RANDY J. PAGULAYAN, KEITH R. STEURY, BILL FULTON, AND RAMON L. ROMERO

CHAPTER 11
DESIGNING FOR FUN: USER-TESTING CASE STUDIES

1. INTRODUCTION

The goal of this chapter is to demonstrate that extending current usability methods and applying good research design based on psychological methods can result in improved entertainment experiences. This chapter will present several case studies where user-centered design methods were implemented on PC and Xbox games at Microsoft Game Studios. The examples were taken from several series of larger studies on Combat Flight Simulator (PC), MechWarrior 4: Vengeance (PC), Halo: Combat Evolved (Xbox), and RalliSport Challenge (Xbox). These examples were chosen to illustrate a variety of user-centered methods and to demonstrate the impact user-centered design principles can have on an entertainment product. Furthermore, these examples are presented in a way that illustrates a progression from addressing usability issues similar to those found in productivity applications, to extending usability methods to address more unique aspects of game design, to using survey methods to address issues related to fun for which standard usability methods do not suffice. For more detailed descriptions of Microsoft Game Studios User-testing methods and laboratory facilities, see Pagulayan, Keeker, Wixon, Romero, and Fuller (2003).

1.1 Methods and Games

The many similarities between productivity applications and games suggest that traditional discount usability methods would be suitable in the entertainment domain. Games have selection screens and menus just like other software applications. Task persistence, performance, ease of use, learnability, and all the potential obstacles to efficiency and productivity are found in games as well. However, it is possible to conceptualize usability in games as including other areas of game design, such as the comprehension of rules and objectives, control of characters, and manipulation of camera (view), to name a few. A game designer must script an experience within a game, so an extension of usability techniques from productivity applications to games becomes clearer. Usability testing in games

Mark A. Blythe, Andrew F. Monk, Kees Overbeek and Peter C. Wright (eds.), Funology: From Usability to Enjoyment, 137—150
(and productivity applications for that matter) can be viewed as an experiment designed to assess whether users will interact with a given product in the way the designer intended.

Video games also differ from productivity applications in a variety of ways. For example, productivity applications represent tools as a means to an end, whereas games are designed to be pleasurable for the duration of gameplay. This distinction means that we must ensure that the user has an appropriate level of challenge and engagement while playing a game, rather than focusing on how efficiently they can achieve their goals. Goals are often defined externally in productivity applications, whereas games define their own goals. This implies that goals and objectives in games must not only be clear to the user at all times, but that they must be interesting as well. Another difference between games and productivity applications lies in the number of choices available. While there are relatively few productivity applications designed for a specific purpose, there are many games. Games must compete with other game titles for consumer attention, but they must also compete with different forms of entertainment, such as watching television or reading a book (Pagulayan et al., 2003). The implication is that games that are not immediately enjoyable will be dismissed quite easily in favor of other forms of entertainment.

It is our intent to show that the HCI field has access to methods drawn from current usability methodologies and experimental psychology that can be adapted to address issues in the entertainment industry. Extending current usability methods and utilizing good research design can result in improved entertainment experiences.

2. CASE STUDIES

Below are four examples of user-centered design methods that were used on video and computer games at Microsoft Game Studios. The first case study begins with more traditional usability methods. Subsequent case studies extend the use of current methodologies to situations that demonstrate how one can address some of the unique issues encountered with games.

2.1 Combat Flight Simulator

This case study is an example of a usability issue identified using limited-interactive prototypes. This method is not substantially different from standard usability methods. The focus of the usability test in this case study was on the game shell screens for the PC game, Combat Flight Simulator (CFS). Game shell screens consist of all menus and screens (e.g., Main Menu, Options, etc.) which are used to set the particular desired gaming experience, but not encountered during actual gameplay. For example, in the game shell a user can often change the difficulty level of the game and set other gameplay parameters.

A usability test was performed early in the development cycle of CFS and used a limited-interactive prototype of the game shell screens (i.e., functional screen widgets including radio buttons, check boxes, and toggles). Techniques utilized included user and task observation (Nielsen, 1993). Some of the tasks were performance based. In general, the shell screens while thinking aloud.

In CFS, parameters users can select include location, time of day, action, enemies. During the exploratory task, participants were asked to detect and identify threats. Commonly used in the video game industry, commonly used in the video game industry, commonly used in the video game industry, commonly used in the video game industry, commonly used in the video game industry.

For example, as “AI” as well as the marines that the(actual name) sits or marines are referred to as “AI” referred to the skill level of controls. Participants in several different with the screen presented to participants.

Figure 1. Portion of Op

The term “AI Level” was not used. The term was seven understood the intended representation for AI Level (Low, Med, High). The main problem was with the term developers, it is unfamiliar to many had experience with computer games with flight games. A poor choice of
 CHAPTER 11

included user and task observations, scenarios, and thinking aloud protocols (e.g., Nielsen, 1993). Some of the tasks presented to users were exploratory and not performance based. In general, the users’ tasks were to explore a particular game shell screens while thinking aloud.

In CFS, parameters users can modify using the game shell include aircraft selection, start location, time of day, weather, number of enemies, and difficulty of enemies. During the exploratory task for the Options screen, a usability problem was detected related to terminology. “AI” is an acronym for artificial intelligence commonly used in the video and computer games industry to refer to the behavior of non-human entities. For example, the cars players drive against could be referred to as “AI” as well as the marines that fight along side of players, providing that in both situations the cars or marines are not actively controlled by a human player. In CFS, “AI” referred to the skill level of computer-controlled enemy pilots, which confused participants in several different ways. Figure 1 represents a portion of an Options screen presented to participants.

![Difficulty Presets]

**Figure 1. Portion of Options screen presented to participants**

The term “AI Level” was not well understood by most—in fact, just two of seven understood the intended meaning of the term. In addition, the options presented for AI Level (Low, Med, High) failed to provide helpful context cues. The main problem was with the term “AI”. While it is a jargon term for game developers, it is unfamiliar to many gamers. It is worth noting that all participants had experience with computer games, and the majority of them also had experience with flight games. A poor choice of fonts further compounded the problem. The font
used was sans serif resulting in participants confusing the uppercase “I” with a lowercase “i”, or the numeral “1”. This introduced incorrect cues that some participants used as a basis for guessing (e.g., “is it ALtiitude?”). These quotes from participants were representative of the problems they encountered.

“Al-One-level? I don’t know what that means, I have no idea…”

“A. I uh...this uh...AL level? A.I level? I don’t understand this terminology.”

“A-One-level, I’m not sure what that means, or AL level.”

Figure 2. Final interface for Options screen.

Figure 2 represents the final solution. The term “AI Level” was replaced with “Enemy Level”. The options that were originally “Low,” “Medium,” and “High” were replaced with “Rookie,” “Veteran,” and “Ace”. Subsequent testing detected no difficulties with these terms, and the options are relatively clear in the context of a WWII fighter pilot game. This terminology has remained consistent through subsequent releases of the Combat Flight Simulator series.

In this example, information obtained from users through current usability practices was used to improve upon the usability of the game shell. This is a case where the user goals for the options screen did not substantially differ from user goals in productivity applications; the tools were a means to an end. Users should be able to set their preferences with maximum learnability and minimum errors.

2.2 MechWarrior 4: Vengeance

The popular MechWarrior series is fought by elite warriors who pilot Battlemechs. One of the defining features of this action-oriented game is the limited ammunition, differential fire power for the Battlemech to overheat and weapons. This level of complexity created a simplified solution to the problem - because some of the game’s fun stems from the novice users sufficiently master the following case study, user-testing novices who experienced difficulty with MechWarrior 4: Vengeance (MW4) define the MechWarrior series.

The development team established a new system that was possible for novice users, in particular novice movement and weapon controls were placed in the upper part of the torso aiming weapons while playing the game on-screen aids were included as part of the novice’s experience.

Figure 3 is a screenshot of the player’s interface on the screen that were created within the circular radar placed at the bottom portion of the Battlemech's line of sight would be to the right of the
2.2 MechWarrior 4: Vengeance

The popular MechWarrior series is based in a science fiction future where wars are fought by elite warriors who pilot giant destructive walking tanks called Battlemechs. One of the defining features in MechWarrior games is that they are action-oriented. However, the games are also designed as simulations, which are games that attempt to model the complexity of the real world. Mastering the complexity is part of the fun for those who enjoy this type of game. For example, in MechWarrior, all weapons were designed with limitations that could exist in the real world, such as limited ammunition, differential firing rates for different weapons, and the potential for the Battlemech to overheat and shut itself down with the overuse of some weapons. This level of complexity can be quite intimidating for novice users. A simplistic solution to this problem would be to make the game simpler. However, because some of the game's fun stemmed from its complexity, the goal was to help novice users sufficiently master the intricacies of the game in an enjoyable way. In the following case study, user-testing methods helped identify areas of assistance for novices who experienced difficulties controlling their Battlemechs when playing MechWarrior 4: Vengeance (MW4), without sacrificing the complexity that helps define the MechWarrior series.

The development team established a goal of making MW4 as approachable as possible for novice users. In particular, novices needed to be able to use the basic movement and weapon controls with minimal difficulty. The controls in MW4 are complex because Battlemechs are capable of a movement called torso-twisting, where the top part of the Battlemech can rotate to look in a direction different from its movement trajectory, similar to a gun turret on a modern tank. Weapons are placed in the upper part of the torso, so rotating the upper torso is fundamental to aiming weapons while playing the game. To help users coordinate these movements, on-screen aids were included as part of the standard interface.

Figure 3 is a screenshot of the MW4 interface during gameplay. There are two items on the screen that were created to aid users. The green cone-shaped item within the circular radar placed at the bottom-center of the screen represents the user's line of sight, or field-of-view. It is marked Field of Vision. Figure 4 is an enlarged version. In this situation, the torso is rotated slightly to the left relative to the bottom portion of the Battlemech. If the cone funneled out toward the right, the line of sight would be to the right of the Battlemech.
Figure 3. User interface for MechWarrior 4 during gameplay

Figure 4. Field of Vision indicator and Radar

The second item is a green horizontal scale placed in the center of the screen beneath the targeting reticle where a horizontal bar represents the amount of torso rotation (see Figure 3). This is marked as Torso Twist. This provided users with a graphical cue for torso rotation on the x-axis relative to the bottom part of the Battlemech. For example, if a Battlemech is facing straight forward (i.e., the torso is

perfectly aligned with the lower part of the reticle. The length of the torso moves away from the lower part of the reticle, the length of the bar grows as the Battlemech rotates. The upper (torso) and lower portion of the display represents the Battlemech rotated left or right. When the Battlemech is rotated much further, the display

Figure 5. Torso Twist indicator

Usability testing of these interface elements using on-screen counting techniques. During this testing, the basic controls of the game were observed. The Battlemech’s torso, frequently changing direction in which the Battlemech was facing, and few people were using the on-screen counting techniques. Of seven participants, three of them noted that the Battlemech’s torso was facing different directions. No participants noticed or used the on-screen counting techniques. It became apparent that torso rotation was not well understood by the participants.

It became apparent that torso rotation was not well understood by the participants. Based on this feedback, the development team chose to create a training mission to educate users about the visual interface and the safe environment to practice the Battlemech’s torso rotation. The training mission decreased the number of participants who were not able to identify the correct or avoid torso-twisting problems. Participants who completed the training mission were able to identify and avoid torso-twisting problems.
perfectly aligned with the lower portion of the Battlemech) there is no horizontal bar beneath the reticle. The length of the horizontal green bar is a function of the distance the torso moves away from the center in either direction. In other words, the length of the bar grows as the Battlemech rotates further from perfect alignment of the upper (torso) and lower portions of the Battlemech. In Figure 5, the green bar represents the Battlemech rotated slightly to the left in panel A. In panel B, the Battlemech is rotated much further away from the center toward the left.

![Figure 5. Torso Twist indicator. A) slightly turned to the left, B) severely turned to the left](image)

Usability testing of these interface items included common pass/fail and error counting techniques. During this test it was shown that novice users struggled with the basic controls of the game. In particular, they struggled to control the Battlemech’s torso, frequently confusing the direction of the upper torso for the direction in which the Battlemech was moving. The fundamental problem was that few people were using the on-screen cues created to avoid this exact problem. Four of seven participants did not notice or use the field of view reference (green cone), and no participants noticed or used the green rotational bar. Only two participants demonstrated a skillful use of torso twisting, but all demonstrated difficulties steering their Battlemech.

It became apparent that torso-twisting with Battlemechs is a skill with a gradual learning curve. Based on these results and other convergent evidence, the development team chose to create a training mission with the primary goal of educating users about the visual cues in the interface while giving them a relatively safe environment to practice the use of torso-twisting. Follow-up testing showed that the training mission decreased the torso-twist problem, but did not eliminate it. In a follow-up study, all participants knew how to use the on-screen visual cues to correct or avoid torso-twisting problems, though they still struggled to control their Battlemechs. The critical difference between the game prior to the addition of the
training mission and after is that although users still experienced difficulties learning the controls, they were empowered to fix the problems themselves after completing the training mission.

Educating novice players through training makes the game more accessible to a larger population without alienating the current population of MechWarrior gamers. The training mode gives more time to novice users to adapt to the game's complexity, and for more experienced users, the training mode is optional.

2.3 Halo: Combat Evolved

A skilled game designer can create an experience that people consistently enjoy. However, as with any product, a user may not use it in the way the designer intended. In the case of video games, this may lead to an experience that is less fun for the user. This case study presents an example from the development of Halo: Combat Evolved (Xbox). It presents one of the biggest challenges in game design: making the user feel like they are making interesting decisions while ensuring that they are playing the game in a way that leads to the most fun. This example illustrates how discount usability methods enabled the designers to see how users approached their game, to understand barriers that were preventing users from experiencing the game as intended, and to refine the game so that it drew the user into the optimal experience.

One of the greatest difficulties involved in game design is perfecting the balance between scripting an experience intended by the designer without making the user feel restricted in the choices they can make. For example, a designer may want the user to move the character to a particular location in the virtual world, where an exciting battle awaits them. This could be accomplished by creating a narrow pathway, or by placing barriers in the virtual world restricting the user to only one path, with no other choices. However, such restrictions may be unsatisfying to users because they feel like they have no control over their decision-making. Alternatively, the designer could implicitly control the user’s behavior by giving him or her free range of movement, but placing something attractive (e.g., a new weapon) at intermediate points between the user’s current location, and the designer’s desired location (i.e., the location of the exciting battle). This may draw the user down a certain path, while making him or her feel in control. While tactics such as this are common, there are a number of different variables encountered in game design that interact with one another, making it very difficult to predict exactly how users will play the game.

Different mechanisms affect a user’s experience of combat in games. Because Halo is a game about combat, it was very important to the team that combat be compelling. The development team spent a lot of time refining the aiming controls, enemy behavior, weapon variety, and the layout of the environments to make combat as enjoyable as possible. In addition, the designers created a wide variety of combat situations and areas where users could engage in different tactics to succeed. Certain situations were created where a particular strategy would yield the most rewarding and fun experience. The assumption was that if users approached the combat situations as the designers did.

The Halo team wanted to see how the game, and if not, what issue information, eight participants recruited to play the game in a usability study.

Participants were presented with Halo at home. For this test, they were used. Instead, the focus was on level in comparison with the ‘ideal’ level present at all sessions. This was to ensure players to respond to the cues users were not seeing.

The designers had created a level freedom to approach situation to environments afforded players. Testing revealed that participants tended to avoid shooting at the enemy as soon as they encountered a player or move closer to the player. At greater distances, the experiment was designed to be most enjoyable when the player was not at an optimal distance. From observing the usability solutions. First, they revised subtle. The aiming reticle refers to a graph that represents where the user is aiming from the enemy, the reticle would make him feel like his weapon was aiming at the enemy behavior so that it turns combat distance intended by the designer. The enemy from a long distance, the reticle would move closer. When the user reached closer, they would step out of cover and engage the enemy would approach the player.

Subsequent usability testing showed that users to engage in combat from the interesting components of their experience, which would yield the most rewarding and fun experience. The assumption was that if users approached the
chance that people consistently enjoy. Not use it in the way the designer lead to an experience that is experience from the development of Halo: the biggest challenges in game design: testing decisions while ensuring that leads to the most fun. This example helped the designers to see how users were preventing users from the game so that it drew the user.

Game design is perfecting the balance the designer without making the user example, a designer may want the situation in the virtual world, where an accomplished by creating a narrow world restricting the user to only one friction may be unsatisfying to users control over their decision-making. control the user’s behavior by giving something attractive (e.g., a new user’s current location, and the the exciting battle). This may draw or her feel in control. While tactics of different variables encountered in making it very difficult to predict exactly chance of combat in games. Because important to the team that combat be time refining the aiming controls, out of the environments to make designers created a wide variety of engage in different tactics to succeed. A similar strategy would yield the most was that if users approached the

The Halo team wanted to see whether the typical user would have fun playing the game, and if not, what issues were blocking them from doing so. To provide this information, eight participants who liked to play console action games were recruited to play the game in a usability test format.

Participants were presented with a mission and asked to play the game as they would at home. For this test, traditional error counts and pass/fail criteria were not used. Instead, the focus was on the tactics that participants used to complete the level in comparison with the ‘ideal’ tactics created by the designers. Designers were present at all sessions. This was critical because they best understood how they wanted players to respond to their game, and they could also see, first hand, what cues users were not seeing.

The designers had created large, outdoor environments that gave players more freedom to approach situations in a variety of ways. However, the large environments afforded players the ability to see enemies from a long distance. Testing revealed that participants were trying to fight enemies from much further away than the designers intended. During usability sessions, novice players began shooting at the enemy as soon as they could see them, but the designers had wanted the players to first move closer to the enemies, and then fight. While all users were able to successfully complete the missions using these tactics, the game was designed to be most enjoyable when combat was engaged at a much closer distance. At greater distances, the experience was less satisfying. For example, the weapons were not designed to be effective at these longer distances resulting in participants complaining that their weapons were not accurate enough.

From observing the usability sessions, designers quickly identified several solutions. First, they revised subtle cues that they had placed in the aiming reticle. The aiming reticle refers to a graphical indicator in the center of the screen that represents where the user is aiming a weapon. In the original design, at any distance from the enemy, the reticle would turn red when placed over an enemy, informing the user that their weapon was aimed properly at the enemy. The designers modified the reticle behavior so that it turned red when the enemy was targeted, but only at a combat distance intended by the designers. Therefore, when the player aimed at the enemy from a long distance, the reticle would not change color, providing a subtle cue that they were not at an optimal distance, thus encouraging users to move closer. The designers also increased the diameter of the reticle to emphasize its inaccuracy at greater distances. Second, the team modified the enemy behavior. When the user shot at the enemy from a much greater distance, the enemy now would not return fire, but would dodge the shot or move behind a rock or tree, requiring the user to move closer. When the user reached the optimal engagement point, the enemy would step out of cover and engage in battle with the player. At other times, the enemy would approach the player to maintain the intended combat distance.

Subsequent usability testing showed that these changes encouraged the majority of users to engage in combat from the intended distance. Users were able to see more of the interesting components of the game, were more satisfied with the performance of their weapons, and as predicted, enjoyed the game more.
In standard usability practices, the goal of the test is to ensure that the design of product effectively supports what the user is trying to accomplish. In this case study, the perspective of usability test goals has been shifted away from the end user, and toward the designer. In the Halo example, the goal was to ensure that the user experience matched designer intention. These types of situations often require less structure than more common usability methods.

Once we are confident that users are playing the game in the way that designers intended, we can begin to get feedback about subjective preferences on the game's design. Although this may not require direct observation, a larger number of users must be utilized to ensure reliability of participant response, as the next case study illustrates.

2.4 RalliSport Challenge

RalliSport Challenge (RSC) is a racing game on the Xbox console. As in many popular racing titles, the Career Mode is one of the core pieces of this game. In order to understand how user-centered design methods were able to influence the design of this game, one must first understand how the Career Mode functions in RSC.

The Career Mode starts with a limited set of cars available to the user, and a limited set of driving events they can participate in. Depending on the users' performance, new cars and more challenging events are made available to users as they progress through their careers.

Each event consists of several stages (or races). After each stage, users gain points based on a number of factors: top speed, what place they finished (e.g., 1st, 2nd, etc.), lap times, and amount of car damage. Once an event is over, all of the points gained from each stage are added up to obtain the user's total points. The user's goal is to accrue a specified number of points to gain access to the next set of cars and events that are currently unavailable to them. Reaching the specified point total is the only way to progress through the game. This is often referred to as "unlocking" cars and events.

This design is not unique in racing games. It is common to specify a certain performance criterion in order to progress through the game. However, keeping in mind that playing games is a choice (i.e., users are not forced to play), the manner in which this design is implemented could cause frustration for many users. In RSC, the issue revolved around the users' ability to re-race a stage within an event to earn more points.

To better illustrate this issue, consider a scenario where a user has three racing events that are currently available to them, each comprising several stages: Event A (stages A-1 through A-4), Event B (stages B-1 through B-5), and Event C (stages C-1 through C-6). To unlock the next set of events and cars, the user must accumulate a minimum set of points by racing through most (and sometimes all) of the stages in Events A, B, and C. If a user races all Events, but does not earn enough points to unlock new cars and events, what options are they left with? According to the developer's design, their only option would be to re-race an entire Event, including all stages within that Event. If the user selected Event C, all of the previous points earned from each stage (C-1 through C-6).

The developers preferred this easy otherwise, and also because rally racing. This is a valid argument: the appropriate level of challenge potentially puts users in the situation they are already happy with. If the user doesn't perform very poorly, they could result in extreme frustration.

To address this issue, a series of scenarios may be to users. The game was as possible a situation where users would not accrue enough points. Some users had to experience the game and the user's experience with high points to achieve a sense of progression. Therefore, a survey was conducted to ensure not to adjust the design to fit the user's needs.

Working with designers, a modify was made. Each Event, each with a minimum of the population of users was created. The appropriate console games were brought into the standard experimental research protocol, set up so that responses from each user's interaction between users.

Participants were instructed to unlock the next set of events. Once they finished the stage, they were given the following options: a) Re-race any of the previous 3 Events. b) Re-race certain stages in your total score.

Then participants were asked to unlock the next event:

- Re-race an entire event (including all stages).
- Re-race any stage of my own design.

Lastly, users were asked the same questions: whether the Events consisted of nine stages, or the actual game, although they did not change.

The data gathered provided feedback for re-racing the entire Event. The majority of users preferred the ability to re-race a single stage as they became more experienced when provided with the percentage of users preferred to be
CHAPTER 11

It is important to ensure that the design of the game is not too complicated. In this case study, it was necessary to find a way to avoid the user from getting stuck in situations where they would lose points. This was done by carefully designing the game's structure, ensuring that a larger number of users could experience success, as the next case study will demonstrate.

Xbox console. As in many other pieces of this game, in order to avoid the game design from influencing the design of node functions in RSC, the available to the user, and a dependency on the users’ preferences on the game’s design. In the next section, a larger number of users will be able to experience success, as the next case study will demonstrate.

After each stage, users gain points based on whether they finished first or later (e.g., 1st, 2nd, 3rd, 4th). Each event is over, all of the user’s total points. The user can access the next set of stages via the total points threshold (this is often referred to as the “level”). It is common to specify a certain threshold for a certain stage within an event to earn the points.

For instance, if a user has three racing stages (Event A through Event C), and Event C stages C-1 through C-3, and the user must accumulate points (by default) of the stages in order to earn points to unlock the next event. According to the game design, an entire Event, including all of the previous points earned from each stage (C-1 through C-6) are erased, and the user must then re-race stages C-1 through C-6.

The developers preferred this model because they felt the game would be too easy otherwise, and also because this system is much closer to the actual sport of rally racing. This is a valid argument. Games should not be too easy and must ensure the appropriate level of challenge and success. On the other hand, this strategy potentially puts users in the situation where they lose points on several stages they are already happy with. If the user performed very well on C-1, C-2, C-3, C-5, and C-6, but performed very poorly on C-4, they would lose points on all stages, which could result in extreme frustration.

To address this issue, a series of tests was designed to assess how frustrating this situation may be to users. The goal in designing these tests was to simulate as close as possible a situation where users would actually play through a series of events, but not accumulate enough points. Simply asking users about this situation would not suffice; users had to experience it. In addition, data collected from a traditional usability study sample size would not be sufficiently reliable to measure a subjective preference. Therefore, a survey methodology that would allow for a larger sample size and not require direct observation was used.

Working with designers, a mock career mode was created that consisted of three events, each with a minimum of four stages. Depending on the user’s total points, they were able to access the next set of stages within the event they played. The manner in which this was implemented for many users was as follows. In RSC, a stage within an event to earn points (total points) was associated with a certain threshold. It is common to specify a certain threshold for a certain stage within an event to earn points.

1. The user enters a career stage (Event A through Event C), and each event has three racing stages (Event A through Event C).
2. Each stage has a certain threshold (total points) to unlock the next set of stages within the event.
3. The user can accumulate points by completing stages in order to unlock the next set of stages within the event.
4. The user must accumulate points on all stages in order to unlock the next set of stages within the event.
5. Lastly, users were asked the same question, but presented with a situation where the Events consisted of nine stages (which was representative of some Events in the actual game, although they did not experience this).

The data gathered provided fairly convincing evidence against forcing users to re-race the entire Event. The majority of participants reported a preference for being able to re-race a single stage as opposed to an entire event and this preference became even stronger when presented with the nine stage event. A smaller percentage of users preferred to be able to re-race only entire events, suggesting that
they agreed with the developers, who felt that allowing users to re-race stages made the game too easy.

Based on these survey data, the recommendation was to implement multiple difficulty levels, one of which allowed the re-race option (Normal difficulty), but the more advanced levels that did not (Difficult or Advanced difficulty). This solution met the needs of both user preferences, those who want to be able to re-race any stage at any time, and “hard-core” users who prefer more of a challenge and penalty when playing through their career. This was presented to the developer, who in turn implemented the functionality into the design of the final retail product—Normal (no re-race) and Easy (re-race). The developer felt the no re-race option was truer to the sport of rally racing, which is why it was termed Normal. More importantly, the decision to implement multiple difficulty levels was also validated by some reviews from popular gaming magazines and gaming sites after the game was released.

...punishingly difficult and repetitive, depending on which of the two available skill levels you choose when you start the game...With no “redo” button available in Normal mode to simply restart a race if you screw up, some gamers will find themselves angered and frustrated with having to start over from the beginning of the series, while others will embrace the degree of challenge with open arms - Mahood, 2002

We were also ready to slap the game down for being too hard, until we tried it on the easy difficulty setting. Normal difficulty, as well as giving you some stuff times to beat, doesn’t allow you to retry a stage – if you mess up, you have to restart the whole event. Even racing veterans would be well advised to play Easy mode to get to grips with the tracks at first, as this allows you to retry. - Smith, 2002

In this example, the question being addressed (to allow re-racing or not) did not have a clear answer before data were collected, and did not have a specified task-based performance criterion. Finding a solution for this design question was not necessarily related to the learnability of the product, or the efficiency of use, or the number of errors a user would make. The solution was based on what was most fun for users.

3. CONCLUSION

There are a variety of approaches that can be taken from existing HCI-related methods to address the unique aspects of user-centered design issues found in games. Although the case studies discussed above focus on specific issues, they should also be viewed as a broad representation of some methods currently being utilized by the User-Testing Group at Microsoft Game Studios. Each of these examples is part of a larger series of tests performed throughout the development cycle of each of the games. It is not sufficient to run a single test for one game. As in any HCI-related field, to be maximally beneficial, user-testing must be integrated into the development process from the very beginning.

Each of the methods presented in the case studies has limitations, and should only be used when appropriate. A good user-testing engineer must understand the points in the development cycle at which certain methods should be used to address particular problems, and can collect through user testing.

In general, standard usability methods, subjective preferences, due to the limited number of participants. However, usability methods can provide rich information about a game and help designers with greater understanding of their game.

Larger sample methods are limited because they rely on a user’s actions, are better in traditional usability testing, it may be sufficient to design a game. Users can on not”). From there, it is up to the designer to define the root causes of issues.

Unfortunately, there is no variability in the process through the methods used for addressing these issues.

Furthermore, there are many methods in the user experience when playing utilized in testing adults prove from children on child-oriented a game (when two or more players), and difficulties. Interactions between observations are independent, with the results (i.e., subjective ratings) within the game, or pre-existing.

Finally, most often our results are made in our research and developers are so immersed with the experience is like for someone development teams would also be interested in understanding the user experience.

For more in-depth case study 2, Oddworld: Munch’s Oddysee of Empires II).
particular problems, and convince development teams of the general value of data collected through user testing.

In general, standard usability methods are not appropriate when assessing subjective preferences, due to the small sample size that accompany typical usability studies. With fewer participants, reliability of the results comes into question. However, usability methods used in conjunction with more open-ended tasks provide rich information about how a user interacts with a product which can provide a designer with greater understanding of how a user approaches and plays their game.

Larger sample methods are good for assessing subjective preferences, but they are limited because they rely on self-report. Usability studies, which typically focus on a user’s actions, are better for obtaining in-depth, behavioral information. Just as in traditional usability testing, it is also impossible to look to the user to tell you how to design a game. Users can only evaluate aspects of the game (“this is fun”, “that is not”). From there, it is up to the user-testing engineer and designer to work together to define the root causes of issues and create compelling solutions.

Unfortunately, there is no set formula for how the pieces fit together. The variability in the process through which games are developed must be reflected in the methods used for addressing user-testing issues.

Furthermore, there are many challenges with obtaining feedback on all areas of the user experience when playing games. For example, the survey methods that we utilized in testing adults proved to be problematic when trying to obtain feedback from children on child-oriented games. Additionally, testing multiplayer portions of a game (when two or more people play the same game together) presents further difficulties. Interactions between participants do not allow us to assume that the observations are independent, which makes it difficult to know the extent to which the results (i.e., subjective ratings) are influenced by the players' social interactions within the game, or pre-existing social relationships between them.

Finally, most of our research focuses on the initial experience of a game. Information based on this stage of the user’s experience is critical because designers and developers are so immersed in the game that they often forget what the initial experience is like for someone who has never played their game. While many development teams would also like to obtain detailed feedback about gameplay that occurs beyond the initial experience, doing so with current development practices has many practical constraints. As such, these issues need to be explored further to provide cost-efficient user-testing methods to obtain feedback on all areas of the user experience.

4. ACKNOWLEDGMENTS

Special thanks to the development teams on CFS, MW4, Halo, and RSC and Digital Illusions. In particular, we’d like to express our gratitude to Christina Chen, TJ Wagner, David Luehmann, Jaime Greeves, John Howard, Hamilton Chu, and Peter Wong for permitting us to use the actual games in our examples. We’d also like to thank the Microsoft Game Studios User-Testing Group, in particular, Lance Davis, Kyle Drexel, Kevin Goebel, David Quiroz, Bruce Phillips, John Davis, and Kevin Keeker.

5. REFERENCES


The nature of user studies is complex, and it is relevant to study and how to study people’s actions but the social and behavioral. Now, with the so-called persuasive can be enjoyed by the user; in order for a product to be satisfying and pleasurable in the personal sphere and useful emotions? How do we interpret?

The work of Gaver et al. (1986) shows that or creative techniques can be used to design experiences is a possible way to reveal meaningful data. We believe that preliminary studies in the design produce are not scientific in a manner, while designing for products must be measurable and must be interpreted.

This paper presents the application of the project was to develop communication between people. To explore the perception of distance; an additional which to access and study people’s investigations, we propose core products we design and the maps.