User-centered Design in Games

Randy J. Pagulayan[1], Kevin Keeker[2], Thomas Fuller[1], Dennis Wixon[3], Ramon L. Romero[1], and Daniel V. Gunn[1]

Microsoft Game Studios[1]

Zynga[2]

Microsoft Surface[3]

Introduction

The application of User-Centered Design (UCD) methodologies has had a tremendous impact on the aviation, medical, and technology industries over the past several decades (Preece et al., 1994; Salas & Maurino, 2010; Wickens & Hollands, 1999). User-centered design principles and evaluation methods have successfully made our interaction with tools faster, safer, and more efficient. Yet, there has been growing interest in the adaptation of these techniques to make products more pleasurable (Jordan, 2000). Instead of focusing on making tools faster, safer, and more efficient, greater emphasis has been placed on making them enjoyable to use (Norman, 2005). For more than a decade at Microsoft, we have been applying, refining, and inventing new UCD techniques to improve not only the *usability* of our games, but, more importantly, the *enjoyability*. The purpose of this chapter is to review some principles and challenges in applying UCD techniques to game improvement and evaluation. First, we discuss why games are important followed by a differentiation between games and productivity software. Next, we discuss the principles and challenges that are unique to the design of games. That discussion provides a framework to illustrate the core variables we think one should measure to aid in game design testing and evaluation. Finally, the chapter will conclude with some examples of how those variables have been operationalized in methods used by the Games User Research group at Microsoft Game Studios.

Why Games Are Important

In the 1950s, computers imposed a technological and economical barrier to entry that was not easy to overcome for the general population. Only scientists, engineers, and highly technical persons with access to universities or large companies were able to use these machines (Preece et al., 1994). As computers became less expensive, more advanced, and more reliable, the technology that was once available to only a small subset of the population began proliferating throughout society and became integral to everyday life. In order to make this transition possible, well designed interfaces between the user and the technology were required. Video games come from similar origins and appear to be heading down a similar path. Some early attempts at making commercial video games failed due to unnecessarily high levels of complexity. As Nolan Bushnell (co-founder of Atari) states, “No one wants to read an encyclopedia to play a game” (Kent, 2000, p.28). In retrospect, some of the most successful video games were the ones that were indeed very simple. The more recent adoption of “broadening interfaces” such as Nintendo’s Wiimote, Microsoft’s Project Natal, or Sony’s Move, as well as the large number of people buying games, shows a similar proliferation of games in society. This too is only possible with well designed interfaces between the user and technology.

Games Industry Generates Enormous Revenue

Within the first twenty-four hours of its release, the video game Halo 3 generated approximately $170 million dollars in sales, in the U.S. alone, rivaling the film industry’s opening day record for Spiderman 3 as well as the book industry’s opening day record for Harry Potter and the Deathly Hallows (Chalk, 2007; Microsoft, 2007). Video games are one of the fastest growing forms of entertainment to date, and today the video games industry generates enormous revenue. According to the Entertainment Software Association (ESA), revenue from computer and console games has grown from 2.6 billion dollars in 1996 to 11.7 billion dollars in the year 2008 (ESA, 2009). Approximately 298 million games were sold in 2008 and approximately 43% of Americans purchased or planned to purchase one or more games in 2009 (ESA, 2009). These statistics do not even consider the international importance of video games. To put this into perspective, the combined US/Canada movie box office for 2008 was $9.6 billion (MPAA, 2009) which is approximately $2 billion less than the revenue from computer and console games from the US alone.

Games Aren't Just for "Gamers" Anymore

The revenue created by the games industry, one that rivals the US film industry, suggests that gaming appeals to more than just a niche market. In fact, video games are no longer associated with the stereotypical image of a gamer as an adolescent male; women aged 18 and older represent a larger portion of the game playing population (34%) compared to boys aged 17 and younger (18%), and the average age of those who play video games is 35 (ESA, 2009). In addition, gaming has become increasingly more popular with the older population with 25% of all gamers being over the age of 50 (ESA, 2009). All together, a majority of American households play computer or video games and with game revenue rivaling the motion picture revenue, it is no wonder that the people who play games are as diverse as the games they play.

Games Push Technology and Research

In addition to driving revenue, video games have also helped to shape and drive technology and research. For example, during the early 1980s, IBM commissioned Sierra Online to develop a game to show off the advanced graphics capabilities and three channel sound of their upcoming IBM PCjr computers. While the PCjrs didn't fare well in the marketplace, Sierra's creation actually might have helped IBM's competition. The resulting game, King's Quest, arguably helped drive sales of the PCjr compatible Tandy 1000 computer by showcasing its enhanced graphics and sound capabilities. A more recent example of games driving technology can be seen in the Xbox 360 Xenon processor developed by IBM and the Sony cell processor developed by Sony, Toshiba, and IBM. These processors were purposefully built for the Xbox 360 and Playstation 3, required several years of design, research, and iteration in order to build, and pushed the then current microprocessor technology to its limits – all for the primary goal of providing these consoles with the computing power required to support next generation games (Shippy & Phipps, 2009).

Games are a Source of Interface Innovation

Productivity applications typically strive for interface consistency from one version to the next. In that market segment, learning a new interface is considered a cost by business and a burden by users. Changing interfaces should only be done if the gains in efficiency outweigh the costs. Indeed, even the input devices associated with productivity applications have been relatively consistent for a long time. Yet, while the mouse and keyboard have stayed essentially the same for 30 years, game controllers evolve and many new games use new and novel input devices (such as guitars, game mats, NES power gloves, Natal). In fact, gamers demand novelty, and, consequently, these evolving and novel devices create game design and ergonomic challenges that are very important to study. Moreover, each interface innovation developed for a game can potentially touch millions of people. This creates a chance to extend the interface innovation out of the games market and into other venues. For this reason, games can and should be viewed as an experimental interface testing ground.

Games vs. Productivity Applications

As the games industry matures, games become more complex and varied making them difficult to narrowly define. Articulating a clear and succinct set of principles that capture the essence of games is not straightforward. Yet, shedding some light on the difference between games and productivity applications might make it easier to see how traditional UCD techniques must be adapted in order to have the most impact on games. When comparing games to productivity software, there are principles and methods that can be successfully applied to both. Regardless of the domain, the same techniques would be applied to understand a misleading button label or a confusing process model. At the same time, there are fundamental differences between productivity applications and gaming applications. Some of the differences can be fairly clear; games are usually more colorful, wacky and escapist than productivity applications, with the inclusion of interesting story lines, or animated characters. Others may not be. In productivity applications, unclear labeling in the user interface (UI) is affecting my *productivity*; in games, the unclear labeling in the UI is affecting my *fun*.

User-centered design principles have not reached game makers to the degree that they have influenced other electronic applications. The following sections will describe some differences that have important implications for user research on games.

The Goals are Different

At a very basic level, the design goals, as well as user goals, are often quite different between games and productivity applications. If approached from a tools perspective, productivity applications are tools that help consumers be productive whereas games help consumers have fun. This perspective is useful because it allows us: 1) to prioritize usability and enjoyability issues based on their impact on fun; 2) to see why the role of designer intent is so important in usability/enjoyability testing of games; and 3) to focus on the similarities of UCD principles between tool and game testing to more clearly see how traditional UCD techniques can be adapted to game development.

Because productivity applications are focused on making the user productive, the focus of design and usability is to allow the user to produce an improved work product or result with less effort, higher quality, and greater satisfaction. However, games are intended to be a pleasure to play. Ultimately, games are like movies, literature, and other forms of entertainment. They exist in order to stimulate thinking and feeling. Their outcomes are more experiential than tangible. This is not to say that word processors or other tools cannot be a pleasure to use, or that people do not think or feel when using them, but pleasure or other feelings are rarely their primary design intention. In a good game, both the outcomes and the journeys are rewarding. This fundamental difference leads us to devote more of our effort to measuring perceptions while productivity applications focus more on task completion.

The Role of Designer Intent

The general design goal for productivity applications is to *enable users to be productive*; for games, the goal is to *enable users to have fun.* For testing purposes, it is generally easier for usability practitioners to spot user deviations away from designer intent within productivity applications than it is in games. Most people can easily see when productivity is blocked; determining if, and when, fun is blocked is a bit more elusive. For example, it is unlikely that the designer intends for users to take four attempts to print their work so that it does not appeared cropped in a productivity application. If the blockers to more efficient performance could be identified and removed, and users could successfully print on average with just one attempt, the redesigned application would make the users more productive and map better onto the designer intent. However, in games, four attempts at beating a boss might map perfectly onto the designer's intent. In fact, defeating a boss with only one attempt might actually be a blocker to fun if is perceived as too easy. Six attempts might make the game too hard, but four attempts might be "just right." Moreover, four attempts might be just right for *some* users and not others. Thus, for games, we must rely more heavily on mapping user behavior against explicit designer intent. The designer holds the vision for their creation as well as the vision for what makes their game fun. User Research on games requires close collaboration with Design in order to spot deviations away from intent that could impact fun.

Games Must Challenge

Perhaps the most important distinction between games and productivity applications is the complex relationship between challenge, consistency, and frustration. Productivity applications, and usability testing efforts associated with them, strive to remove blockers to productivity – challenging flows, difficult navigation, unclear labeling, etc. With games, we strive to remove blockers to fun – cumbersome weapon changing, uncomfortable controller mappings, confusing gameplay, etc. Yet, certain kinds of gameplay challenges and difficulty don't block fun but are intentionally placed there by the designer to enhance it; they add to the *enjoyability*. To illustrate this difference consider the following example. It has been said that the easiest game to use would consist of one button labeled "Push." When you push it, the display says "YOU WIN." This game would have few, if any usability issues; but it wouldn't be fun either. Indeed, it has enjoyability issues galore: lack of challenge, lack of strategy required, lack of gameplay tradeoffs, lack of interesting goals, and lack of replayability (to name just a few) all conspire to make this a pretty stale game. With user research in games, testing efforts should focus more on mitigating those factors that most impact enjoyability and impede fun. One should ensure that challenge adds to enjoyability and doesn't detract from it and that it is part of an intentional gameplay element and not a usability issue.

Industry Competition is Intense

Competition within the games industry is more intense than other software domains. Games compete with each other as well as many other forms of entertainment for your attention. Pretend that you have to write a book chapter. For that task, there are a limited number of viable tools: hand writing, dictation, a typewriter, or a limited set of software applications. But, imagine that you've got a few hours to kill and you want to avoid productivity at all costs. You could read the biography of Nikola Tesla, argue politics with your best friend, watch a mystery film, or race your Audi R8 160 mph down Nürburgring in Forza Motorsport 3 (Microsoft, 2009). This element of choice, combined with the large number of games, means that many games will fail (just as many films, and TV shows fail). In contrast, a productivity application that is the "standard" will enjoy massive and sustained success.

Many of the characteristics mentioned above are not unique to games. But all of these characteristics are relevant to the discussion of the ways in which user research must adapt to be useful for game development. Later, we will discuss how these differences create particular challenges for both the design and evaluation of games.

Types of Games

Below we review some common gaming platforms and game types.

Platforms

One of the simplest classifications of games is by the platform or hardware that they are played on. Cassell & Jenkins (2000) differentiates games played on a PC from those played on a console. Crawford (1982) divides games into even finer categories: arcade/coin-op, console, PC, mainframe, and handheld. Different gaming platforms can be differentiated by their technical capabilities, physical control interface, visual interface, and the context in which they are played.

PC. A useful distinction can be made between PC games that are normally acquired through retail outlets on a CD/DVD, persistent world games in which much of the content "lives" on the Internet, and casual, web-based games. While there are many technical differences between these kinds of games, the important distinction is the business model and level of investment required by the players. Most retail games need to rely heavily on flash, reputation, and recognizable novelty to attract users to a relatively large investment. Continued investment in the game is only useful to build a reputation or to convince customers with higher thresholds that they will get their money's worth from the game. But massively multiplayer online persistent worlds (MMO) require continued user investment to obtain monthly subscription fees. The long learning curves and reinforcement schedules in (MMO) are tailored to make players invest more time, money, and effort for long-term rewards. Many casual gamers are attracted to free, familiar games. For these casual users, game play is squeezed in between activities that are more important to them. As a result, games with very little learning investment are very appealing to casual gamers. Removing penalties for setting aside the game (for minutes, hours, days or forever) can mean the difference between successful and unsuccessful casual games. See Casual Games section for more details.

Console. For user research purposes, the most important unique characteristic of console video game systems is the advantage of a fixed set of hardware. By contrast with PC games, there is very little game setup, minimal maintenance efforts (e.g., few software patches, video card conflicts, and so on), and a consistent set of input device capabilities. Some game genres are more popular on console than PC. This difference in popularity can usually be attributed to differences in the input devices that are typically associated with each platform. It's hard to enter text using a controller. Thus, most popular MMO's are still published for PC's in spite of the fact that fast internet connections and hard drives are becoming more popular additions to console systems. This may change as it becomes easier to attach other devices such as keyboards or voice command hardware to consoles which would make communication more robust.

More recently, casual games are also available for all console systems, including Xbox 360, where consumers can download, try, and purchase casual games directly from the console via a broadband connection.

Portable Devices. First released in 1989, the Game Boy and Game Boy Color (1998) combined to sell over 118 million units, and this success has only accelerated with the Gameboy Advance (2001; 80 million+ units) and Nintendo DS (2004;125 million+ units) (Business Week, 2006; Nintendo, 2005). Nintendo succeeded largely by catering to the massive mainstream demand for accessible games; starting in 1989 by selling a whopping 35 million units of Tetris and 14 million units of Super Mario Land, and later selling more than 34 million Pokemon-related cartridges (Nintendo, 2005). In fact, the portable gaming market is littered with devices that have tried to beat Nintendo with higher-tech devices and more action-oriented games. More recent trends point to an impending explosion in mobile phone gaming. Since 2005, Tetris, now 26 years old, has sold to more than 100 million cell phones (Maximejohnson, 2010). As mobile phones becomes more accessible and improve the visual display, gaming upon them becomes more and more popular.

Consumers primarily view mobile phones as communication devices, which are not meant to replace the gaming experience available on handhelds such as the Nintendo DS and Sony PSP (Moore & Rutter, 2004). Unlike handheld gaming devices, mobile phones are generally carried at all times. This creates an opportunity to provide gaming at any time, anywhere (Hyman, 2005; IDGA, 2005). However, with mobility comes the greater need to design for time, location, social, emotional, and motivational context (Sidel & Mayhew, 2003).

Game Types

Retail Games (NPD Classifications). The NPD Group (a marketing company) uses a fine grained classification scheme for game type which is referred to quite often in the games industry. They offer the following classes and subclasses as seen in Table 1. Games can also be categorized in a more granular fashion by splitting each genre up into a subgenre. For instance, action games can be divided into Action Driving Hybrid (Crackdown 2, Saints Row 2, etc.), Action combat (Tom Clancy's Splinter Cell: Conviction, Tomb Raider: Underworld, etc.), and Platformer (Ratchet & Clank Future: A Crack in Time, Super Mario Bros Wii, etc.). In general, the genres represent types of gameplay mechanics, themes, or content.

Insert Table 1 about here

Casual Games. Casual games is a rapidly growing segment of the video games market, producing an estimated $500 million in online sales in 2005 (IDG, 2005). Casual games as a whole are appealing to a broad range of users, can be found on all platforms, and represent several genres. For these reasons, they do not fit into either of the previous categorization schema. Casual Games generally meet the following requirements (1) are easy to start and control; (2) can be enjoyed in small time intervals; (3) don't require an extensive investment in time to enjoy or progress, and therefore usually don't have deep linear storylines; and (4) have discreet goals and rules that are intuitive or easy to learn. Casual games often have a significantly lower development cost, can be distributed digitally (or via physical media), offer trial versions and lower retail prices (or are free), and utilize lower system requirements and storage space. These requirements and attributes make these games well suited for platforms such as PC distributed via the web and mobile devices, where they are most common.

Websites such as Real Arcade, Pogo.com, Yahoo! Games, and MSN Games provide free online casual games, pay per play casual games, subscription services, downloadable games that include a free trial. The downloaded games can be purchased for $5-$20. Mobile phone carriers and a variety of websites provide casual games for mobile devices. Although there are feature rich, 3D non-casual games available for mobile devices, the top selling titles are casual games such as Tetris, Bejeweled, Bowling, and Pac Man (Hyman, 2005).

Recent Trends in Gaming

Physical Gaming and New Input Paradigms. New input paradigms are an ongoing area of exploration and innovation in games that will continue into the future. The most recent innovation has been the advent of physically-based at-home video gaming as with the highly popular *Wii gaming system* (Nintendo, 2006). Additionally, in recent years there have been several highly successful games that require the user to simulate musical instrument play via rhythm-based gaming like in *Guitar Hero* (Activision, 2005) or *Rock Band* (Electronic Arts, 2007) using simulated guitar-controllers or, in some cases, to closely mimic actual musical play like when using drums in *Rock Band* or singing in *SingStar* (Sony Computer Entertainment, 2004). Historically these video game types date back to at least the 1990s. There were arcade games in the mid-to-late 90s requiring physical movement to play including *Alpine Racer* (Namco, 1995) and *Dance Dance Revolution* (Konami, 1998). Karaoke-style singing was popularized in Japan in the 1970s, though the idea of turning it into a game with a score system did arise with the aforementioned *SingStar*. Looking into the future it appears the trend towards at-home physically-based videogaming will only continue as the *Playstation Move* (Sony Computer Entertainment, 2010) and *Project Natal* (Microsoft, 2010) have each been recently announced and both systems rely on improved gesture and physically-based control-input systems.

New input paradigms create immense complications when considering user-centered game design because we must now account for differing talent levels of the individual players. In the game *SingStar*, to be successful (from the game design's perspective), a player must have some ability to hear and sing at various pitches in order to succeed in the game. In some cases, the game does mimic exactly the real world activity, but instead is based on an approximation of how to do something in the real world. This is the case with most *Rock Band* and *Guitar Hero* gameplay. In these cases, having actual experience and skills with a guitar may impede a player's ability to adapt to the modified guitar-controller interface. Finally, there are other factors to consider. Similar to the talent requirement for singing, in physically based gaming we must account for the fatigue level, flexibility, and coordination of potential players –even when considering a relatively simple game such as the Archery mini-game in *Wii Sports Resort* (Nintendo, 2008).

Social Network Gaming. Gaming on social network sites, such as Facebook, is the most popular new trend in internet gaming. FarmVille launched on Facebook in June 2009 and had bloomed to over 80 million active users as of April 2010 (Gardner, 2009; Facebook, 2010).

Earlier social successes, such as Second Life (2003), innovated on AOL's chat rooms by giving players a "sand box" like virtual world that they could mold and shape to fit their identities and their group connection needs. Inside these spaces, persistent clubs could form to bring people with common interests together across otherwise insurmountable distances. MMO's such as World of Warcraft built economies in their game world to necessitate the formation of friendships, allies and guilds out of strangers.

Social network games borrow from these two types of games and extend the social equation in a way that has successfully attracted a mainstream audience. Like World of Warcraft, social connections are required to succeed. Players must recruit friends as neighbors and connect with them daily through gift-giving in order to advance. This recruiting process serves a viral marketing channel to bring new players into the game. The extension comes primarily from reducing social barriers. Virtual worlds with strong in-group communities require intense time commitment and expert knowledge. Social networking games are free and can be played in 5 minutes a day, asynchronously, to suit busy, varied schedules, and time zones. They use socially normative themes such as farming, pets, restaurants, and cities as their inspiration, title, and language. Then they use familiar shared events such as holidays and seasons to introduce freshness and novelty on a daily basis. They provide social safety by using a trusted social network (Facebook) for authentication and identity and largely eliminate direct interaction. Virtual gifts are free to the giver and the receiver but still valuable for advancing in the game. Through these social lubricants, social networking games have attracted a massive, mainstream audience. The average player is 43 and female. This typical player might have logged thousands of hours on Solitaire and Minesweeper but wouldn't have called themselves gamers.

Cross platform. Cross platform convergence is an emerging trend in the games industry that complicates the classification of games by platform. Developers are now creating multiple versions of a game that interact with one another across platforms. For example, Nintendo games such as *Animal Crossing* (Nintendo of America, 2002), *The Legend of Zelda: The Wind Waker* (Nintendo of America, 2003), *Harvest Moon* (Nintendo of America, 2003) allow users to connect a Gameboy portable device to the GameCube console to access additional features and content, or use the Nintendo DS as a touch screen controller. Another example of cross platform games would be a title where the player may execute specific planning or strategic tasks on a mobile device at any time, then have those actions affect the full game experience the next time it is played on a console or PC. This cross-platform design effect has been referred to as a reciprocation effect (Yuen, 2005).

Cross platform convergence creates new challenges in user centered game design. The act of switching between different platforms means the game must be adapted to multiple interfaces. Users must be informed how and why they should take advantage of the alternate platform. Users must also be made aware of the alternate platform version's existence so they can choose to take advantage of it. For example, in 2010 Microsoft released a Facebook game called *Toy Soldiers: Match Defense* (Microsoft Game Studios, 2010) and shortly thereafter released the Xbox 360 game *Toy Soldiers* (Microsoft Game Studios, 2010). Several relationships between the games were established. Playing the Facebook version of the game unlocked or awarded certain features in the Xbox version of the game. Meanwhile, the Xbox version of the game included several hints or loading tips, pointing out that there were benefits to playing the Facebook version. Ensuring success at each point in the seams between the games and platforms represents a new and ongoing responsibility for user-centered game design now and into the future.

The next generation of cross platform experiences may involve matching players for real-time, or asynchronous, competitive and cooperate multiplayer experiences regardless of what platform they are on (portable, PC, console, etc). As these trends become realized, interesting challenges and opportunities for game developers will mount as games will have to be designed with the limitations of the platform in mind. Indeed, spanning platforms with a single game will not be as simple as porting it. Careful consideration will have to be made with regards to the unique user needs associated with the each platform they are interacting with.

Principles and Challenges of Game Design

Having differentiated games from other applications we can look at some of the unique issues in game design and evaluation.

Identifying the Right Kind of Challenges

Games are supposed to be challenging. This requires a clear understanding of the difference between good challenges and frustrating usability problems. Most productivity tools struggle to find a balance between providing a powerful enough tool set for the expert, and a gradual enough learning curve for the novice. However, no designer consciously chooses to make a productivity tool more difficult. After all, doing so would go against the main design goal of making users productive. The ideal tool enables users to experience challenge only in terms of expressing their own creativity. For games, learning the goals, strategies, and tactics to succeed is part of the fun.

Unfortunately, it is not always clear which tasks should be intuitive (i.e., easy to use) and which ones should be challenging. Input from users becomes necessary to distinguish good challenges from incomprehensible design. Take a driving game for example. It's not fun having difficulty making your car move forward or turn. But learning to drive is still a fundamental part of the challenge in the game. While all cars should use the same basic mechanisms, it may be fun to vary the ways that certain cars respond under certain circumstances. It should be challenging to identify the best car to use on an icy, oval track as opposed to a rally racing track in the middle of the desert. The challenge level in a game must gradually increase in order to maintain the interest of the player.

Addressing Different Skill Levels

Unfortunately, all players don't start from the same place in terms of gaming experience or talent. Obviously, frequent failure can be a turn off. Success that comes too easily can also become repetitive. Games must address the problem of meeting all players with the correct level of challenge. Tuning a game to the right challenge level is called "game balancing".

There are many ways to balance the difficulty of the game. The most obvious way is to let players choose the difficulty themselves. Many games offer the choice of an easy, medium or hard difficulty level. While this seems like a simple solution, it is not simple to identify exactly how easy the easiest level should be. Players want to win, but they do not want to be patronized. Too easy is boring and too hard is unfair. Either experience can make a person cease playing.

Another approach to varying skill levels is to require explicit instruction that helps all users become skilled in the game. You might imagine a tutorial in which a professional golfer starts by explaining how to hit the ball and ends by giving instruction on how to shoot out of a sand trap onto a difficult putting green. Instruction, however, need not be presented in a tutorial. It could be as direct as automatically selecting the appropriate golf club to use in a particular situation with no input from the user, similar to the notion of an adaptive interface, where the interface provides the “right information” at the “right time.”

The environments, characters, and objects in a game provide another possibility for self-regulation. Most games will offer the player some choices regarding their identity, their opponents, and their environment. The better games will provide a variety of choices that allow users to regulate the difficulty of their first experiences. With learning in mind, it is not uncommon for the novice player to choose a champion football team to play against a weak opponent. As long as the player can distinguish the good teams from the bad ones, and the teams are balanced appropriately, users will be able to manage their own challenge level.

Some games take it even further by identifying the skill level of the player and regulating the instruction-level appropriately. In this situation, instruction can be tuned to the skill level of the player by associating it with key behavioral indicators that signifies that the player is having difficulty. If the game does not detect a problem, it does not have to waste the player’s time with those instructions. In *Halo 2* (Microsoft Game Studios, 2004) the game detects difficulties that a player may have with certain tasks. For example, to get into a vehicle, the player must press and hold the ‘X’ button on their controller when standing next to the driver’s seat of that vehicle. If the player is standing in the right position, but taps the ‘X’ button repeatedly (instead of holding the button down), the game will present a more explicit instruction to press and *hold* the button. This is just one of many dynamic instructions that appear throughout Halo 2 based on behavioral indicators of player difficulty.

Productivity tools have implemented similar problem-identification features, but often with mixed success due to the open nature of tasks in most productivity applications. Good game tutorials have succeeded by setting clear goals and completely constraining the environment. Doing so focuses the user on the specific skill and simplifies the detection of problem behavior. Other lessons from game tutorial design will be described in later sections of this chapter.

Another in-game approach to auto-regulating the difficulty level requires adjusting the actual challenge level of the opponents during the game. Evaluating the success of the player and adjusting the opponent difficulty during the game is often called “dynamic difficulty adjustment”, or “rubber-banding.” When players perform very skillfully, their performance is moderated by computer-generated bad luck and enhanced opponent attributes. In a football game, the likelihood of fumbling, throwing an interception, or being sacked, may increase as the player increases their lead over their opponent. Even though this may seem like a good solution, there can be a downside. Most people would prefer to play a competitive game (and win) than to constantly trounce a less skilled opponent. However, over-developed rubber-banding can cheat a skilled player out of the crucial feeling of mastery over the game.

A final approach focuses on providing tools that maximize the ability of the trailing player to catch up with the leading player. The key for the game designer is to think of ways to maintain challenge, reward, and progress for the unskilled player without severely hampering the skilled player. One interesting and explicit example of this is found in *Diddy Kong Racing* (Nintendo of America, 1997). In this game, the racer can collect bananas along the roadway.

Each banana increases the top speed of your car. The player can also collect missiles to fire forward at the leading cars. Each time you hit a car with a missile it not only slows the car's progress, but it jars loose several bananas that the trailing player can pick up. Thus, trailing players have tools that they can use to catch the leaders even if the leaders are not making any driving mistakes. The chief distinction between this and dynamic difficulty adjustment is that the game is not modifying skills based on success. Instead, the rules of the game provide the trailing player with known advantages over the leader.

Rewarding Players Appropriately

Explicit or slow reinforcement schedules may cause users to lose motivation and quit playing a game. Because playing a game is voluntary, games need to quickly grab the user's attention and keep them motivated to come back again and again. One way to accomplish this is to reward players for continued play. Theories of positive reinforcement suggest behaviors which lead to positive consequences tend to be repeated. Thus, it makes sense that positive reinforcement can be closely tied to one's motivation to continue playing a game. However, it is less clear which types of reinforcement schedules are most effective.

Although it's not necessarily the model that should be used for all games, research suggests that continuous reinforcement schedules can establish desired behaviors in the quickest amount of time (Domjan, 2010; Mazur, 2006). Unfortunately, once continuous reinforcement is removed, desired behaviors extinguish very quickly. Use of partial reinforcement schedules take longer to extinguish desired behaviors, but may take too long to capture the interest of gamers. Research suggests that variable ratio schedules are the most effective in sustaining desired behaviors (Jablonsky & DeVries, 1972). This kind of schedule is a staple of casino gambling games in which a reward is presented after a variable number of desired responses. Overall, there is no clear answer. Creating a game that establishes immediate and continued motivation to continue playing over long periods of time is a very complex issue.

Another facet of reinforcement systems that may impact enjoyment of a game is whether the player attributes the fact that they have been playing a game to extrinsic or intrinsic motivations. Intrinsic explanations for behavior postulate that the motivators to perform the behavior come from the personal needs and desires of the person performing the behavior. Whereas extrinsically motivated behaviors are those that people perform in order to gain a reward from or please other people. In research on children's self-perceptions and motivations, Lepper, Greene, and Nisbett (1973) discovered that children who were given extrinsic rewards for drawing were less likely to continue drawing than those who had only an intrinsic desire to draw. The conclusion that they drew is that children perceived their motivation to draw as coming from extrinsic sources and thus discounted their self-perception that they liked to draw.

The same may be true of reward systems in games (Lepper & Malone, 1987; Malone, 1981). To a certain degree, all reinforcement systems in games are extrinsic because they are created or enabled by game developers. But, some reward systems are more obviously extrinsic than others. For instance, imagine the following rewards that could be associated with combat in a fantasy role-playing game (RPG). The player who slays a dragon with the perfect combination of spell casting and swordplay may acquire the golden treasure that the dragon was hoarding. In this situation, the personal satisfaction comes from being powerful enough to win and smart enough to choose the correct tactics. The gold is an extrinsic motivator. The satisfaction is intrinsic. By analogy from Lepper, Green and Nisbett's research, feelings of being powerful and smart (intrinsic motivators) are more likely to keep people playing than extrinsic rewards.

Collecting and Completing

The chief goal of many games is to acquire all of the available items, rewards or knowledge contained in the game. In games such as *Pokemon Crystal* (Nintendo Japan, 2000) the central challenge is to acquire as many of the Pokemon characters as you can and learn all of their skills well enough to outsmart your opponent at selecting the right characters for a head-to-head competition. Not coincidentally, the catch phrase for the *Pokemon Crystal* game is “Gotta catch ‘em all!”

This game mechanic is also used by numerous games to add depth and repeat play. Though this isn’t the primary mechanic in *Madden NFL 06* (Electronic Arts Inc., 2005), the ability to collect electronic player cards and use them strategically in games provides incentive for gamers to experiment with much of the content that they may not experience if playing through a standard season.

Story

Characters and narrative help gamers attach meaning and significance to action sequences. There are those in the games industry that propose that many games neither have nor need a story. It is our contention that the key to understanding narrative in games is to realize that storylines may be both embedded in the game, or they may emerge in the course of playing a game.

When most consumers think about story, they think about embedded storylines (Levine, 2001). *Final Fantasy X* (Square Co., Ltd., 2001) tells its story by cutting action sequences with a series of full-motion cut scenes, real-time cut scenes and player-driven character interactions. The embedded story forms a central part of the appeal of the game. But Ken Levine (2001) points out that much of the narrative in a game emerges in the successes and failures experienced by players throughout the course of the game. This is especially true of multiplayer games, in which story is often generated exclusively by the interactions of the participants. As Levine describes it, the story is generated by replaying an abstract narrative structure with a strict set of rules and possibilities within a novel set of circumstances. Sporting events both within and outside of the video game world provide an excellent example of this type of narrative. No author scripted the result of the last World Cup tournament, but each such event has the potential to be an epic narrative for both participants and viewers.

Technological Innovation Drives Design

There is a great deal of pressure on designers to utilize new technologies which may break old interaction models. The desire to experience ever more realistic and imaginative games has pushed game developers into engineering and computer science research domains. Likewise, technology often drives game design in order to showcase new capabilities. The constant demand for novelty can be strong enough incentive for game makers to try untried designs, “spruce up” familiar interfaces and break rules of consistency. For example, the original *NFL2K* series (Sega, 1999) sported a new interface model in which users selected interface areas by holding the thumbstick in a direction while pressing a button. It is possible that Sega chose the new design primarily because it was new and different from existing interfaces. It required the somewhat new (at that time) mechanics of the thumbstick on the controller, it minimized movement because one could point directly at any given item in the menu, and it was cleverly shaped like a football (which made more sense for *NFL2K* than for another sport). However, errors due to more error-prone targeting and the hold and click interaction metaphor made the system harder for first-time players to use.

Perceptual-motor Skill Requirements

The way that functions are mapped onto available input devices can determine the success or failure of a game. A crucial part of the fun in many games comes from performing complex perceptual-motor tasks. While today’s arcade-style games include increasingly more sophisticated strategic elements, a core element of many games is providing the ability to perform extremely dexterous, yet satisfying physical behaviors. These behaviors are usually quick and well-timed responses to changes and threats in the environment. If the controls are simple enough to master and the challenges increase at a reasonable difficulty, these mostly physical responses can be extremely satisfying (Csiksentmihalyi, 1990).

Problems can arise when games require unfamiliar input devices. This is a common complication in console game usability research because new console systems usually introduce new input device designs unique to that system (see Figure 1). Furthermore, game designers often experiment with new methods for mapping the features of their games to the unique technical requirements and opportunities of new input devices. Unfortunately, this does not always result in a better gameplay experience.

Insert Figure 1 about here

Balancing Multiplayer Games

As we've seen, different strategies can be used to support a broad range of skill player skill levels. While this is complicated in single-player games, it becomes even more daunting when players of different skills make up both the opponents and the allies. By far the most common strategy for regulating the challenge level of online multiplayer games is to provide strong matchmaking tools. *Internet Backgammon* (Microsoft Windows XP, 2001) automatically matches players from around the world based on their self-reported skill level. Other games use algorithms that count player winning percentages and strength of opponent.

Another strategy seeks to solve skill balance problems outside of the game by offering players a wide array of arenas and/or game types. Rather than actively connecting players of like skill, this approach provides a broad array of places to play and allows players to self-select into game types that suit their style and attract the players that they want to associate with. A final approach, used frequently by instant messaging clients (e.g., Yahoo IM, MSN Messenger), is to make it easy to start a game with friends. Though skill levels of friends may not always match, you are presumably less likely to perceive the match as unfair against people that you know.

Game designers also employ a variety of in-game strategies to balance out the competition. The most common way to allow less skilled players to compete effectively with more skilled players is to play team games. Many games allow players to take on a variety of roles. Capture the Flag is a common backyard tag-style game that forms the basis for a game type in many first-person shooters. Some players go on offense to capture the opponent's flag, some stay back to defend their flag. Likewise, some players may take a long-range sniper weapon to frustrate the enemy, while others take weapons that are more effective at close range. The defending and sniping roles can be more comfortable for some novice players because they allow the player to seek protective cover, they require less knowledge of the play field and one-on-one combat can be avoided.

Some first person shooters also allow the game host to set "handicaps" for successful players and bonuses for less successful players. For example, one version of Unreal Tournament increased the size of the player's character with every point that they won, and decreased the size of the character every time that the character died. The size differences made successful players easier targets and unsuccessful players more difficult targets.

Enabling Social Networking

Skill level is not the only dimension of importance. Systems such as the Xbox Live service seek to group players of similar attitudes and behaviors. The goal is to place more antagonistic players with like-minded others who appreciate tough talk, while protecting players who don't want to or shouldn't be exposed to aggressive or offensive language.

This is just one of many tools that games have developed to promote group play and improve communication between online players. Most online games include some form of in-game messaging. Often this messaging is tailored for fast and efficient communication of key functions or timely game events. Most massively multiplayer online role-playing games have negotiation systems that help people trade objects safely and efficiently. Most first-person shooters present automatic messages telling you which players have just been killed by whom. Real-time strategy games allow gamers to set flares or markers on the map to notify your allies of key positions. Each game genre has a key set of in-game communication tools to support the game play.

More and more frequently game designers are starting to embed cooperative tools into the environment itself. Much of the benefit of vehicles in first-person shooter games comes from their use as cooperative tools. Used in concert these tools can often be extremely effective. From a game design perspective, they provide a great incentive for people to come together, strategize and work cooperatively. These shared successes can be a huge part of the fun in online games.

Massively multiplayer games employ a wide variety of incentives for players to form groups. Success comes from taking on missions that are far too dangerous for any single player to accomplish on their own. Players choose a role and learn to compliment the strengths of the other members of their group in order to overcome fearful enemies and accomplish great quests. Recent massively multiplayer games have invested even more in the creation of large-scale guilds. Most massively multiplayer games intentionally keep large areas of information secret from their players – to provide the need and the opportunity for symbiotic relationships between experts and novices. In addition, modern MMO’s (massively multiplayer online games) often provide pyramid-scheme style bonuses in wealth and experience to the leaders and captains of guilds. In return, members receive physical, material, educational and social protection from their leaders and peers.

Mobile Gaming Environmental Factors

The success of Tetris points out several fundamental truths of portable gaming. The most successful portable games fit efficiently and effectively into the user’s lifestyle. They don’t require prolonged concentration and are robust to environmental distractions - allowing the busy person to pause when needed and resume without difficulty. This is why so many successful portable games are brief, turn-based or pause-able. They’re with you when you have time to play – on the bus to school, or train to work, in the back of a car on the way to practice or at a friend’s house for head-to-head competition. They’re challenging, but they don’t require special experience or knowledge to be successful at the start of the game. Many portable successes are simple, familiar or particularly resistant to creating failure states.

Many of the same fundamental truths apply to mobile phone games. Mobile games appear poised to break new ground in social gaming due to increased accessibility to communication, connectivity, location and identity features. The barriers to mobile phone gaming come largely from lack of hardware and software standardization. There are dozens of mobile game development platforms and thousands of hardware form factors. For example, while iPhone has a substantial amount of mind-share in the mobile space, it was estimated that there were a little over 6 million of them in operation in the US in 2009 (Nielsenwire, 2009).

Important Factors in Game Testing and Evaluation

Most game genres are subtly different in the experiences that they provoke. It may seem obvious that the point of game design is making a fun game. Some games are so fun that people will travel thousands of miles and spend enormous amounts of money to participate in gaming events. However, we would like to propose a potentially controversial assertion. That the fundamental appeal of some games lies in their ability to challenge, to teach, to bring people together, or to simply experience unusual phenomena. Likewise, the definition of fun may be different for every person. When you play simulations games, your ultimate reward may be a combination of learning and mastery. When you play something like the *MTV Music Generator* (Codemasters, 1999), your ultimate reward is the creation of something new. When you go online to play card games with your uncle in Texas, you get to feel connected. *Flight SimulatorX* (Microsoft Corporation, 2006) lets people understand and simulate experiences that they always wished they could have. While these may be subcomponents of fun in many cases, there may be times when using "fun" as a synonym for overall quality will lead to underestimations of the quality of a game. While a fun game is often synonymous with a good game, researchers are warned to wisely consider which measures best suit the evaluation of each game that they evaluate.

Game Designer Intent

As stated earlier, the design goal of a game is to create a pleasurable experience and to help users have fun, and the main user goal is to have fun. Goals in a game are not necessarily derived from external user needs as in productivity applications. In games, goals are defined in accordance with the game designer's vision, which is a novel position because historically, success in has been defined by the accomplishment of user-tasks and goals (Pagulayan, Gunn, & Romero, 2006). When approaching a game for user-centered design or testing, it is best to assume the role of facilitating the designer's vision for the game (Pagulayan & Steury, 2004; Pagulayan, Steury, Fulton, & Romero, 2003) because many times, it is only the designer who can recognize when the player experience is not being experienced as intended. In traditional usability testing, it is often very recognizable when there is user error, but not so in games.

Davis, Steury, and Pagulayan (2005) discuss a case study in the game *Brute Force* (2003) which revealed that players were not encountering certain gameplay features early enough in their gameplay experience. It was not the case that players were failing, but that players were taking much longer to play through the second mission than intended. By understanding the design vision, the authors were able to work with the designers to provide feedback which resulted in shortening the second mission, and also re-ordering other missions to match the design intent of the game.

Ease of Use

The ease of use of a game's controls and interface is closely related to fun ratings for that game. Think of this factor as a gatekeeper on the fun of the game. If the user must struggle or cannot adequately translate their intentions into in-game behaviors, they will become frustrated. This frustration can lead the user to perceive the game as being unfair or simply inaccessible (or simply not fun). Thus, it becomes very clear why usability becomes very important in games. Ease of use should be evaluated with both usability and attitude-measurement methodologies, which are discussed later in the chapter.

Basic Mechanics. The basic mechanics of a game are best imagined by this example. In chess, each player expresses more strategic desires by turn-taking, movement, checking, and capturing. These actions are the core mechanics. Combined with the board and pieces, roles and rules, and strategies and situations, they make up the experience of chess. Of these mechanics, movement is one of the most important; yet, it is simple. Each piece has a role with defined movements, but it also contains the magic of chess. The power of the Queen is expressed by the lack of restraint on her movement. Every game, similarly, has a set of core mechanics. Getting the core mechanics right is fundamental to making a great game.

Starting a Game. Starting the kind of game that the user wants is an easily definable task with visible criteria for success. This is something one can measure in typical usability laboratories. Though designers often take game shell (the interface used to start the game) design for granted, a difficult or confusing game shell can limit users' discovery of features and impede their progress towards enjoying the game. The most immediate concern for users can be starting the kind of game that they want to play. Games often provide several modes of play. When the game shell is difficult to navigate, users may become frustrated before they have even begun the game. For example, we have found that many users are unable to use one of the common methods that sports console games use to assign a game controller to a particular team. This has resulted in many users mistakenly starting a computer vs. computer game. Depending on the feedback in the in-game interface, users may think that they are playing when, in fact, the computer is playing against itself! In these cases users may even press buttons, develop incorrect theories about how to play the game, and become increasingly confused and frustrated with the game controls. The most effective way to avoid these problems is to identify key user tasks and usability test them.

Pagulayan et al. (2003) discuss a case study where they found issues with difficulty settings in the game shell. In early usability testing, participants were having problems with setting the difficulty level of opponents in *Combat Flight Simulator* (Microsoft Corporation, 1998). This is a case where the users' gameplay experience would have been quite frustrating because of a usability error in the game shell if it were not addressed.

Tutorials or Instructional Gameplay. Tutorials are sometimes necessary to introduce basic skills needed to play the game. In this situation, instructional goals are easily translated into the usability lab with comprehension tasks and error rates.

One of the risks of not testing tutorials or instructional missions is inappropriate pacing, which can often result from an ill-conceived learning curve at the start of the game. Many games simply start out at too difficult a challenge level. This is an easy and predictable trap for designers and development teams to fall into because when designers spend months (or even years) developing a game, they risk losing track of the skill level of the new player. A level that is challenging to the development team is likely to be daunting to the beginner.

Unfortunately, the reverse can also be troubling to the new user. Faced with the task of addressing new players, designers may resort to lengthy explanations. Frequently, developers will not budget time to build a gradually ramping learning process into their game. Instead, they may realize late in the development cycle that they need to provide instruction. If this is done too abruptly, the learning process can end up being mostly explanation, and to be frank, explanation is boring. The last thing that you want to do is to bore your user with a longwinded explanation of what they are supposed to do when they get into your game. It is best to learn in context and at a measured pace or users may just quit the game.

A very positive example is the first level of *Banjo Kazooie* (Nintendo of America, 2000). At the start the player is forced to encounter a helpful tutor and listen to a few basic objectives. Then they must complete some very basic objectives that teach some of the basic character abilities. Much of the tutorial dialogue may be skipped, but the skills necessary to continue must be demonstrated. In this way, the game teaches new skills but never requires tedious instruction. The player learns primarily by doing. All of this is done in the shadow of a very visible path onto the rest of the game so the user never loses sight of where they need to go.

Camera. The camera perspective (i.e., the view that the player sees into the virtual world) in a game is often treated as an advanced game mechanic. That is, known difficulties in seeing the environment and threats and opportunities can be exploited to create challenge and tension. When not done effectively, this can result in a poor experience for many users who are powerless to see something that they believe they could see in real-life. This can be frustrating, resulting in a loss of immersion in the game world.

A 3D isometric view is a very effective camera perspective for viewing game boards or maps or moving a character through a game world. But there is an important tradeoff to consider when thinking about how much distance to have between the camera and the objects of view. The farther away the camera is, the more environment the user can consider in their strategy. For this reason, it's generally easier to drive a car, command troops, or play Monopoly from a bird's eye view. On the other hand, you may lose a very important part of the visceral experience by being too far away. Viewing a race from the inside of the car will cut down on your awareness of competitor cars and upcoming turns, but it feels a lot faster and more intense. Because of this inherent tradeoff and different preferences around it, most racing games allow the user to choose between several viewing distances. The same effect can be seen in a shooting game. Gears of War feels extra gritty and dangerous in-part due to the intentional placement of the camera lower over the shoulder of the main character than in many shooting games (Microsoft Game Studios, 2006).

It's also worth noting that tight spaces can cause havoc for over-the shoulder cameras. User research can identify problem areas and suggest custom cameras to help users see what they are supposed to see. Because designers know, and professional software testers learn, where to go and what to avoid, user-centered design can be a necessary way to discover camera blind spots that can be extremely frustrating to the mass of users who only play through an experience once.

At the extreme end, with very good tuning, nontraditional camera perspectives can be part of the fun of a game. Crash Bandicoot successfully kept the game fresh and increased dramatic tension by switching the camera after several levels to look directly into frightened Crash's eyes as he ran from a charging dragon (Universal Interactive Studios, 2001). Likewise, Resident Evil increased suspense and fear in its games by employing fixed cameras at awkward locations and forcing people to enter areas and fight zombies blindly (Capcom, 2005).

In-game Interfaces. In-game interfaces are used primarily to deliver necessary status feedback and to perform less-frequent functions. We measure effectiveness with more traditional lab usability testing techniques and desirability with attitude-measurements such as surveys (see next section for example).

Some PC games make extensive use of in-game interfaces to control the game. For example, simulation and RTS games can be controlled by keyboard and mouse presses on interface elements in the game. Usability improvements in these interfaces can broaden the audience for a game by making controls more intuitive and reducing tedious aspects of managing the game play. In-game tutorial feedback can make the difference between confusion and quick progression in learning the basic mechanisms for playing. In this situation, iterative usability evaluations become a key methodology for identifying problems and testing their effectiveness (see next section for example).

Many complex PC and console video games make frequent use of in-game feedback and heads-up displays (HUD) to display unit capabilities and status. For example, most flight combat games provide vital feedback about weapons systems and navigation via in-game displays. Without this feedback, it can be difficult to determine distance and progress towards objectives, unit health and attack success. This feedback is crucial for player learning and satisfaction with the game. With increasing game complexity and three-dimensional movement capabilities, these displays have become a crucial part of many game genres. Usability testing is required to establish whether users can detect and correctly identify these feedback systems. See Pagulayan et al., (2003) for detailed example of usability testing the HUD in *MechWarrior 4: Vengeance* (Microsoft Corporation, 2000).

Mapping Input Devices to Functions. A learnable mapping of buttons, keys or other input mechanisms to functions is crucial for enjoying games. We measure effectiveness with usability techniques and desirability with attitude-measurements. Without learnable and intuitive controls, the user will make frequent mistakes translating their desires into onscreen actions. We have seen consistently that these kinds of mistakes are enormously frustrating to users, because learning to communicate one's desires through an 8-button input device is not very fun. The selection of keys, buttons, and other input mechanisms to activate particular features is often called "control-mapping." Players tend to feel that learning the control-mapping is the most basic part of learning the game. It is a stepping stone to getting to the fun tasks of avoiding obstacles, developing strategies, and blowing things up.

By contrast with other ease of use issues, evaluating the control-mapping may involve as much subjective measurement as behavioral observation. Button-presses are fast, frequent, and hard to collect automatically in many circumstances. Furthermore, problems with control-mappings may not manifest themselves as visible impediments to progress, performance, or task time. Instead, they may directly influence perceptions of enjoyment, control, confidence, or comfort. Due to differences in experience levels and preferences between participants, there may also be significant variation in attitudes about how to map the controls.

Dissatisfaction with the controller design can also be a central factor that limits enjoyment of all games on a system. For example, the results of one whole set of studies on the games for a particular console game system were heavily influenced by complaints about the system's controller. Grasping the controller firmly was difficult because users' fingers were bunched up and wrists were angled uncomfortably during game play. Ratings of the overall quality of the games were heavily influenced by the controller rather than the quality of the game itself.

Because of these concerns and the importance of optimizing control-mappings, we recommend testing control-mappings with both usability and attitude-assessment methodologies.

Challenge

Challenge is distinct from ease of use and is measured almost exclusively with attitude-assessment methodologies. This can be a critical factor to the enjoyment of a game, and obviously can be highly individualized and is rightly considered subjective.

Consumers may have difficulties distinguishing the "appropriate" kinds of challenge that result from calculated level and obstacle design from the difficulty that is imposed by inscrutable interface elements or poor communication of objectives. In either case, the result is the same. If not designed properly, the player's experience will be poor. Thus, it is up to the user research professional to make measurement instruments that evaluate the appropriateness of the challenge level independently of usability concerns. In one example, Pagulayan et al. (2003) used attitude-assessment methodologies to determine the final design of the career mode in *RalliSport Challenge* (Microsoft Corporation, 2002). In this situation, finding a solution to the design problem was not necessarily related to ease of use issues, or other usability-related issues. The final design was based on what was most fun and appropriately challenging for users.

Pace

We define pace as the rate at which players experience new challenges and novel game details. We measure this with attitude-measurement methodologies.

Most designers will recognize that appropriate pacing is required to maintain appropriate levels of challenge and tension throughout the game. You might think of this as the sequence of obstacles and rewards that are presented from the start of the game to the end. However, the way a designer will address pace will depend on a variety of issues, including game type, game genre, and their particular vision for the gameplay experience. In a tennis game, pace can be affected by a number of things, including the number of cut-scenes in between each point, to the actual player and ball movement speed (Pagulayan & Steury, 2004).

One group at Microsoft uses a critical juncture analogy to describe pacing. As a metaphor, they suggest that the designer must attend to keeping the user's attention at 10 seconds, 10 minutes, 10 hours and 100 hours. The player can always put down the game and play another one, so one must think creatively about giving the user a great experience at these critical junctures. Some games excel at certain points but not others. For example, the massively multiplayer game may have the user's rapt attention at 10 seconds and 10 minutes. And the fact that hundreds of thousands pay $10/month to continue playing indicates that these games are very rewarding at the 100 hour mark. But anyone who has played one of these games can tell you that they are extremely difficult and not too fun to play at the 10-hour mark. At this point, you are still getting "killed" repeatedly. That is no fun at all. Arcade games obviously take this approach very seriously. Though they may not scale to 100 hours, good arcade games attract you to drop a quarter and keep you playing for long enough to make you want to spend another quarter to continue.

Pacing may also be expressed as a set of interwoven objectives much like the subplots of a movie. Again, *Banjo Kazooie* provides an excellent example of good pacing. Each level in *Banjo Kazooie* contains the tools necessary to complete the major objectives. Finding the tools is an important part of the game. New abilities, objectives, skills, and insights are gradually introduced as the player matures. While progressing towards the ultimate goal (of vanquishing the evil witch and saving the protagonist's sister), the player learns to collect environmental objects that enable them to fly, shoot, become invincible, change shape, gain stamina, add extra lives, and unlock new levels. This interlocking set of objectives keeps the game interesting and rewarding. Even if one is unable to achieve a particular goal, there are always sets of sub goals to work on. Some of which may provide cues about how to achieve the major goal.

Summary

Attitude methodologies are better apt to measure factors such as overall fun, graphics and sound, challenge, and pace. The typical iterative usability is an exploratory exercise designed to uncover problem areas where the designer's intentions don't match the user's expectations, as a result, we typically choose to not use usability test to assess "fun" or challenge. When attempting to assess attitudinal issues such as "overall fun" and "challenge" we make use of a survey technique that affords testing larger samples. Internally, we have adopted the term Playtest or sometimes Consumer Playtest for this technique. At the same time, we use typical iterative usability methods to determine design elements which contribute to or detract from the experience of fun.

User Research in Games

Introduction to Methods– Principles in Practice

In the following section we propose various methodologies and techniques that attempt to accurately measure and improve game usability and enjoyment. Many of the examples are taken from techniques used and developed by the Games User Research group at Microsoft Game Studios.

Our testing methods can be organized by the type of data being measured. At the most basic level, we categorize our data into two types: *behavioral* and *attitudinal*. *Behavioral* refers to observable data based on performance, or particular actions performed by a participant that one can measure. This is very similar to typical measures taken in usability tests (e.g., time it takes to complete a task, number of attempts it takes to successfully complete a task, and task completion). *Attitudinal* refers to data that represent participant opinions or views, such as subjective ratings from questionnaires or surveys. These are often used to quantify user experiences. Selection of a particular method will depend on what variables are being measured and what questions need to be answered.

Another distinction that is typically made is between *formative* and *summative* evaluations, which we apply to our testing methods as well. *Formative* refers to testing done on our own products in development. *Summative* evaluations are benchmark evaluations, either done on our own products, or on competitor products. It can be a useful tool for defining metrics or measurable attributes in planning usability tests (Nielsen, 1993), or to evaluate strengths and weaknesses in competitor products for later comparison (Dumas & Redish, 1999).

While these methods are useful, they do not allow us to address issues with extended gameplay, that is, issues that may arise after playing the game for a couple of days or more. This is problematic, because one of the key challenges in game design is longevity. With the competition, the shelf life of a game becomes very limited.

Usability Techniques

Traditional usability techniques can be used to address a portion of the variables identified as important for game design. In addition to measuring performance, we use many standard usability techniques to answer "how" and "why" process-oriented questions. For example, how do users perform an attack, or why are controls so difficult to learn? We use (a) structured usability tests, (b) rapid iterative testing and evaluation (RITE) (Medlock et al., 2002; Medlock et al., 2005), and (c) other variations and techniques, including open-ended usability tasks, paper prototypes, and gameplay heuristics. For clarity of presentation, each technique will be discussed separately, followed by a case study. Each case study will only contain information pertinent to a specific technique, thus examples may be taken from a larger usability test.

Structured Usability Test. A structured usability test maintains all the characteristics that Dumas & Redish (1999) propose as common to all usability tests: (a) goal is to improve usability of the product, (b) participants represent real users, (c) participants do real tasks, (d) participant behavior and verbal comments are observed and recorded, and (e) data are analyzed, problems are diagnosed, and changes are recommended. We have found that issues relating to expectancies, efficiency, and performance interaction are well-suited for this type of testing. Some common areas of focus for structured usability testing are in game shell screens, or control schemes. The game shell can be defined as the interface in which a gamer can determine and or modify particular elements of the game. This may include main menus, options screens (i.e., audio, graphics, controllers, etc.).

An example which uses this method is in the *MechCommander 2* (Microsoft Corporation, 2001) usability test. Portions of the method and content have been omitted.

<u>Case Study: MechCommander 2 Usability Test</u>. *MechCommander 2 (MC2)* is a PC real-time strategy (RTS) game where the gamer takes control of a unit of *mechs* (i.e., large giant mechanical robots). One area of focus for this test was on the *'Mech Lab*, a game shell screen where mechs can be customized (see Figure 2). The gamer is able to modify weaponry, armor, and other similar features, and are limited by constraints such as heat, money, and available slots.

Insert Figure 2 about here

The first step in approaching this test was to define the higher-order goals. Overall, the game shell screens had to be easy to navigate, understand, and manipulate, not only for those familiar with mechs and the mech universe, but also for RTS gamers who are not familiar with the mech universe. Our goal was for gamers to be able to modify/customize mechs in the *'Mech Lab*.

As mentioned, one of the most important steps in this procedure is defining the participant profile(s). Getting the appropriate users for testing is vital to the success and validation of the data since games are subject to much scrutiny and criticism from its gamers. To reiterate, playing games is a choice. For *MC2*, we defined two participant profiles which represented all of the variables we wanted to cover. The characteristics of interest included those who were familiar with RTS games (experienced gamers) and those who were not RTS gamers (novice gamers). We also wanted gamers that were familiar with the mech genre, or the mech universe. Overall, we needed a landscape of gamers that had some connection or interest that would make them a potential consumer for this title, whether through RTS experience, or mech knowledge.

Tasks and task scenarios were created to simulate situations that a gamer may encounter when playing the game. Most importantly, tasks were created in order to address the pre-defined higher order goals. Participants were instructed to talk aloud, and performance metrics were recorded (i.e., task completion, time). The following are examples from the usability task list.

1. Give the **SHOOTIST** jumping capabilities.

This task allowed us to analyze participant expectations. To succeed in this task, one had to select a "CHANGE WEAPONS" button from a different game shell screen which brought them into the *'Mech Lab*. If the task was to change a weapon on a mech, the terminology would probably have been fine. Thus, this task had uncovered two main issues, (a) could they get to the *'Mech Lab* where you modify the mech, and (b) were they able to discover the process of modifying the mech. It was accurately predicted that gamers would have difficulties with the button terminology. Thus, that button was changed to "MODIFY MECH".

To change the components (i.e., add the jump jets), participants could either select the item, and drag it off the mech, or select the item, and press the "REMOVE" button (see Figure 2). One unexpected issue that arose was that participants unknowingly removed items because the distance required for removing an item was too small. The critical boundary that was implemented was too strict. In addition, participants had difficulties adding items by dragging and dropping because the distance required for adding an item was too large (i.e., the item would not stay on the mech unless it was placed exactly on top of the appropriate location). Appropriate recommendations were made, and implemented.

2. Replace the **MG Array** with the **Flamer**.

One of the constraints presented for modifying a mech was heat limit. Each weapon had a particular heat rating. For example, if the heat limit for a mech is 35, and the current heat rating is 32, only weapons with a rating of 3 or fewer could be added. In this task, the "Flamer" had a heat rating much larger than the "MG Array", thus making impossible to accomplish this task without removing more items. The issues here were the usability of the heat indicator, heat icons, and the discoverability of heat limit concept. None of the participants figured this out. Recommendations included changing the functionality of the Heat Limit Meter, and to add better visual cues to weapons that exceed the heat limit. Both of these changes were implemented.

Rapid Iterative Testing and Evaluation Method (RITE). Medlock et al. (2002; 2005) have documented another common usability method used by the Games User Research Group at Microsoft Game Studios, which they refer to as the RITE method. In this method, fewer participants are used before implementing changes, but more cycles of iteration are performed. With RITE, it is possible to run almost two to three times the total sample size of a standard usability test. However, only one to three participants are used per iteration with changes to the prototype immediately implemented before the next iteration (or group of one to three participants).

The goal of the RITE method is to be able to address as many issues and fixes as possible in a short amount of time in hopes of improving the gamer's experience and satisfaction with the product. However, the utility of this method is entirely dependent on achieving a combination of factors (Medlock et al., 2002; 2005). The situation must include: (a) a working prototype, (b) the identification of critical success behaviors, important, but not vital behaviors, and less important behaviors, (c) commitment from the development team to attend tests and immediately review results, (d) time and commitment from development team to implement changes before next round, and (e) the ability to schedule and/or run new participants as soon as the product has been iterated. Aside from these unique requirements, planning the usability test is very similar to more traditional structured usability tests.

It is very helpful to categorize potential usability issues into four categories: (a) clear solution, quick implementation, (b) clear solution, slow implementation, (c) no clear solution, and (d) minor issues. Each category has implications for how to address each issue. In the first category, fixes should be implemented immediately, and should be ready for the next iteration of testing. In the second category, fixes should be started, in hopes that it can be tested by later rounds of testing. For the third and fourth category, more data should be collected.

The advantage of using the RITE method is that it allows for immediate evaluation and feedback of recommended fixes that were implemented. Changes are agreed upon, and made directly to the product. If done correctly, the RITE method affords more fixes in a shorter period of time. In addition, by running multiple iterations over time we are potentially able to watch the number of usability issues decrease. It provides a nice, easily understandable, and accessible measure. In general, the more iterations of testing, the better. However, this method is not without its disadvantages. In this situation, we lose the ability to uncover unmet user needs or work practices, we are unable to develop a deep understanding of gamer behaviors, and we are unable to produce a thorough understanding of user behavior in the context of a given system (Medlock et al., 2002; 2005).

The following example demonstrates how the RITE method was used in designing the *Age of Empires II: The Age of Kings* (Microsoft Corporation, 1999) tutorial. Again, portions of the method and content have been omitted. See Medlock et al., (2002) for more details.

Case Study: *Age of Empires II: The Age of Kings Tutorial*. *Age of Empires II: The Age of Kings* (*AoE2*) is an RTS game for the PC where the gamer takes control of a civilization spanning over a thousand years from the Dark Ages through the late medieval period. In this case study, a working prototype of the tutorial was available, and critical concepts and behaviors were defined. Also, the development team was committed to attending each of the sessions. And, they were committed to quickly implementing agreed upon changes. Finally, the resources for scheduling were available. The key element in this situation for success was the commitment from the development team to work in conjunction with us.

In the *AoE2* tutorial, there were four main sections, (a) Marching and fighting (movement, actions, unit selection, the “fog of war”[1]), (b) Feeding the army (resources, how to gather, where to find), (c) Training the troops (use of mini-map, advancing through ages, build and repair buildings, relationship between housing and population, unit creation logic) , and (d) Research and technology (upgrading through technologies, queuing units, advancing through ages). Each of these sections dealt with particular skills necessary for playing the game. In essence, the tutorial had the full task list built in. In the previous case study, this was not the case.

At a more abstract level, the goals of the tutorial had to be collectively defined (with the development team). In more concrete terms, specific behaviors and concepts that a gamer should be able to perform after using the tutorial were identified, then categorized into the three levels of importance: (a) essential behaviors that users must be able to perform without exception, (b) behaviors that are important, but not vital to product success, and (c) behaviors that were of lesser interest. Table 2 lists some examples of concepts and behaviors from each of the three categories. This is an important step because it indirectly sets up a structure for decision rules to be used when deciding what issues should be addressed immediately, and what issues can wait.

Insert Table 2 about here

The general procedure for each participant was similar to other usability tests we often perform. If participants did not go to the tutorial on their own, they were instructed to do so by the specialist.

---

[1] The *fog of war* refers to the black covering on a mini-map or radar that has not been explored yet by the gamer. The fog of war “lifts” once that area has been explored. Use of the fog of war is most commonly seen in RTS games.

During the session, errors and failures and were recorded. In this situation, an error was defined as anything that caused confusion. A failure was considered an obstacle that prevented participants from being able to continue. After each session, a discussion ensued among the specialist and the development team to determine what issues (if any) warranted an immediate change at that time.

In order to do this successfully, certain things had to be considered. For example, how can one gauge how serious an issue is? In typical usability tests, the proportion of participants experiencing the error is a way to estimate its severity. Since changes are made rapidly here, the criteria must change to the intuitively estimated likelihood that users will continue to experience the error. Another thing to consider is clarity of the issue, which was assessed by determining if there is a clear solution. We have often found that if issues do not have an obvious solution then the problem not fully understood. And finally, what errors or failures were essential, important, or of lesser interest. Efforts of the development team should be focused on issues related to the essential category when possible.

At this point we broke down the issues into three groups. The first group included issues with a solution that could be quickly implemented. Every issue in this group was indeed quickly implemented before the next participant was run. The second group consisted of issues with a solution that could not be quickly implemented. The development team began working on these in hopes it could be implemented for later iterations of the test. Finally, there were issues with no clear solutions. These issues were left untouched because more data were needed to assess the problem at a deeper level (i.e. more participants needed to be run). Any fixes implemented in the builds were kept as each participant was brought in. Thus, it was possible that many of the participants experienced a different version of the tutorial over the duration of testing.

Overall, seven different iterations were used across sixteen participants. Figure 3 represents the number of errors and failures recorded over time. The number of errors and failures gradually decreased across participants as new iterations of the build were introduced. By the seventh and final iteration of the build, the errors and failures reliably were reduced to zero.

Insert Figure 3 about here

Although we feel that the *AoE2* tutorial was an enormous success, largely due to the utilization of the RITE method, the method does have its disadvantages and should be used with caution. Making changes when issues and/or solutions are unclear may result in not solving the problem at all, while creating newer usability problems in the interface. We experienced this phenomena a couple of times in the *AoE2* study.

Also, making too many changes at once may introduce too many sources of variability and create new problems for users. Deducing specifically the source of the new problem becomes very difficult. A related issue is not following up changes with enough participants to assess whether or not the solution really addressed the problem. Without this follow up, there is little evidence supporting that the implementations made were appropriate (which is a problem with traditional usability methods as well). The last thing to consider is that other important usability issues that may surface less frequently are likely to be missed. Using such small samples between iterations allows for the possibility that those less occurring issues may not be detected.

Variations on Usability Methods. Now that we have presented two general types of usability testing, it is worthy to mention some variations on these methods, (a) open-ended tasks, (b) paper prototyping, and (c) gameplay heuristics.

In general, it is often recommended that tasks in usability tests be small, with a specified outcome (e.g., Nielsen, 1993). However, we have found situations where the inclusion of an open-ended task yields important data as well. In many usability studies, we often include an open-ended task where participants are not instructed to perform or achieve anything in particular.

In other words, there is no specified outcome to the participant. These tasks can be used to analyze how gamers prioritize certain tasks or goals in a non-linear environment. These tasks are also useful in situations where structured tasks may confound the participant experience or situations where we are interested in elements of discovery. An example of an open-ended task is as follows.

1. Play the game as if you were at home. The moderator will tell you when to stop.

This example was taken from a usability test on *Halo: Combat Evolved* (Microsoft Corporation, 2001) (Pagulayan et al., 2003). Participants were presented with a mission with no instruction other than playing as if they were at home. Traditional usability metrics were not tracked or used. Instead, the focus was watching players and the tactics and strategies they employed while playing through the game. Results demonstrated that novice players would start firing at enemies as soon as they were visible, which was not how the designers intended combat to occur. The design intent was for combat to occur at closer ranges.

By allowing participants to play through the mission with no structured task, designers were able to detect the strategies players would employ. As a result, several changes were made to the gameplay to encourage players to engage in much closer combat. See Pagulayan et al., (2003) for more details.

Prototyping, heuristic evaluations, and empirical guideline documents are other techniques we often use when more time-consuming testing cannot be done. In practice, these techniques do not differ when used on games. Nielsen (1993) categorizes prototyping and heuristic evaluations as “discount usability engineering,” and we would agree. We also tend to view empirical guideline documents in a similar manner. Empirical guideline documents are essentially lists of usability principles for particular content areas based on our collective experience doing user research. Examples of some of these content areas include console game shell design, PC game shell design, PC tutorial design principles, movement, aiming, and camera issues in first/third person shooter games, and online multiplayer interfaces. Desurvire, Caplan, and Toth (2004) have developed a list of heuristics targeted at computer and video games. These have been broken down into four general categories: game play, game story, game mechanics, and game usability. For the full list of gameplay heuristics, see Desurvire et al, (2004).

Survey Techniques

The use of surveys has been explored in great depth (e.g., Bradburn & Sudman, 1988; Couper, 2000; Labaw, 1981; Payne, 1979; Root & Draper, 1983; Sudman, Bradburn, & Schwarz, 1996) and is considered a valid approach for creating an attitudinal data set as long as you ask questions that users are truly capable of answering (see Root & Draper, 1983). We conduct surveys in the lab and online in order to collect attitudinal data regarding gameplay experiences, self-reported play behaviors, preferences, styles, and motivations.

Formative Playtest Surveys. We combine hands-on play with surveys to create a formative lab method called “playtest.” In nearly all instances, questioning follows the pattern of a forced choice question (sometimes a set of forced-choice questions) followed by a more open-ended question encouraging the participants to state the reasons behind their response. The goal of a playtest is to obtain specific information about how consumers perceive critical aspects of a game and provide actionable feedback to game designers (Davis, Steury, & Pagulayan, 2005; Amaya et al., 2008). The basic playtest method is used formatively to compare a product at time one and time two during its development. While a modified version of the basic method is used summatively to compare a summative evaluation to other, relevant games.

Playtests have several advantages and characteristics that differentiate them from usability studies and online survey research. For one, they focus mainly on perceptions. A larger sample size is required for statistical power and generalization of the results to the target audience. In most cases, 25-35 participants are used as a pragmatic trade-off between confidence and resource constraints. Specialized studies may require more. Statistical representations of the attitudinal measures are used to identify potential strengths and weaknesses that need to further be explored, provide information about the extent or severity of an attitude, or to describe trends of interest that come to light.

Conducting the studies in the lab as opposed to online, allow us to test in-progress games during early development. It also allows us to take more control of the testing situations and monitor a limited amount of play behaviors that can be paired with the attitudinal data.

On the other hand, there are a few notable disadvantages. First, structured lab studies limit the amount of gameplay that can be tested in a single study. An artificial lab scenario or timeline may prevent the engineer from collecting critical information about the game experience. This is a classic validity tradeoff present in most lab studies. Second, the amount of preparation that is required to design a playtest questionnaire takes some time. The study plan cannot be adjusted during the course of the study. The engineer must be intimately aware of the development team's questions to include all of the relevant questions in the survey. This requires domain knowledge in games and experience in survey design.

Second, the data can sometimes be difficult to interpret if the problem isn't major, or if the game facet is novel. Sometimes the data doesn't provide specific causes for the problem, making it difficult to nail down the best solution. Follow-up studies, (playtest or usability) are often recommended (or necessary) to parse out these issues. Finally, the approachable nature of the data is ripe for misinterpretation by team members who may not be familiar with the current state of the game, design details, or the methods used in playtest. It has been rightfully asserted that usability tests do not necessarily require a formal report upon completion (Nielsen, 1993). We believe that that may be less true for using this kind of technique. Careful and consistent presentation formats are required. Instructions and contextual information that will help with interpretation should be included, along with some form of qualitative data (e.g., open-ended comments). Metrics from games that are in development should focus on problem detection and often cannot be used for predictive purposes.

Development of the Playtest Surveys. Several steps have been taken to increase the effectiveness and efficiency of playtest. First, core questions for each genre, play style, and core game feature have been developed, using a multi-step iteration and evaluation process. This process included construct development, survey item development, formative evaluation with subject matter experts, and formative evaluation of question validity using Cognitive Interviewing techniques and statistical validation. While many survey items have been validated, most playtests requires new items to be created in order to meet the specific needs of that particular game and study goal. Therefore, a repository of previously used customized questions has been created, along with guidelines for writing new survey items.

Type of Formative Studies

There are essentially three types of formative studies that are conducted during development: (a) the critical facet test, (b) the initial experience test, and (c) the extended playtest. We also run tests that do not easily fit into any of these categories, including subtle variations on the above, a few cases of large sample observationally-based studies, and studies that focus on games from a more conceptual level.

As in the usability section, for clarity of presentation, each technique will be discussed separately, followed by a case study. Each case study will only contain information pertinent to a specific technique, thus examples may be taken from a larger playtest.

Critical Facet Playtest. Games often take the form of repeating a core experience within an array of different context and constraints. A driving game is always about aiming an object that is hurtling through space. The critical facet playtest focuses directly on that core experience and making sure that it is fun. While the core experience can often be assessed in usability testing, playtesting is necessary to assess attitudes and perceptions about the core experience.

The following example demonstrates how a critical facet test was applied to the critical facets of *Oddworld: Munch's Oddysee* (Microsoft Corporation, 2001), an Xbox game. *Munch's Oddysee* is a platform/adventure game that allows you to switch back and forth between two main characters as they proceed through the increasingly difficult dangers of Oddworld on a quest to save Munch's species from extinction. The core gameplay is exploring the realm by running, jumping, and swimming through the environment. In this case, there were concerns about the user's visual perspective – which we typically call the camera. Does the camera support exploration of the environment?

Case study: Munch's Oddysee, Camera. Previous usability testing with the *Munch's Oddysee* had determined that while some users indicated dissatisfaction with the behavior of the camera, other participants chose not to mention it at all while engaged in the open-ended usability tasks. The camera's behavior was programmed so as to create maximal cinematic effects (i.e., sometimes zooming out to show the size of an area) and also attempt to enhance gameplay. The camera would often show a specific view with the intent of showing you what was behind "the next door" or on the other side of the wall while still keeping the main character in view. While many users liked the look, style, and behavior of the camera, users often wanted more control over the behavior of the camera. Indeed some participants would actively say things such as, "That looks really cool right now [after the camera had done something visually interesting] but I want the camera back pointing this way now." Because feedback from the usability lab contained both positive and negative feedback, the development team didn't see the usability data as conclusive. Further, changing the camera would be a major cost to the game in terms of redesign and re-development time.

After having played the game for an hour, 25 participants were asked for general perceptions of the game. More specific questions followed. Questions related to the camera were asked in the latter portion of the questionnaire because previous experience in the usability lab had shown that merely mentioning the camera as part of a task would often cause participants previously silent on the subject to vociferously criticize aspects of the camera's behavior. With the knowledge that we wanted to factor out any priming-related effects, two analyses were conducted.

The first analysis was based on the response to the questions related to the behavior of the camera itself. Nearly half of the participants (46%) indicated that the camera did not give them "enough flexibility" of control.

The second analysis went back through individual responses to determine the attitudes of those participants who mentioned the camera before the survey first broached the subject. 43% of the participants were found to have mentioned the camera in a negative fashion prior to being asked specifically about the camera questions.

Based on this data and other anecdotal evidence, the development team chose to give the players more flexibility of camera control. The result was more frequent use of a camera behavior we termed a "3rd-person follow camera". The behavior of this camera had the double advantage of being more easily controlled by users and of conforming to a set of behaviors more often expected by users. It maintained focus on the main character without major adjustments to point of view (i.e, to "look over a wall" or "behind a door"). Other camera behaviors (i.e., still camera behaviors that pan with the character) are still a part of the game but have been localized to areas where these alternative camera behaviors can only create an advantage for the user.

Initial Experience Playtest. As with many things, first impressions are a key component of overall satisfaction with a game. Given that many games are a linear experience/narrative there is a lot of value in obtaining attitudinal data related to the first portions of gameplay. Lessons learned at first are often applied throughout the game. Obviously, the later portions of the game will never be experienced unless the first portions of the game are enjoyed.

The following example explains how a set of formative initial experience tests were run for *MechCommander 2*, the RTS game described earlier in the chapter. The earlier usability test focused on the choices that users could make prior to starting a mission, whereas this test focused on in-game components, such as user's ability to take control of their squad and lead them through battles, while experiencing satisfaction and motivation to continue playing.

Case study: MechCommander 2, Initial Missions. 25 participants, who were representative of the target market, were brought onsite and were asked to begin playing from the first mission of the game. After each mission, participants were asked to stop playing and report their impressions of the "fun", the "excitement", and the "clarity" of objectives. Participants also indicated their "comfort level" with the basics of controlling the game. The participants were able to play through as many as three missions before the session ended. For purposes of brevity, this case study will focus on the results related to the first mission.

Although the first mission had been designed to offer tutorial elements in a brief and fun way, the experience was generally not satisfying to participants. Approximately a third of the participants had a poor initial impression of the game. They felt that the "challenge" was "too easy" and that the mission was "not exciting". Furthermore there were some problems related to clarity of goals. By combining the survey results with opportunistic observation, it was noted that some participants moved their units into an area where they were not intended to go. This disrupted their experience and limited their ability to proceed quickly. Responses to more open-ended questions indicated that some of the core gameplay components and unique features did not come across to users. Some users complained about the "standard" (predictable or commonplace) nature of the game. Finally several participants complained about the fact that they were "being taught" everything and wanted to "turn the tutorial off".

A number of actions were selected from team insight and user testing recommendations. First, the team decided to separate the tutorial missions from the required course of the game – in order to save experienced players from the requirement of completing the tutorial. Second, the scope of the first mission was expanded so that users would have more time in their initial experience with the game. Third, the clarity of objectives was improved via minor interface changes. Fourth, addressing the same issue, the design of the "map" on which the mission took place was re-vamped to require a more linear approach to first mission that limited the likelihood of users becoming "lost." Fifth, the amount and challenge of combat was increased. Finally, one of the unique components of the game was introduced. The new mission included the ability to "call in support" from off-map facilities. Despite all these changes, the mission was still targeted to be completed within approximately 10-15 minutes.

A follow-up playtest was intended to verify the efficacy of the changes. 25 new participants were included in the study. Results from this test indicated that the design changes had created a number of pay-offs. Far fewer participants felt the mission was "too easy" (13% vs. 33%), only 3% indicated that the mission was "not exciting", measures of "clarity of objectives" improved and, surprisingly, there was no drop-off on ratings of "comfort" with basic controls as a result of the tutorial aspects being improved. Results were not a total success, as some participants were now rating the mission as "too hard" while others in response to open-ended questions complained about the "overly linear" nature of the mission (i.e., there was only one way to proceed and few interesting decision related to how to proceed through the mission). In response to these results the development team decided to "re-tune" the first mission to make it a little easier, but not to address the comments related to the linearity of the mission. The second mission of the game was far less linear and so it was hoped that, with the major constraints to enjoyment removed, most participants would proceed to the second mission and find the mission goals to be more engaging. The data from mission 2 was rated far more "fun" than mission 1, validating their assumption.

Extended Playtests. While the initial experience with a game is very important, the success or failure of a game often depends on the entire experience. In an attempt to provide game developers with user-centered feedback beyond the first hour of game play, we conduct extended playtests that test several hours of consecutive gameplay. This is done by conducting playtest studies that run for more than 2 hours, or by having participants participate in more than one study. Attitudinal data is taken at significant time or experience intervals (i.e. after a mission). Participants can also provide self-directed feedback about a specific point in the game at anytime during the session. Basic behaviors and progress (i.e., completed level 4 at 12:51pm) can be recorded by playtest moderators and used in conjunction with the attitudinal data.

Focus Groups. In addition to usability and survey techniques, we sometimes employ other user-centered design methods which are qualitative in nature. Focus groups provide an additional source of user-centered feedback for members of the project team. Our playtest facilities offer us the opportunity to precede a candid user-discussion with hands-on play or game demonstration. In this setting, participants can elaborate on their experience with the product, provide feedback about the user-testing process, speculate about the finished version of the product, generate imaginative feature or content ideas or find group consensus on issues in a way that models certain forms of real-world shared judgments.

For example, a focus group discussion regarding a sports game revealed that users had varying opinions about what it meant for a game to be a "simulation" versus an "arcade-style" game. Additionally, they had many different ideas as to what game features made up each style of game. From this conversation, we learned a few of the playtest survey questions were not reliable and needed revision.

Besides the general limitations inherent in all focus group studies (Greenbaum, 1988; Krueger, 1994), the focus groups described here typically include practical limitations that deviate from validated focus group methods. Generally, focus group studies comprise of a series of four to twelve focus groups (Greenbaum, 1988; Krueger, 1994), however, we typically run one to three focus groups per topic. This makes the focus group more of a generative research tool rather than a confident method for identifying "real" trends and topics in the target population. Fewer group interviews make the study more vulnerable to idiosyncrasies of individual groups.

Testing Physical Games

When testing with physically-oriented games such as for the Wii or drumming in Rock Band, the basic concept of how to test is very similar to what we have noted previously. In a usability scenario we will use a mixture of open-ended and directed tasks and in a playtest scenario we will still ask users to play and then fill out brief surveys about the experience. The key considerations mostly focus on how the details of a test design will interact with the physical nature of the test. These differences touch on every aspect of research design.

Participants. When considering who to include in the study some thought should be given to the pre-existing skill-level that you would prefer to see from your study participants. In the case of a singing game the level of skill a participant brings may be a subject of interest prior to the test. And for a physically-oriented game, testing with people of different physical capabilities may be a matter of some importance.

You have to ensure that the participant is ready for and physically capable of participating in the study. Prior to the study, the participants might be asked to show up wearing clothes and shoes they can comfortably move around in. Liability issues may require you to modify any pre-test agreements to cover potential physical issues that can occur.

A separate but related point is that the act of using a talent or physical capability in even a private, enclosed venue such as a research laboratory may touch on feelings of anxiety and an individual's discomfort with their own ability to perform an action. While considerations like this are a factor in most research, the fact that a person is actually performing the action or skill-based task removes an abstraction layer that is more commonly a part of testing with standard game controller-based games. For example, if a player struggles with a game that requires expert use of a game controller then the player has a few strategies to manage self-esteem and express frustration. She has the option of choosing from statements such as "I cannot do that" or "the game is too hard" or "I am no good with this controller." Meanwhile in the context of a physical or talent-based game, while the same options are still available the "I cannot…" and "I am no good…" statements can point out deficits that are more personal in nature, are uncomfortable to discuss, and perhaps not pertinent to the needs of the test. Standard participant management techniques are called for to reduce anxiety and discomfort.

Finally, some consideration should be given to the idea of recruiting participants in groups. While not always true it is very common for a physically oriented game to be oriented towards group play. The use of groups can complicate the assessment and that will be touched on in the method section below. But the use of groups also has the pleasant side effect of reducing some of the awkwardness of asking a user to perform a talent-based task and potentially failing in front of strangers.

Facilities. A key differentiator for physical-based gaming is that the facilities must support numerous scenarios. Sometimes the games are quite loud, in which case some sound-proofing may be a helpful factor in the test setup. At other times the key factor would be providing a sufficient amount of space for two or more people to comfortably play together, while swinging their arms and perhaps moving through the rooms. Additional considerations should go to safety concerns; ensuring the floor surfaces in the testing environment are not and cannot become especially slick and to either move furniture out of the way or to ensure it is adequately padded in case of a fall.

From the perspective of the testing, the facilities should have multiple cameras watching the play experience, preferably from a frontal view, a side view and a top-down view. All views should have a wide an angle as possible available so that the tester can capture a reasonable range of motion without having to adjust the cameras in real-time. Getting by without all the views is possible but the tester will miss details about the depth or actual size of a motion if only some of the fields of view are available.

Test Methods. For physical gaming, the primary adjustment to the methods occurs in usability testing. Put most simply, the think aloud technique may not be appropriate when the participant may be out of breath. It is more advisable to just concentrate on logging behaviors while a participant is playing and then to follow up with a more interview-oriented approach to understand what they were thinking at various points when they were playing.

Since the testing is of a physical experience, the researcher must divide attention between the game status as presented on screen to the participant and on the participant himself as he moves while playing. This only represents a change in that in physical scenarios there are numerous movements of interest, requiring multiple cameras to detect and study them.

From our experience, groups of participants are commonly treated as a 'single participant set'. So in the standard discussion of how many participants you want for a study (Nielsen & Landauer, 1993) the number of individuals must be replaced with the number of groups. The plusses and minuses of using groups as the minimal measure of interest are best detailed within a context devoted to the subject. In our testing, to avoid problems associated with groupthink (Janis, 1972) all participants are instructed to be open and honest and are informed that we are not trying to reach a consensus and that it is okay to disagree. These instructions are based on the standard approach to conducting a focus group (Greenbaum, 1988; Krueger, 1994). When coding the usability test, the behavior of the group and its success, failure and/or time on task is treated in similar fashion to coding the behavior of an individual. So if an 8-year old child is on the cusp of figuring out an interface element but his mother insists that he is on the wrong track, then the nature of the failure is treated similarly to an individual going through the same process. The interface element is still judged to have failed the task, though the initial attractiveness is noted as well as the reason for group attention moving elsewhere.

Data Analysis and Reporting. The analysis of physically-based gaming data is not especially different from the typical. Data can be coded and communicated similarly to data from other kinds of games. In the reporting, however there will often be a much stronger emphasis on visual analysis and accurately describing a physical motion. Time spent poring over video will likely increase and efficiencies can and must be sought from the use of strong video and data-coding software. Additionally the use of high-powered video editing software will often be required to create strong clear presentations of results.

TRUE Instrumentation

For several years, we have been utilizing an instrumentation technique we call Tracking Real-Time User Experience (TRUE). TRUE instrumentation involves having the game track key user behaviors during gameplay, along with contextual variables, and allowing users to answer survey questions at key points in time. The TRUE method provides broad coverage, detailed recommendations, and quick turnaround of data collection, analysis, and recommendations. Its unique combination of quantitative analysis of behavior data, collection and analysis of evaluative data (ratings and preferences), and ability to link all of these to qualitative data (video) make it a very powerful tool in a user researcher's arsenal. It combines the best of problem detection with in-depth analysis; the in-depth analysis makes the data actionable by design teams. The advantages of this combination deserve some discussion, and useful and practical recommendations for applying TRUE are discussed elsewhere (Kim et al., 2008; Schuh et al., 2008).

Behavior, Evaluation, and Context. To fully understand how effectively a game meets the intent of the designers one needs to know: what people are doing at any point in the game (behavior), what conclusions they are drawing about the game (evaluations), and what in the mechanics of the game is leading to these behavioral and evaluative results (context). By collecting both behavior and evaluation and linking those results to indexed video, the development team can quickly identify (detect) problem areas and then zoom down to video to determine what elements of the game are creating those problems. Put another way, the TRUE method captures the primary elements needed to empirically understand and change the mechanics of a game in order to produce improved dynamics (behavior) and aesthetics (evaluation). Other methods (e.g., usability testing) may capture all these elements but they often do so in an informal way (thinking out loud and watching participants). In addition, the labor intensive nature of these data collection and analysis techniques means that, from a practical standpoint, they cannot cover the entirety of a sizable game. This increase in efficiency provided by TRUE does not just make work easier, it opens new possibilities. The scope of the application of TRUE combined with its breadth and depth make it unique.

Issue Discovery. Like other user research methods TRUE can be thought of as a "discovery" method. It collects and plots data so that anomalies and patterns can be spotted. Once an anomaly is spotted, e.g. more deaths than expected at a given place or time, or users not progressing past a given point in a mission or on a map, one can look at more and more detailed elements of the data until the likely cause for the unexpected outcome is discerned. This often involves a progressive drill down into the data which may lead ultimately to viewing the video showing exactly what players did and in what context they did it. It shows what the team as a whole could not have predicted or at least did not anticipate. This "detection" quality is an essential characteristic of many user research methods (traditional usability test, RITE tests, Heuristic reviews, and others). TRUE is not really intended as a method to compare designs. Other methods like playtest and benchmark testing are better suited to those tasks.

Like other methods TRUE is best applied in some contexts and not at all applicable in some others. However the conditions for success for TRUE are more subtle than those of other methods. Specifically, TRUE is best applied when the development effort is characterized by common framework and shared definitions. For example, TRUE works well when missions are defined in a similar way throughout the game. Specifically, a mission begins when a prior climatic battle is won (e.g., the previous boss is vanquished) and ends when a final climatic battle is won (e.g., the current boss is killed). It also is well applied to games where the player navigates a map facing puzzles to solve and enemies to vanquish or in racing games where there is a clear start and finish of a gameplay segment. Its application is difficult if not impossible to a game where missions are defined in an inconsistent way, e.g. sometimes a mission ends when a boss is vanquished and sometimes it ends when a boss is encountered. This kind of consistency needs to exist in all parts of the game. If the player relationship between a damage and weapon use is not lawful and uniform then analyzing the game with automated tools becomes difficult or impossible. The analyst is confronted with anomalies that defy explanation or are explained by lapses in consistency. Needless to say such haphazard elements will also make the game difficult to play and not in the good sense of that term. That is not difficult because the game is challenging but because it is arbitrary. That said TRUE can detect bugs in a game – an accidently overly powerful enemy for example. Those kinds of bugs will stand out as anomalies just like a level or puzzle that is too challenging.

To be applied, the TRUE method must have a working prototype. Thus, it cannot be applied to an effort that is at the stage of ideation or story boards. It could be applied to any portion of a game that is playable. It could also be applied to a prior version of a game to identify areas for improvement. It can't be applied to other elements that contribute to the success of a game, such as story line or character depth. However, while these elements may play significant roles in a games success, the game play is what defines a game and differentiates it from other art forms like books or movies.

In summary, TRUE is a powerful and flexible method, but requires a great amount of investment. When done correctly, its application has resulted in hundreds of improvements to dozens of games. It has greatly extended the "reach" of empirical methods into game design and development. It stands as a proven, practical, and effective method that has contributed to the development and commercial and critical success of many games (Thompson, 2007). It represents a methodological breakthrough in games research and stands as a yardstick against which new methods can and should be assessed.

A Combined Approach

There are a variety of user research techniques we use in game development and evaluation – each suited for answering a particular type of question the development team has. As can be seen in this chapter, there are clear differences between the techniques but the end goal is the same: improve the user experience in our games. The important thing to realize is that no one method exists independent of other methods. The following is a brief case study on a PC game called *Rise of Nations: Rise of Legends* (Microsoft Game Studios, 2006) that demonstrates how some of these techniques can be used in tandem.

<u>Case Study: Rise of Nations: Rise of Legends</u>. *Rise of Nations: Rise of Legends* is a Real-Time Strategy (RTS) game in which the player must acquire resources, build up their armies and cities, and defeat/overtake any enemies in an effort to take control of the map. Our initial concern was the core controls – users must be able to successfully gather resources, successfully build units and buildings, and successfully engage in combat. A secondary concern was related to the Conquer the World Campaign and the mission progression throughout it.

The first issues addressed were the core controls and gameplay. We wanted to know if users could successfully build, resource, and engage in combat, so we decided to run a RITE usability test on the core gameplay and controls. This RITE testing was only possible because we had the development team from Big Huge Games on site with the ability to make quick iterative changes to the game between participants.

For the initial building controls, in order to place a building, the user had to determine where they wanted it, and then *right-click* the mouse button to place the building on the map in that location. During our initial RITE study, we observed several participants attempting to place buildings using a *left-click*. Moreover, they continued to make this error even after they discovered how to properly execute this action (right-click). We attempted to fix this issue by adding a visual prompt when placing buildings (i.e., a visual icon of a mouse with the right button highlighted green) and tested this fix with four subsequent participants. All four understood the prompt; however, all continued to sporadically make left-click errors. We decided then to modify the controls to eliminate this usability issue and the game was changed to allow either left *or* right-click building placement. No user following the change had difficulty placing buildings.

In addition to building, we were also interested in finding out if users could successfully resource. We noticed early on in the study that some participants did not explore the map at all. This type of behavior has repercussions for resourcing in that in order to find resources, one must explore the map. The designer intent for the game was that users would explore the map early on in order to find resources and plan strategies. It became clear that one of the barriers to exploration was that users needed to create scout units to help them explore. Yet, in order to create a scout unit, one had to first build a barracks. Both the time and sequence required for these events meant that users did not always create scout units and if they did, it was usually after a bit of time playing. The team decided to give users a “free scout unit” at the start of the game. The scout unit was placed conspicuously right in front of the player’s home city. As a result, fourteen of the next fifteen users explored the map and resourcing performance was greatly improved.

During this RITE study, we also discovered some small usability issues related to combat. These issues were addressed during the course of the study, changes were made to the game, and these changes were validated. Thus, by the end of the study, we were able to improve the usability of the core controls and gameplay and have those improvements implemented into the game.

After this initial RITE study, we began concentrating on the single-player Conquer the World Campaign. The Conquer the World Campaign involves the player progressing through forty-seven separate missions “scenarios” with the end goal of conquering the world map. The team wanted to apply the RITE techniques to each scenario in order to get fine-grained user data and determine if there were any problematic areas across the campaign. The design questions we wanted answered were: Where are users getting blocked? Where are users not having fun?; Is the challenge ramp appropriate?; Do users think a particular scenario is too easy or too difficult?; What do users like/dislike about the campaign experience after several hours?

Some of these design questions we could not answer via usability techniques alone and RITE testing forty-seven missions was both time and resource prohibitive. Thus, we combined several research methodologies to help answer the design questions. We decided to get broad coverage of the game, follow it up with a fine-grained analysis of the problem areas, fix the issues, go back and cast the broad net, and follow that up by more focused fine-grained testing. To begin, we started running extended playtests over the weekends with TRUE instrumentation. The TRUE instrumentation gave us a mapping of user behavior over time as well as attitudinal data to couple with it. The Monday following the weekend test, we utilized the TRUE data to determine problematic areas to focus our RITE testing on during the week. During the week, we would RITE test those scenarios, all along fixing the builds, and the following weekend, run extended playtests with TRUE instrumentation on the updated game. Over the course of several weeks, we were able to get complete coverage of the entire Conquer the World Campaign and answer most of the design questions we set out to answer. By combining RITE testing, which allowed us to discover and fix many instances of user behavior not matching designer intent, and TRUE instrumentation, which allowed us to discover patterns in user behavior over the course of extended gameplay as well as giving us rich qualitative information and attitudinal data, we were able to help better match the user experience to the design vision across the entire game.

Conclusion

The need for the continued development of user-centered design methods in video games has indeed arrived. Games drive new technologies, generate enormous revenue, and affect millions of people. In addition, games represent a rich space for research areas involving technology, communication, attention, perceptual-motor skills, social behaviors, virtual environments, just to name a few. It is our position that video games will eventually develop an intellectual and critical discipline, like films, which would result in continually evolving theories and methodologies of game design. The result will be an increasing influence on interface design and evaluation. This relationship between theories of game design and traditional human-computer interaction evaluation methods has yet to be defined, but definitely yields an exciting future.

User-centered design methods are beginning to find their way into the video games industry and commercial game companies, such as Ubisoft Entertainment, Electronic Arts, Inc., in addition to Microsoft, employ some level of user-centered methodologies to their game development process. Games share as many similarities as differences to other computer fields that have already benefited from current user-centered design methods. Thus, it makes sense to utilize these methods when applicable but also to adapt methods to the unique requirements that we have identified in video games.

In this chapter, we emphasized the difference between games and productivity applications in order to illustrate the similarities and differences between these two types of software applications. We also chose to reference many different video games in hopes that these examples would resonate with a number of different readers. Case studies were included to demonstrate, in practice, how we tackle some of the issues and challenges mentioned earlier in the chapter. It is our intention that practitioners in industry, as well as researchers in academia, should be able to take portions of this chapter and adapt them to their particular needs when appropriate, similar to what we have done in creating the actual methods mentioned in this chapter. That said, we are upfront that most, if not all of our user research methods are not completely novel. Our user research methods have been structured and refined based on a combination of our applied industry experience, backgrounds in experimental research, and of course, a passion for video games. This allows us to elicit and utilize the types of information needed for one simple goal: to make the best video games that we can, for as many people as possible.

References

Age of Empires II: Age of Kings (1999). [Computer Software]. Redmond, WA: Microsoft Coporation.

Alpine Racer (1995). [Computer Software]. Tokyo, Japan: Namco.

Amaya, G., Davis, J., Gunn, D., Harrison, C., Pagulayan, R., Phillips, B., & Wixon, D. (2008). Games User Research (GUR): Our Experience with and Evolution of Four Methods. In K. Isbister, & N. Schaffer, *Game Usability: Advice from the experts for advancing the player experience* (pp. 35-64). Burlington, MA: Morgan Kaufmann Publishers.

Animal Crossing (2002). [Computer Software]. Redmond, Washington: Nintendo of America.

Banjo Kazooie (2000). [Computer Software]. Redmond, WA: Nintendo of America.

Bradburn, N. M., & Sudman, S. (1988). *Polls and surveys: Understanding what they tell us.* San Francisco: Jossey-Bass.

Brute Force (2003). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Business Week (2006, October 18). *A Brief History of Game Console Warfare.* Retrieved March 22, 2008, from http://images.businessweek.com/ss/06/10/game_consoles/source/7.htm

Cassell, J., & Jenkins, H. (2000). Chess for girls? Feminism and computer games. In J. Cassell, & H. Jenkins, *From Barbie to Mortal Kombat: Gender and computer games* (pp. 2-45). Cambridge, MA: The MIT Press.

Chalk, A. (2007). Halo 3 Sets New First-Day Sales Record. Retrieved from: http://www.escapistmagazine.com/news/view/77341-Halo-3-Sets-New-First-Day-Sales-Record.

Combat Flight Simulator (1998). [Computer Software]. Redmond, WA: Microsoft Corporation.

Couper, M. P. (2000). Web surveys: A review of issues and approaches. *Public Opinion Quarterly , 64*, 464-494.

Crackdown 2 (2010). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Crash Bandicoot: The Wrath of Cortex (2001). [Computer Software]. Los Angeles, CA: Universal Interactive Studios.

Crawford, C. (1982). *The art of computer game design.* Berkeley, CA: Osborne/McGraw-Hill.

Csikszentmihalyi, M. (1990). *Flow - The psychology of optimal experience.* New York, NY: Harper & Rowe.

Dance Dance Revolution (1998). [Computer Software]. Tokyo, Japan: Konami.

Davis, J. P., Steury, K., & Pagulayan, R. J. (2005). A survey method for assessing perceptions of a game: The consumer playtest in game design. *Game Studies: The international journal of computer game research , 5* (1).

Desurvire, H., Caplan, M., & Toth, J. (2004). Using heuristics to improve the playability of games. *CHI 2004: Conference on human factors in computing systems.* Vienna, Austria: ACM's Special Interest Group on Computer-Human Interaction.

Diddy Kong Racing (1997). [Computer Software]. Redmond, WA: Nintendo of America.

Domjan, M. (2010). *The Principles of Learning and Behavior.* Belmont, CA: Wadsworth.

Dumas, J. S., & Redish, J. C. (1999). *A practical guide to usability testing (Rev. ed.).* Portland, OR: Intellect Books.

Entertainment Software Association (ESA) (2009). *2009 Sales, Demographics and Usage Data: Essential Facts About the Computer and Video Game Industry.* Washington, DC: Entertainment Software Association.

Facebook (2010). *Facebook's Farmville Application Page.* Retrieved April 25, 2010, from Facebook: http://www.facebook.com/apps/application.php?id=102452128776

Final Fantasy X (2001). [Computer Software]. Tokyo, Japan: Square Co., Ltd.

Flight Simulator X (2006). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Forza Motorsport 3 (2009). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Game Boy (1989). [Computer Hardware]. Redmond, WA: Nintendo of America.

Game Boy Color (1998). [Computer Hardware]. Redmond, WA: Nintendo of America.

Gameboy Advance (2001). [Computer Hardware]. Redmond, WA: Nintendo of America.

Gardner, J. (2009, September 29). *Futurology: FarmVille on Facebook.* Retrieved October 11, 2009, from London Today: http://www.thisislondon.co.uk/lifestyle/article-23749479-futurology-farmville-on-facebook.do.

Gears of War (2006). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Greenbaum, T. L. (1988). *The practical handbook and guide to focus group research.* Lexington, MA: D.C. Heath and Company.

Guitar Hero (2005). [Computer Software]. Santa Monica, CA: Activision.

Halo 2 (2004). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Halo: Combat Evolved (2001). [Computer Software]. Redmond, WA: Microsoft Corporation.

Harvest Moon (2003). [Computer Software]. Redmond, WA: Nintendo of America.

Hyman, P. (2005). State of the industry: Mobile games. *Game Developer Magazine , 12* (14), 11-16.

IDG Entertainment (2005). *IDG Entertainment Casual Games Market Report.* Oakland, CA: IDG Entertainment.

International Game Developers Association (2005). *IDGA 2005 Mobile Games Whitepaper.* Presented at the Game Developers Conference 2005 by the Mobile Games SIG, San Francisco.

Internet Backgammon, Windows XP (2001). [Computer Software]. Redmond, WA: Microsoft Corporation.

Jablonsky, S., & DeVries, D. (1972). Operant conditioning principles extrapolated to the theory of management. *Organizational Behavior and Human Performance, 7,* 340-358.

Janis, I. L. (1972). *Victims of Groupthink.* Boston: Houghton Mifflin Company.

Jordan, P. W. (2000). *Designing pleasurable products: An introduction to the new human factors.* Philadelphia: Taylor & Francis.

Kent, S. L. (2000). *The first quarter: A 25-year history of video games.* Bothell, WA: BWD Press.

Kim, J., Gunn, D., Schuh, E., Phillips, B., Pagulayan, R., & Wixon, D. (2008). Tracking Real-Time User Experience (TRUE): A comprehensive instrumentation solution for complex systems. *Proceedings of the SIGCHI conference on Human factors in computing systems.* Florence, Italy.

Krueger, R. A. (1994). *Focus groups: A practical guide for applied research.* Thousand Oaks, CA: Sage.

Labaw, P. (1981). *Advanced questionnaire design.* Cambridge, MA: Abt Books Inc.

Lepper, M. R., & Malone, T. W. (1987). Intrinsic motivation and instructional effectiveness in computer-based education. In R. E. Snow, & M. J. Farr, *Aptitude, Learning and Instruction III: Conative and Affective Process Analyses.* Hillsdale, NJ: Erlbaum.

Lepper, M., Greene, D., & Nisbett, R. (1973). Undermining children's intrinsic interest with extrinsic rewards. *Journal of Personality and Social Psychology , 28*, 129-137.

Levine, K. (2001). New opportunities for Storytelling. *Paper presented at the Electronic Entertainment Exposition.* Los Angeles, CA.

Madden NFL 06 (2005). [Computer Software]. Redwood City, CA: Electronic Arts Inc.

Malone, T. W. (1981). Towards a Theory of Intrinsic Motivation. *Cognitive Science , 4*, 333-369.

Maximejohnson (2010). *Tetris atteint les 100 millions de téléchargements payants (et une petite histoire du jeu).* Retrieved May 24, 2010, from maximejohnson: http://www.maximejohnson.com/techno/2010/01/tetris-atteint-les-100-millions-de-telechargements-payants-et-une-petite-histoire-du-jeu/

Mazur, J. (2005). *Learning and Behavior (6th ed).* Upper Saddle River, NJ: Prentice Hall.

MechCommander 2 (2001). [Computer Software]. Redmond, WA: Microsoft Corporation.

MechWarrior 4: Vengeance (2000). [Computer Software]. Redmond, WA: Microsoft Corporation.

Medlock, M. C., Wixon, D., McGee, M., & Welsh, D. (2005). The Rapid Iterative Test and Evaluation Method: Better Products in Less Time. In G. Bias, & D. Mayhew, *Cost Justifying Usability* (pp. 489-517). San Francisco, CA: Morgan Kaufmann.

Medlock, M. C., Wixon, D., Terrano, M., & Romero, R. (2002). Using the RITE Method to improve products: a definition and a case study. *Usability Professionals Association.* Orlando, FL.

Microsoft (2007). *Global Entertainment Phenomenon "Halo 3" Records More Than $300 Million in First-Week Sales Worldwide*. Retrieved from Microsoft Newscenter: http://www.microsoft.com/presspass/press/2007/oct07/10-04Halo3FirstWeekPR.mspx

Moore, K., & Rutter, J. (2004). Understanding consumers' understanding of mobile entertainment. In Moore, & Rutter, *Proceedings of Mobile Entertainment: User-centred Perspectives, CRIC* (pp. 49-65). University of Manchester.

MPAA (2009). *Theatrical market statistics.* Washington, DC: MPAA.

MTV Music Generator (1999). [Computer Software]. Warwickshire, United Kingdom: The Codemasters Software Company Limited.

NFL2K (1999). [Computer Software]. San Francisco, CA: Sega of America, Inc.

Nielsen, J. (1993). Usability engineering. San Francisco, CA: Morgan Kaufmann.

Nielsen, J., & Landauer, T. K. (1993). A mathematical model of the finding of usability problems. *Proceedings of ACM INTERCHI'93 Conference*, (pp. 206-213). Amsterdam.

Nielsenwire (2009). Retrieved June 2009, from iPhone Users Watch More Video… and are Older than You Think: http://blog.nielsen.com/nielsenwire/online_mobile/iphone-users-watch-more-video-and-are-older-than-you-think/

Nintendo (2005). *2005 Annual Report.* Retrieved March 22, 2008, from http://www.nintendo.com/corp/report/NintendoAnnualReport2005.pdf.

Nintendo DS (2004). [Computer Hardware]. Redmond, WA: Nintendo of America.

Nintendo Wii (2006). [Computer Hardware]. Redmond, WA: Nintendo of America.

Norman, D. (2005). *Emotional Design.* Cambridge, MA: Basic Books.

Oddworld: Munch's Odyssey (2001). [Computer Software]. Redmond, WA: Microsoft Corporation.

Pagulayan, R. J., & Steury, K. (2004). Beyond usability in games. *Interactions , 11* (5), 70-71.

Pagulayan, R. J., Gunn, D. V., & Romero, R. L. (2006). A gameplay-centered design framework for human factors in games. In W. Karwowski, *2nd Edition of International Encyclopedia of Ergonomics and Human Factors.* London, England: Taylor & Francis.

Pagulayan, R. J., Steury, K., Fulton, B., & Romero, R. L. (2003). Designing for fun: User-Testing case studies. In M. Blythe, A. Monk, K. Overbeeke, & P. Wright, *Funology: From Usability to Enjoyment.* (pp. 137-150). Netherlands: Kluwer Academic.

Payne, S. L. (1979). *The art of asking questions.* Princeton, NJ: Princeton University.

Playstation Move (2010). [Computer Hardware]. Tokyo, Japan: Sony Computer Entertainment.

Pokemon Crystal (2000). [Computer Software]. Tokyo, Japan: Nintendo Japan.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., & Carey, T. (1996). Human-computer interaction. Reading, MA: Addison-Wesley.

Project Natal (2010). [Computer Hardware]. Redmond, WA: Microsoft.

RalliSport Challenge (2003). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Ratchet & Clank Future: A Crack in Time (2009). [Computer Software]. Tokyo, Japan: Sony Computer Entertainment.

Resident Evil 4 (2005). [Computer Software]. Osaka , Japan: Capcom.

Rise of Nations: Rise of Legends (2006). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Rock Band (2007). [Computer Software]. Redwood City, CA: Electronic Arts.

Root, R. W., & Draper, S. (1983). Questionnaires as a software evaluation tool. *Paper presented at the ACM CHI, Boston, MA.*

Saints Row 2 (2008). [Computer Software]. Agoura Hills, CA: THQ.

Salas, E., & Maurino, D. (2010). *Human Factors in Aviation (2nd ed.).* Burlington, MA: Elsevier.

Schuh, E., Gunn, D., Phillips, B., Pagulayan, R., Kim, J., & Wixon, D. (2008). TRUE Instrumentation: Tracking Real-Time User Experience in Games. In K. Isbister, & N. Schaffer, *Game Usability: Advice from the experts for advancing the player experience* (pp. 237-265). Burlington, MA: Morgan Kaufmann Publishers.

Second Life (2003). [Computer Software]. San Francisco, CA: Linden Research, Inc.

Shippy, D., & Phipps, M. (2009). *The race for a new game machine.* New York: Kensington Publishing Corp.

Sidel, P. H., & Mayhew, G. E. (2003). The Emergence of Context: A Survey of MobileNet User Behaviour. Retrieved from: www.mocobe.com/pdf/EmergenceofContext1.pdf .

Singstar (2004). [Computer Software]. Tokyo, Japan: Sony Computer Entertainment.

Sudman, S., Bradburn, N. M., & Schwarz, N. (1996). *Thinking about answers: The application of cognitive processes to survey methodology.* San Francisco, CA: Jossey-Bass.

Super Mario Bros Wii (2009). [Computer Software]. Redmond, WA: Nintendo of America.

The Legend of Zelda: The Wind Waker (2003). [Computer Software]. Redmond, WA: Nintendo of America.

Thompson, C. (2007). Halo 3: How Microsoft Labs Invented a New Science of Play. *Wired , 15,* 140-147.

Tom Clancy's Splinter Cell: Conviction (2010). [Computer Software]. Paris, France: Ubisoft.

Tomb Raider: Underworld (2008). [Computer Software]. London, England: Eidos.

Toy Soldiers (2010). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Toy Soldiers: Match Defense (2010). [Computer Software]. Redmond, WA: Microsoft Game Studios.

Wickens, C. D., & Hollands, J. G. (2000). *Engineering Psychology and Human Performance (3rd ed.).* Upper Saddle River, NJ: Prentice Hall.

Wii Sports Resort (2008). [Computer Software]. Redmond, WA: Nintendo of America.

Yuen, M. (2005, September). The Tipping Point: The convergence of wireless and console/PC game design. *Game Developer Magazine* .

Table 1.
The NPD Group super genre classification scheme for games.

Category	Description
Action	Control a character and achieve one or more objectives with that character.
Fighting	Defeat an opponent in virtual physical hand-to-hand or short range combat combat.
Racing	Complete a course before others do and/or accumulate more points than others while completing a course.
Shooter	goal is to defeate enemies in combat with ranged weapons (1st person shooters, 3rd person shooters).
Strategy	Strategically manage resources to develop and control a complex system in order to defeat an opponent or achieve a goal. This the goal is to defeat opponent(s) using a large and sophisticated array of elements.
Role-Playing	Control a character that is assuming a particular role in order to achieve a goal or mission. Rich story and character development are common.
Family entertainment	The primary objective is to interact with others and/or to solve problems. This genre includes puzzles and parlor games.
Children's entertainment	Same as family entertainment but geared to a younger audience.
Sports	Manage a team or control players, to either win a game or develop a team to championship status. Involves individual, team, and extreme sports,
Adventure	Control a character to complete a series of puzzles that are required to achieve a final goal.
Arcade	Games on coin-op arcade machines, or games that have similar qualities to classic arcade games. Generally, they are fast paced, action games.
Flight	Plan flights and pilot an aircraft in a realistic, simulated environment
All other games	Educational, compilations, non flight simulators, rhythm games, etc.

Table 2.
Age of Empires example concepts and behaviors categorized into three concepts using the RITE method (Medlock, Wixon, Terrano, & Romero, 2002).

Essential Concepts/Behaviors	Important Concepts/Behaviors	Concepts/Behaviors of Lesser Interest
• movement	• queuing up units	• using hotkeys
• multi-selection of units	• setting gathering points	• using mini-map modes
• "fog of war"	• garrisoning units	• using trading
• scrolling main screen via mouse	• upgrading units through technology	• understanding sound effects

Figure Captions

Figure 1. This is a sample of different types of input devices used in games; from the traditional keyboard and mouse, to different console gamepads.

Figure 2. Screenshot of the ‘Mech Lab in MechCommander 2.

Figure 3. A record of failures and errors over time for the Age of Empires Tutorial when using the RITE method (Medlock, et al., 2002).

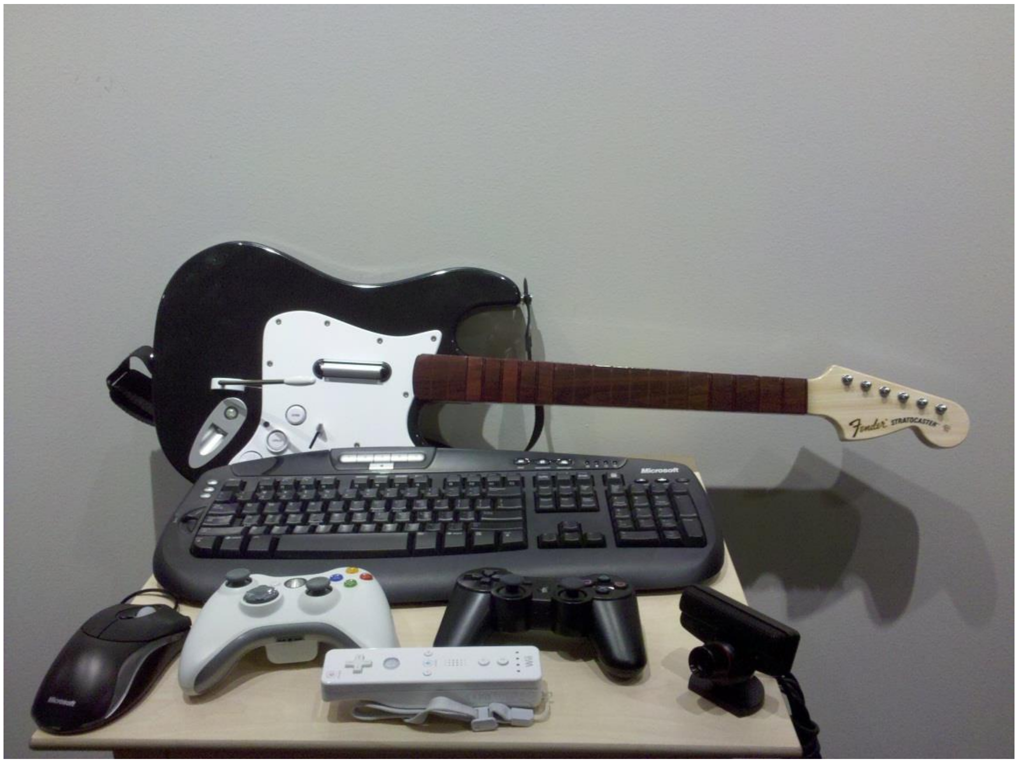

Figure 1.

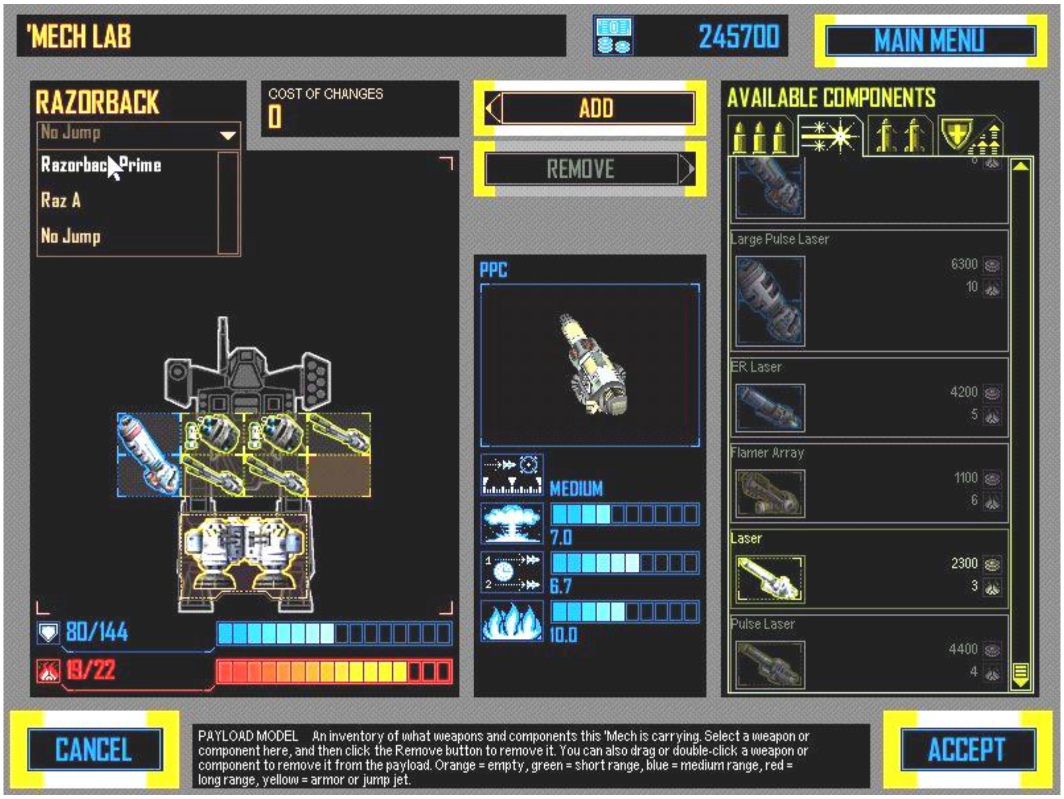

Figure 2.

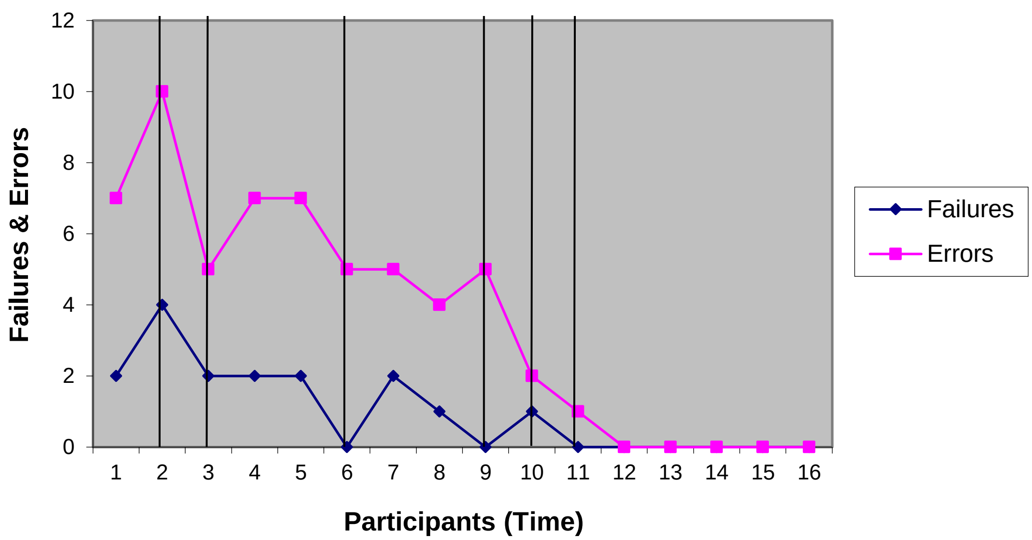

Figure 3.

Footnotes

[1]The *fog of war* refers to the black covering on a mini-map or radar that has not been explored yet by the gamer. The fog of war "lifts" once that area has been explored. Use of the fog of war is most commonly seen in RTS games.

Acknowledgments

We would like to thank the Games User Research Group at Microsoft Game Studios. In addition, we would like to express our gratitude to Paolo Malabuyo, Michael Medlock, and Bill Fulton for their insights and input on the creation of this chapter, and Kathleen Farrell and Rally Pagulayan for their editing assistance and reviewing early drafts. The views and opinions contained in this chapter are those of the authors and do not necessarily represent any official views of Microsoft Corporation.