey.

The questions put the emphasis on the players' experience and their satisfaction with the difficulty of the game and the game in a general perspective. There were three parts to the survey: what type of players they considered themselves, what they thought of the game, and what they thought of the difficulty in the game. The survey took two to three

minutes to complete and it gave us a good comprehension of the players' experience during their play session of NtN.

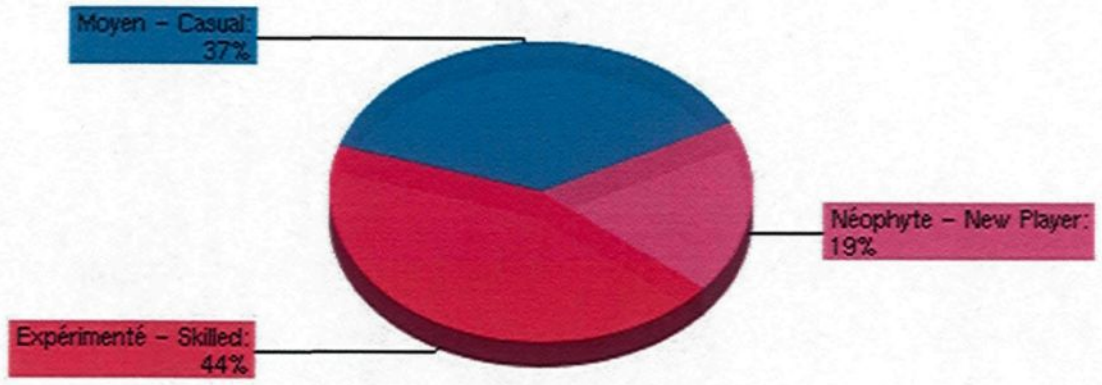

Figure 0.1 : What kind of player are you?

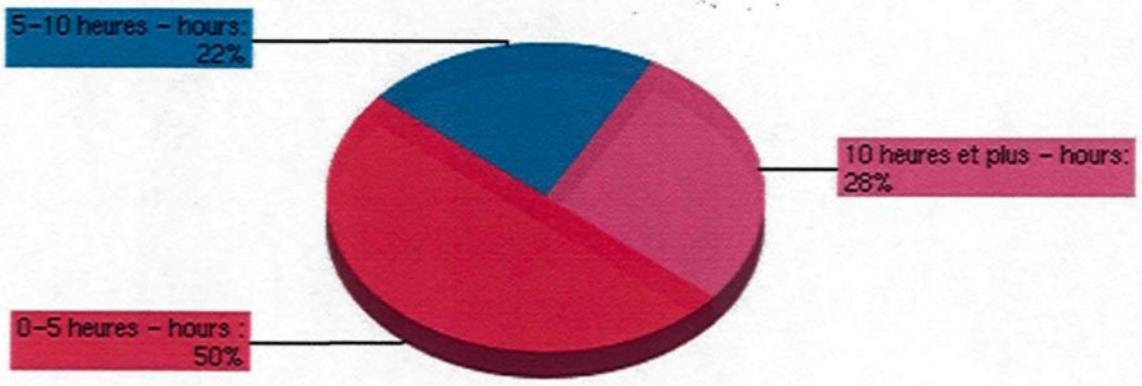

Figure 0.2 : How many hours do you spend playing games?

The first question they were asked was what kind of player are you, see Figure 0.1. As we can notice, most of the player were experimented 44% (14 out of 32). As most of computer science students study video games, we were expecting that kind of result.

As most of the testers are experienced gamers being students with not that much free time, they probably cannot play games as much as they want. In Figure 0.2, we can see that 50% (16 out of 32) spend less than 5 hours per week playing games, but also 50% of the testers spend more than 5 hours per week playing. We think most of our testers play games in a regular basis. Hence, we could say that our testers know how to play games and are really interested in playing games.

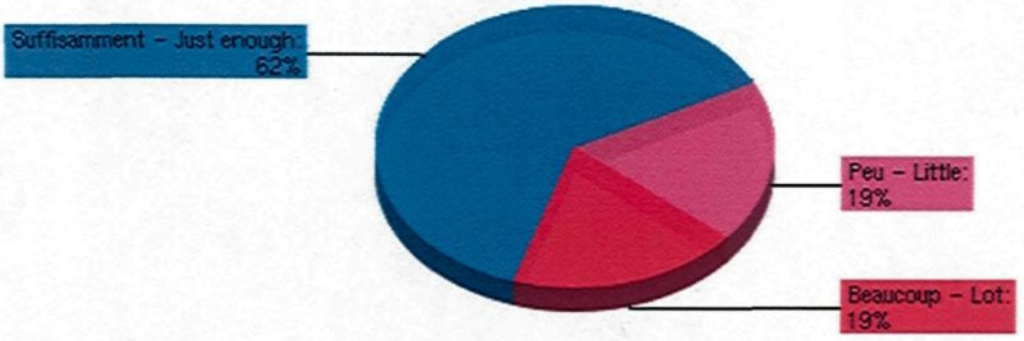

Figure 0.3 : Did you have fun playing NtN?

The third question mark the new section, looking at the in-game player's experience. In Figure 0.3, we can notice that 62% (20 out of 32) of the tester had just enough fun playing NtN. This is a promising indication that the game matched the players'

expectations. That means they enjoyed the game, but it does not tell us directly if it is because the game employs DDA.

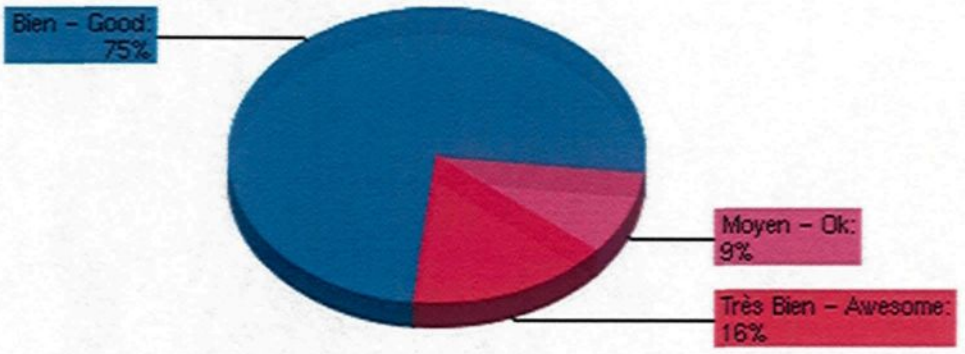

Figure 0.4 : What is your general appreciation of the game?

The fourth question of the questionnaire was about the general appreciation of the game. Here we are looking at how the player seen the game in general, most of them thought the game was good 75% (24 out of 32), which means the concept of the game and the general experience of the game was well received by the players.

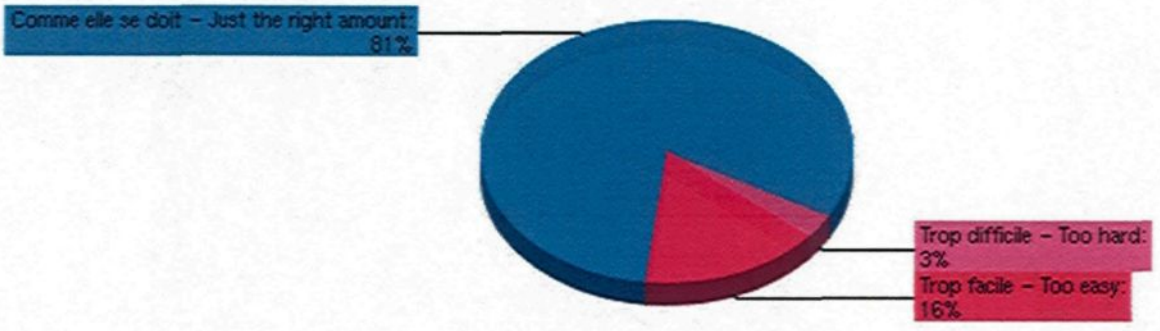

Figure 0.5 : According to you the difficulty of the game was?

Figure 0.5 is a really important chart for our research. This question asked the player to quantify the difficulty of the game. As said before, fun occurs when players can experiment flow state. Flow state can occurs when the game challenge matches the player's skill, in other words when the difficulty is not too hard or too easy for the player. The main goal of our DDA system is to give the player just the right amount of difficulty any kind of player needs. This is clearly achieved as 81% (26 out of 32) of our testers said the difficulty was just difficult enough, see Figure 0.5. It leads us to think that the DDA system worked as expected. In order words, the game gave the player a challenge that understands their skill level. The DDA system seems to have fulfilled its purpose.

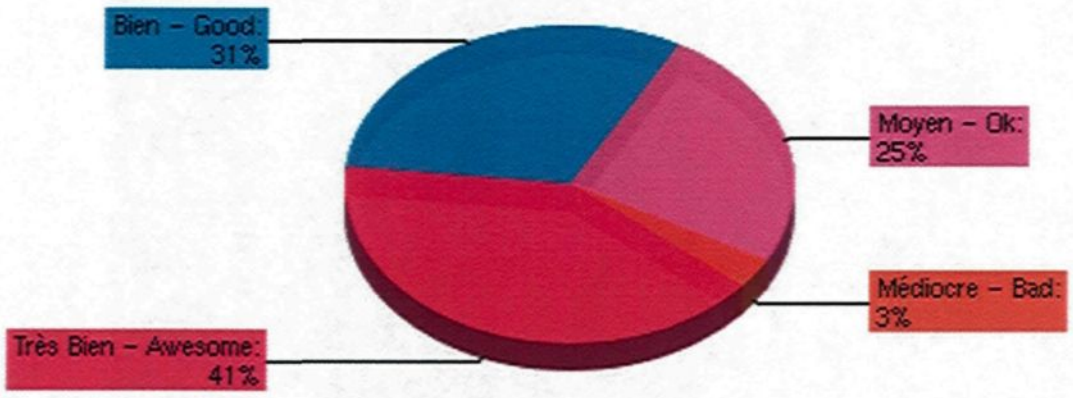

Figure 0.6 : What is your general appreciation of the game difficulty?

As discussed the difficulty seen that have been not too hard or too easy. This question asked the player to qualify the quality of the difficulty. In Figure 0.6, we can see at least 72% (23 out of 32) of the tester thought the game difficulty was at least good. That means the players thought the game was interesting in the point of view of the difficulty.

The player are seen the difficulty as a whole that move through play and how the player is challenged. This is definitely a direct result of our DDA model implemented that could change the difficulty according to the players' expectation and experience.

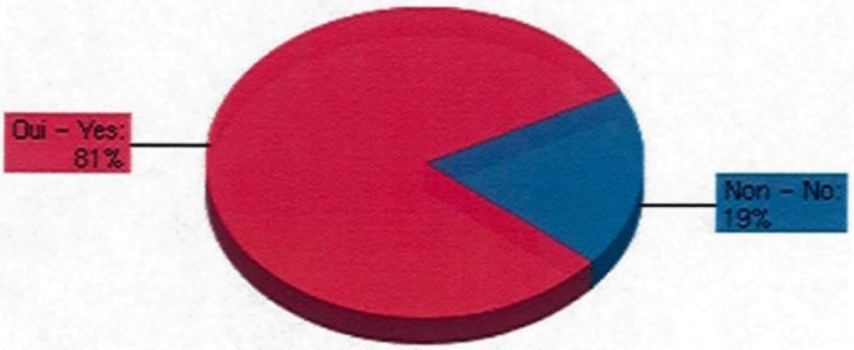

Figure 0.7 : Did you notice a game adaptation of the difficulty?

Although the players discovered a sort of pattern in the difficulty and 81% (26 out of 32) were aware of the presence of DDA, see Figure 0.7. This could be explained by the fact that this research is well known on the campus and the blog we created, where the players could test the game, which had entries related to DDA. Also, the testers could have melt down the difficulty of the AI and the math questions all together, where the arithmetic questions have a clear adaptation to it and the AI adaptation happens in background.

The Figure 0.8 is pretty clear about the idea that a should adapt itself to the player's skill level as 94% (30 out of 32) says. This result is important, it means the players in general are expecting the game to be harder when it is easy and easier when it is too hard.

This helps motivate the creation of more DDA and design model to enhance the player's game experience.

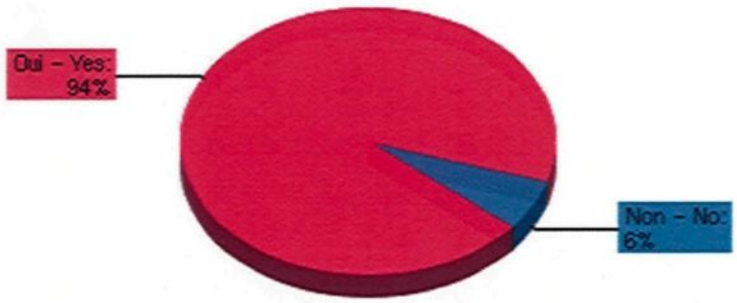

Figure 0.8 : Do you think it is important that the difficulty adapts itself to the player's skills level?

These results are very encouraging. This testing phase clearly suggests that the DDA system enhances the players' experience in the game NtN. Nevertheless, it should be pointed out that the results of a DDA system, namely the fun factor and the human experience in the game, are not trivial to evaluate and impossible to quantify. It is why we have chosen an experimental approach based on a qualitative evaluation of the players' experience.

1.16 PLAYER'S EXPECTATIONS ABOUT DDA

The video games are made on the idea of challenge. Arcade games were always about the player getting as they can, mastering the game mechanics and going until they ran out of money. Recently games are getting more complexes and more challenging than ever. DDA is supposed to constantly give player a constant fun and challenging game

experience. DDA plays with the difficulty of the game based on different formulas based on the player's skills. It can also be applied to different rewards and items the player can get over time.

One assumption about DDA system is that the player does not want to know it exist (Hunicke 2005). In general, they do not want a DDA system that will change the difficulty of the game during game play, they want it to be working not during play time, between levels or combat for example. If they realize the game is changing while they are playing it is most likely to break their magic circle with the game and they will stop playing as they will feel the game is cheating on them. They could also use it at their advantage and kill their experience. As (Koster 2004) pointed out, if a game has nothing else to teach or challenge the player, they should stop playing cause the relationship is now unhealthy. The idea is to make the DDA system as transparent as possible. For example, in the game *Need for Speed: Underground 2* the AI drivers could come back from a huge crash, where they are stop for several seconds, right in the player's tail after sometime, meanwhile the player has been driving full speed.

A DDA system should also respect the world the player is evolving. Imagine a RPG game in where the player puppies, because the DDA is not aware of the logic of the world, it could make the puppy super strong and it would not make any sense for the player to fight against strong puppy. For example, in the game *Oblivion*, city guards are always 2-5 levels above the player and bandits 2-5 levels below the player. After a certain moment of

playing the game, certain units would disappear from the world and in order for the bandits to be as strong as the player is, they would wear end-game amours and weapons. Which does not make sense, they are attacking the player for some gold when they could make a lot more money with their equipments. Non sense DDA experiment could lead to non logical world and destroy the player experience.

Although player is looking for good challenge, they do not want to be stuck in a design impasse because of DDA. In other word, if the DDA system decides to add more challenge for the player at one point and become impossible for the player to process a part of the game, they are going to leave the game and never come back. For example, In *Oblivion*, the strategy guide tells the player to finish certain quest before a certain level, because some enemies get way too strong and are impossible to vanquish.

Figure 0.9 : God Hand

It is possible to use a visible DDA system, but the player has to be well aware and understands its functioning. That DDA system has to be include in the GUI and needs to give clear feedbacks to the player about its status. A good example is the game *God Hand*, it uses the DDA system as a whole game mechanics and strategy for the game. In Figure 0.9, the skull in the bottom left hand corner is a difficulty meter. As the player vanquishes enemies, the crescent shape fills up, and when it reaches the top, the level of difficulty increases and the meter is emptied. As enemy pound the player, the meter decreases, as does the difficulty level. The game then rewards the player both implicitly; by giving the chance to player to succeed over impossible odds and explicitly; by awarding the player extra points which is converted to cash at the end of each level.

In general it is important that DDA respects the player's expectation about the challenge they want to experience. The player never has to know about the DDA system and how it functions. If so the designer has to make sure the DDA system is well explained and understood by the player. The DDA system also needs to respect the game world into the player is evolving and they have to make sure the DDA system does lead the player to a dead-end game.

1.17 LIMITATIONS OF THE PROTOTYPE AND THE MODEL

During the survey, the last questions asked the testers to tell the researchers what they are most likely to change in the game. These answers revealed some design limitations in the game. For instance, 60% (17 out of 32) of the players were expecting better art assets

in the game, even though a lot more time was spent on the graphics than on the first prototype [8]. Getting a real artist to do the art assets would be a way to solve this problem. The combat ergonomic did not please all players, 41% (12 out of 32) think it could be better, as some players pointed out, during a combat the player is only looking at one part of the screen; the arithmetic question. The players had the tendency to forget to look at the enemy and/or their own health bar. A way to solve that particular issue would be to put together the question, avatar, enemy and possible answer in the same location.

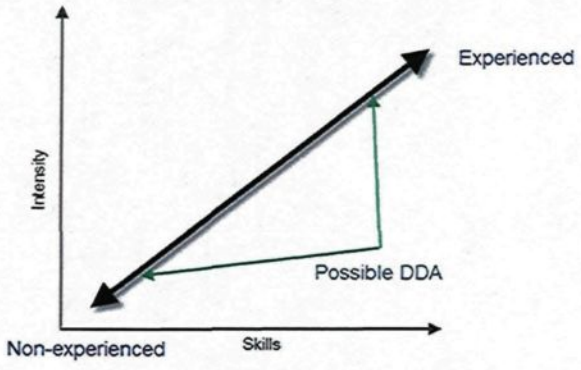

Figure 0.10 : DDA limitations

As we can see in Figure 0.5, approximately 80% of players found the difficulty adequate to their skill level. What about the 20% that found the game too difficult or too easy? Figure 0.10 expresses that idea, when a very experienced player plays the game, the DDA system is not fast enough to keep pace with the player. The same phenomenon happens for non-experienced players, the game evolves too fast for them, while they might not fully understand the game. The game should give them help, a support to their frustration of not being able to accomplish the task to be accomplished in the game. In

figure 8, we can see that the DDA system works great for average players, which includes everyone one that falls within the two arrows. As pointed out by a tester, at some point in the game, where the combat might be too difficult, in order to stay alive, the only possible action for the player is to heal themselves. The game should provide a support for the players so they could keep enjoying the game and fighting. In general, according to the comments and the survey, the game still stands out for its adaptive properties and fun.

CHAPTER 5

CONCLUSION AND FUTURE WORK

The game industry is rapidly growing and trying to reach new markets, this expansion brought lots of inexperience players to the video games world. These players are called casual gamers and their expectations about video games are different then experienced players. Most of casual players are looking for a moderate challenge while experienced players want a difficult or really difficult challenge. Based on that assumption, it is essential to understand and master the process of making video games that every type of player would like to play.

This new reality brings new challenges; game has to provide a certain difficulty level that would suit different kind of player such as experienced and inexperienced. It can be achieve by a propose challenges that matches the players' abilities (Tremblay, Bouchard et al. 2010), in order to keep the player a state of flow (Chen 2006). Most games currently on the market do not provide any system to dynamically adjust the difficulty level to their players' abilities (Hunicke 2005).

Researchers addressed this problematic (Gilleade and Dix 2004; Hunicke and Chapman 2004; Sweetser and Wyeth 2005; Chen 2006). They developed systems that use computational and algorithm approaches in order to adapt the system to the player's skills level. They also used purely designed approach that imbedded the difficulty adjustment in the game mechanics. Nevertheless they showed some limitations, the design approach can be impossible to add to some existing games, some models have never been implemented

and/or can only change the difficulty to be easier. As we discovered limitations, we decided to design and implement our own DDA model using a simple and inexpensive process.

The first goal of this research was to design a DDA system that is flexible and can adapt in real time the difficulty to the player's expectation. In this research, we addressed this issue by proposing a new adaptive model for dynamically adjusting, in real-time, the difficulty level of a game according to the players' performance. The foundation of this model relies on a computational approach for Dynamic Difficulty Adjustment (Hunicke and Chapman 2004) and it can be seen as an extension of our previous model proposed in (Tremblay, Bouchard et al. 2010). Our model is flexible, can be adapted to multiple game genres, and offers two different layers of reasoning, short term adjustment (*tactical*) and long term adjustment (*strategic*). The *tactical layer* is used to understand small actions/moves and to react quickly to punctual problems. The *strategic layer* exploits the theory behind the well-known ranking system ELO (Coulom 2010), which allows the system to estimate the players' performance in the long run and to adjust the game according to their performance.

Our second objective was to enhance the player's experience by introducing DDA in their gaming experience. To achieve this objective, the model has been implemented for validation in the form of simple calculation/combat game called Number to Number. An experiment has been conducted using this prototype where 150 playing sessions have been completed by 32 players. Each player had to answer a detailed questionnaire on their

playing experience. In general, our observation show tendencies that the DDA model introduces and the design of the game enhance the player's experience. The results were very promising, showing the value of the proposed approach and giving us indications for improving the model.

Parts of this new DDA model have been published in the proceedings of the International Conference on Computer Supported Education (CEDU): session "Gaming platforms for education and reeducation" in Valencia, Spain in 2010. This paper was well received by the community and this recognition constitutes a clear sign of the pertinence of the solution proposed in the thesis. An extended version of this previous paper is under review for publication in the Journal of Computer Entrancement, Elsevier.

1.18 FURTHER WORK

Based on these positive results, we are planning to work on the implementation of our adaptive approach into other kinds of video games, such as puzzle, action, strategy, etc. It will be interesting to test the volatility of our approach. Also, in order to better understand the real impact of the DDA model, it will be interesting to build two versions of the testing prototype; one with DDA and another without DDA. In this way, we will be able to compare the results obtained with both versions of the game and to target the exact portion of questions where the DDA produces a significant difference in the rating. Moreover, the questions of learning from game mechanics could be enhanced using a DDA system as the player is experiencing fun for a longer period of time.

In a larger perspective, we think that our work will, over time, help to see more and more adaptive models in video games. It should also be noted that this work is a part of a wider project conducted by the LIARA laboratory of the University of Quebec at Chicoutimi. This project consists of implementing a larger model that includes support and DDA to serious game players such as Alzheimer patients. With this model, we will be able to simulate the cognitive functions of the patients. With a DDA model, while playing, the patients could be experiencing flow for longer, therefore be stimulate and learn from the game for a longer period. In order to propose adequate challenges for them, and to give assistance inside the game when they are in need. In the end, it will help slowing the degenerative process of the disease. A way to enhance the quality of this model would be use fuzzy logic to determine the level of difficulty for each player. Based on the fuzzy logic we could give the player a game that is "harder" but still adapt to the player, rather a game that has a level of difficulty of 5. We can expect DDA to be a big part of serious gaming as well, as doctor are learning procedures and the military is learning tactics from video games, DDA systems will enhance their learning and gaming experience (Liao 2000; Lin, Lin et al. 2009). Hopefully in the future we can look forward to seeing different generations being able to play the same games and experience an equal level of amusement by introducing DDA, as the game will adapt to the players and not the other way around.

1.19 PERSONAL ASSESSMENT ON THIS RESEARCH

Finally, I will conclude with a brief personal assessment of my initiation to research. The journey made throughout this project was very rewarding, both in acquiring new knowledge in a targeted area of expertise (difficulty adjustment in video games) and in term of development of new skills (research methodology, implementation, communication skills, etc.). This rewarding experience also allowed me to make a modest contribution to the scientific community in my field of research. In this sense I would love to continue in the near future my experience in graduate and begin a doctoral studies in relation to my field of expertise.

REFERENCES

Bleiweiss, A., D. Eshar, et al. (2010). Enhanced interactive gaming by blending full-body tracking and gesture animation. ACM SIGGRAPH ASIA 2010 Sketches. Seoul, Republic of Korea, ACM: 1-2.

Bolstad, W. M. (2007). Introduction to Bayesian Statistics, Wiley-Interscience.

Bouchard, B., A. Bouzouane, et al. (2007). "A Keyhole Plan Recognition Model for Alzheimer's Patients: First Results." Journal of Applied Artificial Intelligence (AAI) **22**: 623-658.

Campbell, J. (1982). Grammatical Man: Information, Entropy, Language, and Life, New York: Simon & Schuster.

Chen, J. (2006). Flow in games. School of Cinematic Arts Los Angeles, USA, University of Southern California. **MFA in Interactive Media**.

Claypool, M. (2005). On the 802.11 turbulence of nintendo DS and sony PSP hand-held network games. Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games. Hawthorne, NY, ACM: 1-9.

Coulom, R. (2010). "Le problème des classements." Pour la Science(293): 20.

Crawford, C. (2003). Chris Crawford on Game Design, New Riders Publishing.

Csikszentmihalyi, M. (1990). Flow: the Psychology of Optimal Experience, Harper Perennial.

Dormann, C. and R. Biddle (2008). Understanding game design for affective learning. Proceedings of the 2008 Conference on Future Play: Research, Play, Share. Toronto, Ontario, Canada, ACM: 41-48.

Gardner, M. (1970). "Mathematical Games : The Fantastic Combinations of John Conway's New Solitaire Game "Life" " Scientific America **223**: 120-123.

Gilleade, K. M. and A. Dix (2004). Using frustration in the design of adaptive videogames. Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology. Singapore, ACM: 228-232.

Grissom, S. (2008). "iPhone application development across the curriculum." J. Comput. Small Coll. **24**(1): 40-46.

Ho, S.-H. and C.-H. Huang (2009). "Exploring success factors of video game communities in hierarchical linear modeling: The perspectives of members and leaders." Computers in Human Behavior **25**(3): 761-769.

Hunicke, R. (2005). The case for dynamic difficulty adjustment in games. Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology. Valencia, Spain, ACM: 429-433.

Hunicke, R. and V. Chapman (2004). "AI for Dynamic Difficulty Adjustment in Games."

Johnson, D. M. and J. Wiles (2003). "Effective affective user interface design in games." Ergonomics.

Juul, J. (2009). Fear of Failing? The Many Meanings of Difficulty in Video Games. The Video Game Theory Reader 2. M. J. P. W. B. Perron. New York, Routledge 237-252.

Koster, R. (2004). A Theory of Fun for Game Design, Paraglyph Press.

Lazzaro, N. and K. Keeker (2004). What's my method?: a game show on games. CHI '04 extended abstracts on Human factors in computing systems. Vienna, Austria, ACM: 1093-1094.

LeBlanc, M. (2000). Formal Design Tools: Emergent Complexity, Emergent Narrative. Game Developers Conference.

Liao, S.-h. (2000). "Case-based decision support system: Architecture for simulating military command and control." European Journal of Operational Research **123**(3): 558-567.

Lin, C., C. M. Lin, et al. (2009). "A decision support system for improving doctors' prescribing behavior." Expert Syst. Appl. **36**(4): 7975-7984.

Lindley, C. A., L. Nacke, et al. (2008). "Investigating the Cognitive and Emotional Motivations and Effects of Computer Gameplay." CGAMES.

Lindley, C. A. and C. C. Sennersten (2008). "Game play schemas: from player analysis to adaptive game mechanics." Int. J. Comput. Games Technol.: 1-7.

Magerko, B., C. Heeter, et al. (2008). Intelligent adaptation of digital game-based learning. Proceedings of the 2008 Conference on Future Play: Research, Play, Share. Toronto, Ontario, Canada, ACM: 200-203.

Mandler, J. M. (1984). Stories, Scripts and Scenes: Aspects of Schema Theory Hillsdale, NJ, Lawrence Erlbaum Associates.

Mateas, M. (2002). Interactive drama, art and artificial intelligence, Carnegie Mellon University: 273.

Meier, S. "Civilization." from http://www.civilization.com/.

Nacke, L. and C. A. Lindley (2008). Flow and immersion in first-person shooters: measuring the player's gameplay experience. Proceedings of the 2008 Conference on Future Play: Research, Play, Share. Toronto, Ontario, Canada, ACM: 81-88.

Natapov, D., S. J. Castellucci, et al. (2009). ISO 9241-9 evaluation of video game controllers. Proceedings of Graphics Interface 2009. Kelowna, British Columbia, Canada, Canadian Information Processing Society: 223-230.

Norton, M. and D. Avery (1999). Baldur's Gate: Tales of the Sword Coast Official Strategies, SYBEX Inc.

Rolling, A. and E. Adams (2003). Adrew Rollings and Ernest Adams on Game Design, New Riders.

Salen, K. and E. Zimmerman (2003). Rules of Play: Game Design Fundamentals, The MIT Press.

Schell, J. (2008). The Art of Game Design - A Book of Lenses, Morgan Kaufman

Seitz, L. K. (1997). "Atari Xevious." from http://home.hiwaay.net/~lkseitz/cvg/vvgl/XEVIOUS.html.

Sengers, P. (1998). Anti-Boxology: Agent Design in Cultural Context. Pittsburg, Carnigie Melon University. **PhD Thesis**.

Siwer, S. E. (2010). Video Games in the 21st Century, entertainment software association.

Srinivasan, V., K. Butler-Purry, et al. (2008). Using video games to enhance learning in digital systems. Proceedings of the 2008 Conference on Future Play: Research, Play, Share. Toronto, Ontario, Canada, ACM: 196-199.

Sweetser, P. and P. Wyeth (2005). "GameFlow: A Model for Evaluating Player Enjoyment in Games." Comput. Entertain. **3**(3): 3-3.

Sykes, J. and S. Brown (2003). Affective gaming: measuring emotion through the gamepad. CHI '03 extended abstracts on Human factors in computing systems. Ft. Lauderdale, Florida, USA, ACM: 732-733.

Tremblay, J., B. Bouchard, et al. (2010). Adaptive game mechanics for learning purposes: making serious games playable and fun. Proceedings of the Int. Conf. on Computer Supported Education : session "Gaming platforms for education and reeducation", Valencia, Spain.

Walz, S. P. (2010). Toward a ludic architecture: the space of play and games, ETC Press