

Critic-Proofing: How Using Critic Reviews and Game Genres can Refine Heuristic Evaluations

Ian J. Livingston

Department of Computer Science
University of Saskatchewan
Saskatoon, SK S7N 5C9 Canada
ian.livingston@usask.ca

Regan L. Mandryk

Department of Computer Science
University of Saskatchewan
Saskatoon, SK S7N 5C9 Canada
regan@cs.usask.ca

Kevin G. Stanley

Department of Computer Science
University of Saskatchewan
Saskatoon, SK S7N 5C9 Canada
kstanley@cs.usask.ca

ABSTRACT

Heuristic evaluation – a technique where experts inspect software and determine where the application violates predetermined policies for good usability – is an effective technique for evaluating productivity software. The technique has recently been applied to video games, examining playability and usability for both single and multiplayer games. However, the severity ratings assigned to usability problems and used as a coarse categorization method for triage are still subjectively and somewhat arbitrarily assigned by evaluators, offering limited organizational value. In addition, they fail to account for the diversity found between games and game genres. In this paper we present a modified heuristic evaluation technique, which produces a prioritized list of heuristic violations based on the problem’s frequency, impact, persistence, the heuristic it violates, and the game’s genre. We evaluate our technique in a case study and show that the technique provides substantial value with little additional effort.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation] User Interfaces – *evaluation/methodology, user-centered design.*

General Terms

Measurement, Design, Human Factors, Verification.

Keywords

Critic-proofing, heuristic evaluation, usability evaluation, game genre, severity rating, video games

1. INTRODUCTION

Development in the computer game industry is fast-paced and tightly scheduled. Iterative evaluation of games during development must adhere to this rapid schedule, making time-consuming approaches undesirable. Discount usability evaluation methods have recently been adapted for the games industry due to their low cost, short time requirements, and reliable feedback to developers. For example, in heuristic evaluation, evaluators

examine features and system actions that violate predefined heuristics, and assign a severity rating to any discovered problem [12] – a fast, cheap, and reliable method of evaluation. However, the interaction and interface components of computer games are different from productivity software, and require improvements to discount evaluation processes, such as heuristic evaluation, if they are to be relevant and useful to game developers.

Heuristic principles are often designed to generalize across applications, failing to take into account the diversity in game types, game controls, and game interfaces. The diversity in video games is a testament to creativity, but also drives variety in usability requirements so a given heuristic may not be equally relevant to all games. Despite the differences between games, reoccurring similarities between games exist within the same game genre. Often games that share a genre will have similar control schemes and interface layouts, for example, in the PC shooter genre W, A, S, and D are almost exclusively used for character movement, and the SPACE bar is used for jumping. Players who are accustomed to the shooter genre expect these control mappings. These genre characteristics can aid game designers by providing a template for the interaction, allowing new designs to borrow from previous games. While heuristic evaluation for games must encompass a variety of different interfaces, taking the game’s genre into account can produce a superior starting reference point for analysis.

A heuristic evaluation results in a list of usability problems and an accompanying severity indicator. These severity ratings are generally grouped into coarse-grained categories based on the problem’s impact, frequency, and persistence, and are determined subjectively by an expert evaluator. Although a particular problem can be classified as severe, there is no method for prioritizing the problems within a category. In addition, there is no means of prioritizing problems based on the game’s genre. For example, poor control mappings will be a more significant usability issue for an action game than a strategy game [18]. Our method addresses the issue of genre specifically by using critical reviews as a method of determining the importance of a usability problem. The more often critics encounter a specific usability problem within games of the same game genre, the more important we consider that category of usability problem.

By incorporating critical reviews into our evaluation process we move towards an approach that will ‘critic-proof’ games. The goal of a game developer is to release a game where all potential criticisms by reviewers have already been identified and fixed. Critic-proofing a video game is a difficult task, especially when a game is entirely new and there are no reviews of previous versions. Our modified heuristic evaluation approach describes a method for developers to use in their evaluation process to critic

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ACM FuturePlay 2010, May 6-7 2010, Vancouver, BC, Canada.
Copyright 2010 ACM 978-1-4503-0235-7 ...\$5.00.

proof the usability of their games prior to release. For this paper we refer to the heuristic evaluation process described by Nielsen as the traditional heuristic evaluation process.

We modified the traditional heuristic evaluation process to address the following problems:

1. The feedback from heuristic evaluations tends to be grouped into coarsely-grained categories instead of prioritized lists.
2. Any prioritization that does result is a subjective choice.
3. Heuristic evaluation fails to take the diversity of video games and game genres into account.

We employed our evaluation technique in a case study of a video game currently in development. The results from the case study suggest that our technique provides significant value in the identification and prioritization of usability problems.

In this paper, we present relevant prior work on the heuristic evaluation process, video game genres, and the concept of ‘critic-proofing’ a game. The evaluation technique is then presented in detail, describing how an evaluator would generate a prioritized list of usability problems. Next, a case study is presented, where we analyze the use of our evaluation technique on a game currently under development in a natural development setting. Finally, we discuss the advantages and limitations of our technique and our future plans for research in this area.

2. RELATED WORK

Previous research has investigated the importance of game heuristic techniques for a conceptual understanding of playability [11]. In our work, we have furthermore identified four relevant areas of work that directly relate to the heuristic technique used in this paper: (1) severity ratings, (2) video game heuristics, (3) critic proofing, and (4) video game genres.

2.1 The Heuristic Evaluation Technique

Heuristic Evaluation is one of the discount methods used for usability inspection [16]. In heuristic evaluation, human-computer interaction experts inspect a user interface and identify usability problems. Usability problems are categorized into ten heuristics defined by Nielsen [15], which focus on issues like providing help and consistency. In addition, usability problems are given a severity rating based on the impact, frequency, and persistence of the problem. Heuristic Evaluation is one of the most commonly-used software usability evaluation methods as it is cheap, fast, and reliable, and can be conducted throughout the development cycle, on low-fidelity paper prototypes to fully-realized applications.

2.1.1 Video Game Heuristics

Heuristics are not new to game research. Federoff [4] was one of the first pioneers into game heuristics. Through observations and conversations at a San Francisco game company, and a comprehensive review of literature, Federoff was able to compile a list of heuristics, which focus on three game areas: interface, mechanics, and play. Federoff’s heuristics are quite broad, while we are specifically interested in problems pertaining to usability.

In 2004 Desurvire et al. [1] developed a set of heuristics to evaluate game playability, or HEP. Although similar to Federoff’s heuristics, Desurvire provided a simpler description and organization of the heuristics. Recently Desurvire has published a newer version of the HEP heuristics called PLAY [2]. In this she attempts to make the underlying principles more generalizable by applying the heuristics in three different genres during

evaluations. In the scope of our paper the HEP and PLAY heuristics apply to far more than just usability, and the genre differences were only briefly explored.

Other forms of heuristics have been developed to fill different needs of game evaluation. Korhonen et al. created heuristics for mobile multiplayer games [8]; Pinelle et al. generalized this idea with a set of heuristics for all multiplayer games [19]. Many of these heuristics have similarities and overlap, and some attempts have been made to compare techniques. Korhonen et al. compared two different sets of playability heuristics – HEP [2] and their own mobile games heuristics [7] – to examine the relative strengths and weaknesses of each approach [9]. Koeffel et al. [6] attempted to create a comprehensive list of heuristics from those found in literature, and assess the collection’s effectiveness using a comparison to video game reviews.

In our study we have employed the usability game heuristics created by Pinelle et al. [19]. While others have also provided usability heuristics, Pinelle’s list is specific and short—there are ten principles—making the evaluations clean and simple to perform. Also, in addition to this, Pinelle’s work has been extended in the exploration of usability specific to genres [20].

2.1.2 Severity Ratings

The method of categorizing usability problems is key to deciding which should receive developer attention first. Most recent work in the area of heuristics research focuses on the development or modification of the heuristics themselves, and on testing their validity. These game-specific heuristic techniques use one of two methods for assigning severity ratings, both developed for evaluating productivity applications: Nielsen’s severity rating [14], or Dumas and Redish’s severity rating [3].

Both methods (Nielsen’s and Dumas’s) rely on a combination of evaluator and development judgment in determining the severity of any discovered problems [13]. Nielsen describes a five category method ranging from “*I don't agree that this is a usability problem at all*” to “*Usability catastrophe: imperative to fix this before product can be released*” with the severity being assigned based on the frequency, impact, and persistence of a problem [14]. Dumas and Redish, on the other hand, use four levels ranging from, “*Level 1 problems prevent completion of a task.*” to “*Level 4 problems are more subtle and often point to an enhancement that can be added in the future*”. In addition Dumas assigns one of two scopes to a problem, either: local—applying in a single instance such as one screen, or dialog box—or global—applying to more than one instance perhaps across multiple screens, or systemic across all dialog boxes [3]. Both approaches are limited in two ways: they rely solely on human appraisal, and are coarse in their granularity. Together, these limitations mean that many problems are simply assigned the same severity rating.

Recently there have been attempts to map heuristic results to the ratings of game critics. In a recent book chapter [6], Koeffel et al. presented a study where heuristic evaluations were performed, using Nielsen’s ratings [14], and a score was assigned to each problem based on its severity. The score was then compared to scores received from critics. This is a valuable step forward, however Koeffel et al. still rely on a purely human appraisal method, and while they discuss the importance of game genres, they don’t account for usability differences found between genres.

2.2 Critic-Proofing

This paper describes a technique that uses the reports of game critics to address common usability problems before post-release critical analysis exposes them. By identifying interface factors that have historically caught critics' negative attention, we can prioritize effort to minimize historically common shortcomings.

The voice of a professional critic is powerful. A single influential critic can have a significant effect on any product, influencing consumers purchasing decisions. Larsen [10] explored how a critic's review is in essence an unscientific user experience evaluation. Larsen identifies that game reviewers generally provide both a qualitative (textual review) and quantitative (score) component. Some reviewers evaluate different aspects of a game separately. In many cases these aspects are comparable to hedonic and pragmatic aspects of user experience research.

Koeffel et al. also offers support for the use of the reviews written by game critics [6]. In their recent study, Koeffel et al. compare results of heuristic evaluations to the average critic scores of games, and suggest that a similar trend between review scores and the number and severity of usability problems exists.

The term critic-proofing has recently been used to describe the technique used at BioWare in MassEffect 2. In an interview with Gamasutra [17], producer Adrien Cho described how the development team mapped critic and player feedback from the first game onto the design objectives for the sequel.

"But the other part of the equation was actually taking all the feedback -- I'm not saying some -- absolutely every feedback from press and the fans, and collating all that into a huge list.

Everything eventually fit within certain categories, and when we looked at that, mapped with the things that we wanted to fix, it became really clear. It became a blueprint. It made making the sequel really easy."[17]

This critic-proofing process can only be applied to a sequel as critic feedback is difficult and expensive to collect before release.

2.3 Video Game Genres

Our work draws heavily from work published by Pinelle et al. in [18] and [20]. The underlying premise of Pinelle et al.'s work, as established in [18], is that game reviews can be used as a proxy for providing empirical data for deriving game heuristics when a substantial body of scientific research is not available. To support this hypothesis, they analyzed 108 reviews by 24 different reviewers of moderately to poorly rated games from a popular game review website, and derived a set of novel heuristics for game interface issues, summarized in Table 1. They tested these heuristics by having experts use them in a test of a different, but completed game not included in the original review set.

In their subsequent publication [20], Pinelle et al. further probed their dataset to determine if usability problems from the reviews were correlated with game genre. They used the genre definitions from the review site, which was composed of the common Role playing, Sports, Shooter, Action, Strategy and Adventure genre classifications. Statistical tests were used to show that while some problems spanned all genres, many were genre specific; for example "Skip Content" was only a problem in Adventure games, while controls issues were critical in Action and Sports genres.

While Pinelle et al. discussed the implications for design in [20], they did not provide any empirical evidence for the utility of their genre-specific evaluation technique. They also noted several

Table 1. Pinelle's 12 usability problems found in video games

| Problem category | Key issues |
|----------------------------|--|
| 1. Consistency | poor hit detection, poor in-game physics, inconsistent response to input |
| 2. Customizability | does not allow user to change video and audio settings, difficulty, or game speed |
| 3. Artificial intelligence | problems with pathfinding, problems with computer controlled teammates |
| 4. View mismatch | bad camera angle, view is obstructed, view does not adjust to user's action quickly enough |
| 5. Skip content | cannot skip video and audio clips, frequently repeated sequences |
| 6. Input mappings | bad input mappings, limited device support, limited control customization |
| 7. Controls | oversensitive controls, unnatural controls, unresponsive controls |
| 8. Game status | does not provide adequate information on character, game world, or enemies. visual indicators, icons, and maps are inadequate. |
| 9. Training and help | does not provide default and recommended choices; does not provide suggestions and help; does not provide adequate documentation, instructions, tutorials, and training missions. |
| 10. Command sequences | learning curve is too steep; requires too much micromanagement; command sequences are complex, lengthy, and awkward, making the game difficult to play |
| 11. Visual representations | bad visualization of information, too much screen clutter, too many characters or game elements on the screen at the same time, difficult to visually distinguish interactive content from non-interactive content |
| 12. Response times | slow response time interferes with user's ability to interact with the game successfully |

potential opportunities and shortcomings: first, that many problems are of moderate impact and could be ignored if developers focused only on high-impact issues; second, that genre is a fluid idea, and that many games span genres or contain mini-games or components from several different genres; and finally that while their findings highlight areas of specific interest by genre, they do not provide a mechanism for compromising between broad and focused development.

Our approach codifies a heuristic technique based on the genre and overall rating systems of Pinelle et al. [20], while addressing the shortcomings they identified to provide a simple and robust method for game usability analysis and critic-proofing.

3. CRITIC-PROOFING HEURISTIC TECHNIQUE

When compared with the heuristic approach presented by Nielsen [12], our evaluation system requires little additional complexity while offering substantial additional value. The two techniques are comparable for the evaluation and determination of severity classification. Our approach adds two additional steps – pre-evaluation to identify the appropriate severity framework, and severity calculation to prioritize the usability findings.

3.1 Technique Overview

Our technique is a simple process that can be added to a traditional heuristic evaluation of a game at any point in the

evaluation process. The additional steps we describe are best utilized in the manner we outline—the pre-evaluation at the beginning, and the severity calculation after problem categorization—however the technique is flexible enough that our process may be applied to a completed evaluation.

To perform our technique the steps in the process are:

1. Determine game and sub-component genre ratings.
2. Evaluated entire software for heuristic principal violations.
3. Subjectively assign classification ratings based on the impact, frequency, and persistence of the violation.
4. Calculate the prioritized severity rating according to problem classification, game genre, and the heuristic violated.
5. Average prioritized severity ratings across all evaluators.

The technique adds little additional overhead to the overall evaluation process, as steps 2 and 3, which are common to all heuristic evaluation techniques, dominate the effort. Steps 1 and 4 may be performed post-hoc, but as heuristic evaluation should be completed individually by each evaluator, averaging across evaluators should always be the last step performed.

3.2 Pre-evaluation

The pre-evaluation step should be performed prior to inspection, to establish the severity frameworks used to prioritize problems.

3.2.1 Assessing Gameplay & Determining Genres

Our first step assesses both the game and gameplay elements to be evaluated. We define a gameplay element as a subsystem of a game that can be described as a separate and complete play experience, particularly identifying mini-games separately for evaluation. This assessment is required to ensure the genres for the game and gameplay elements can be identified.

Every game can be categorized into one or more genres. Our technique uses the six primary genres presented by Pinelle et al. [20]. In their work, each genre has a rating for its applicability to each heuristic category – a genre rating. Rather than assuming that each game falls solely into a single genre, we calculate genre ratings using the proportion of the game that falls into each genre.

3.2.2 Calculate Genre Ratings

Because each genre in Pinelle et al.'s [20] work was found to have distinct usability trends, it is reasonable to base the genre ratings on their observations. We use the trends and the values they observed for each problem category in each genre as the basis for calculating a normalized genre rating. If the game or game elements fall completely into a single genre, then the normalized ratings maybe used directly. However, if the game or elements fit into more than one genre, a weighted average must be calculated using formula 1:

$$R_{g_{norm}} = \frac{\sum_{i=1}^n w_i R_g}{\sum_{i=1}^n w_i} \quad (1)$$

Where w is the genre weight (the proportion of the game that is identified as each genre), R_g is the genre rating, and g_{max} is the maximum possible value of R_g . In this paper the summation of w_i is 1; however, to ensure that formula 1 remains generalized, the summation denominator has been included. This step is completed for each heuristic category separately. Although this equation may appear to be complicated, the approach is straightforward.

Consider a game that has been identified as 20% Action and 80% Shooter and we are interested in the Controls heuristic. Pinelle et al.'s ratings for Controls are 12 for Action and 5 for Shooter. The new genre rating would be:

$$R_{g_{norm}} = \frac{(0.2 * 12) + (0.8 * 5)}{12} = 0.533$$

In this paper we employ genre ratings taken from Pinelle et al. [20] summarized in Figure 1, but our system applies equally to other genre ratings frameworks or studies.

Once the normalized genre ratings have been calculated they can be used in the severity calculation step. This new step is performed following the heuristic inspection, and is used to produce a prioritized list of potential usability problems.

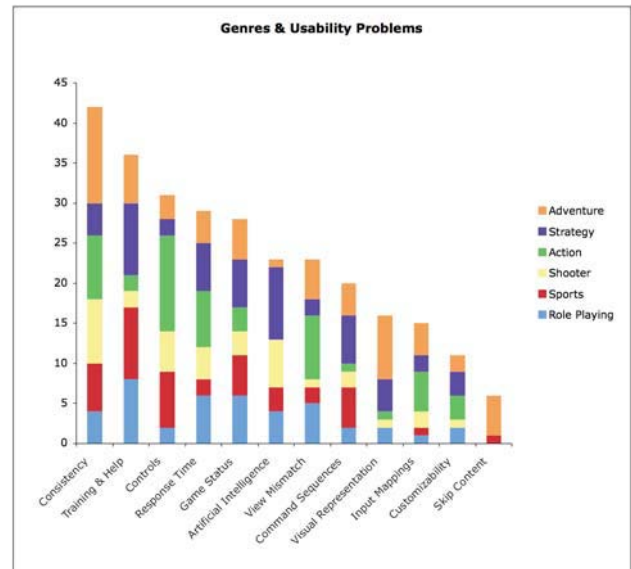


Figure 1: The 12 common usability problems combined across genre into frequency of occurrence of the problem.

3.3 Severity Calculation

In traditional heuristic evaluation, the severity rating is limited by the categorizations available. In many ways the severity rating is akin to triage, coarsely grouping problems into a relatively small number of categories. For example, Nielsen only has five ratings [14], and in his book, Dumas [3] only list four ratings and two different scopes. For traditional software artifacts these categories are sufficient. However, the fast pace and aggressive development cycles in the game industry reduce the value of the evaluation, because a problem's importance within a specific category is ambiguous. It is reasonable to imagine that two problems in a game may both be categorized as a major problem (using Nielsen's ratings) with one problem being significantly more important than the other. Our system removes this ambiguity by assigning a finer-grained severity rating to usability problems, based on empirical evidence presented for specific genres [20]. Each heuristic violation must be categorized into one of the 12 problem categories found in Table 1. The genre rating for the categories can then be used in the prioritized severity calculation.

Our approach classifies problems using a modified set of the severity rating descriptions presented by Nielsen [14]. We add one

Table 2: Normalized Classification Ratings for a slightly modified Nielsen severity ratings

| Classification Rating | Severity Description |
|-----------------------|----------------------|
| -1.00 | Not A Problem |
| +0.00 | Minor |
| +0.33 | Moderate |
| +0.66 | Major |
| +1.00 | Catastrophic |
| +2.00 | Game breaking |

additional rating, ‘game breaking’, to the set. We have assigned values, ranging from negative one to positive two, to each of these descriptions (Table 2). There are two special case entries in this table. The first is the ‘not a problem’ description which has been assigned a classification of -1. This description is used for a problem that has been mistakenly identified. By assigning a value of negative one we ensure that any problems classified in this way will have a negative final normalized priority rating. This will be the case regardless of the genre rating, effectively removing those problems from the list. Similarly the new ‘game breaking’ description refers to a usability problem that renders the game unplayable. Any problem with this classification will be assigned a prioritized rating greater than one, implying it should be addressed regardless of the genre rating.

Any numerically-different classification system may be used in principal, as long as it is normalized according to formula 2:

$$R_{c_{norm}} = \frac{R_c}{c_{max}} \quad (2)$$

Where R_c is a numerical classification rating, c_{max} is the maximum value of R_c , and $R_{c_{norm}}$ is the normalized numerical classification rating.

We then perform a calculation to determine the prioritized severity rating using formula 3:

$$R_p = \frac{R_{g_{norm}} + R_{c_{norm}}}{2} \quad (3)$$

Where $R_{g_{norm}}$ is the normalized genre score ranging from 0 to 1, $R_{c_{norm}}$ is the normalized rating assigned to heuristic severity descriptions, ranging from -1 to 2, and R_p is the final prioritized rating for the usability problem, which ranges between -1 and 2. Generally ratings calculated this way will generally range between 0 and 1, with ratings below 0 or above 1 warranting special treatment. By providing finer granularity in the severity ratings, we remove the guesswork of assessing priority for identified usability problems, providing greater confidence in the results of the evaluation.

The advantage of normalizing all ratings throughout this process is that both the genre rating and classification rating systems can be modified while maintaining a consistent prioritization system.

As an example lets imagine we have an action game that violates Pinelle’s 7th game usability heuristic (“Provide controls that are easy to manage, and that have an appropriate level of sensitivity and responsiveness.”), and the problem has been classified as *minor* (0.00). Lets also imagine there is a problem that violates Pinelle’s 9th heuristic (*Provide instructions, training, and help.*), which is classified as a *major* (0.66) problem. The first problem is

a *Controls Problem* with a rating of 1.00, and the second is a problem with *Training & Help*, which has a genre rating of 0.17. In this case the calculations would be as follows:

$$R_{controls} = \frac{1.00 + 0.00}{2} = 0.500$$

$$R_{help} = \frac{0.17 + 0.66}{2} = 0.422$$

As can be clearly seen, even though the usability problem relating to training and help was classified as a major usability problem, the minor usability problem is assigned a higher priority rating, because of the consistent negative impact of controls problems in the action genre (see Figure 1).

In this section we presented a novel critic-proofing technique for usability evaluation. Our technique differs from previous heuristic evaluation techniques in the unique way that it incorporates common genre problems into the severity calculations. We presented two additional simple steps that, when added, will provide developers with a prioritized list of usability problems. The following section presents a case study where our technique was applied successful to a game currently under development.

4. CASE STUDY: Capsized

In this section we present a case study examining the use of our heuristic evaluation technique on a game currently under development. We will cover the original evaluation of the game using a traditional heuristic process with the heuristic principles presented in [12] and [15]. We will discuss the implementation of our modified evaluation technique and its application to usability problems identified in the original heuristic evaluation. We will also present a comparison of the two techniques highlighting differences in the evaluation findings. We conclude this section with a discussion of feedback from the developer.

4.1 The Game

Capsized (Figure 2) is a PC and xBox Live Arcade (XBLA) game in development from Alientrap Software, designed by Lee Vermeulen (programmer) and Jesse McGibney (artist), using sound effects by Michael Quinn and music (used with permission) from Solar Fields.

The gameplay is based on classic platformers, focusing on taking the intensity of the first person shooter genre into the 2D world. The gameplay is enhanced with the use of rigid body physics to allow for more player interaction with the environment, giving the user a unique experience while exploring the world. In the story, a space traveler whose ship has ‘capsized’ on an unknown planet must avoid hostile alien inhabitants and rescue his stranded crewmates. To win, the player must finish 15 levels, all with different objectives, puzzles, enemies, and weapons.

To interact with the game, players use both the keyboard and mouse. Mimicking the traditional shooter control scheme, the mouse is used for aiming and firing and the keyboard is used for movement. Although the game can be played using a standard xBox controller, the evaluation presented here was performed on the PC version of the game. The game also features a traditional non-diegetic HUD system displaying the player’s current health, jet fuel, objective direction, and ammunition.



Figure 2: Gameplay screenshot from Capsized

4.2 A Basic Evaluation

Before the heuristic evaluation was conducted, Pinelle et al.'s heuristic principals were presented on the Capsized Play Testing & Development Forum. The beta version of the game was tested, representing a vertical slice of the entire game.

After the initial evaluation was conducted, one of the six (Table 2) severity classifications were assigned to each of the 28 identified usability problems. The problems ranged in severity from *minor* to *major*. No *Catastrophic* or *Game Breaking* problems were found. The majority of problems were classified as *moderate* or *minor* with a small portion being classified as *major*, and one classified as *not a problem* (Figure 3).

The initial evaluation was sufficient to identify the usability problems; however, the coarse severity ratings were insufficient for the needs of the developer. Thus, we applied our new critic-proofing technique to the evaluation results in a post-hoc manner.

4.3 Critic-Proofing Heuristic Approach

Before the new severity ratings could be applied to the usability problems, Capsized needed to be assessed using our pre-evaluation step. Based on feedback received from the developer, the game was classified into the action and shooter genres. Each genre was weighted at fifty percent and a weighted average was calculated providing the genre ratings for the game (Table 3).

Each of the previously identified heuristic problems was assigned

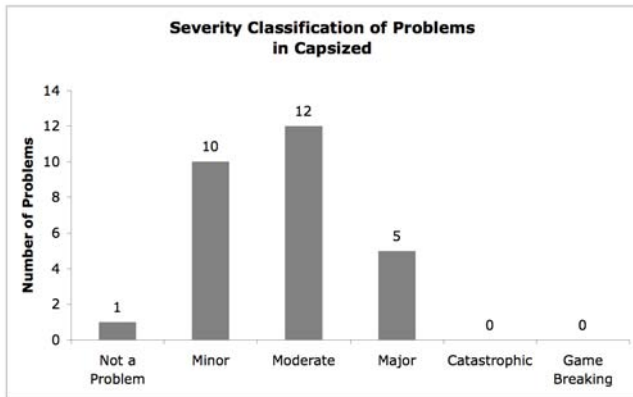


Figure 3: This graph shows the distribution of usability problems identified in Capsized, October 2009

Table 3: The calculated genre ratings for Capsized based on the Action and Shooter Genres.

| | Action | Shooter | Genre Rating |
|-------------------------|--------|---------|--------------|
| Consistency | 8 | 8 | 0.667 |
| Training & Help | 2 | 2 | 0.167 |
| Controls | 12 | 5 | 0.708 |
| Response Time | 7 | 4 | 0.458 |
| Game Status | 3 | 3 | 0.250 |
| Artificial Intelligence | 0 | 6 | 0.250 |
| View Mismatch | 8 | 1 | 0.375 |
| Command Sequences | 1 | 2 | 0.125 |
| Visual Representation | 1 | 1 | 0.083 |
| Input Mappings | 5 | 2 | 0.292 |
| Customizability | 3 | 1 | 0.167 |
| Skip Content | 0 | 0 | 0.000 |

to one of the 12 common usability problems, and we performed our new severity rating calculation using both the newly-calculated normalized genre ratings, and the normalized classification ratings. The technique produced prioritized ratings ranging from 0.684 to -0.167, with 18 unique classifications.

The increased granularity of severity made the organization of usability problems more obvious and provided the Capsized developers with a report outlining the most significant usability problems. Both heuristic evaluation results were presented, and feedback comparing the two techniques was collected.

4.4 Comparing Results

The initial severity classifications were done with text descriptions, (i.e. minor, moderate, major). To compare these descriptions directly with the prioritized ratings calculated in our technique we used the normalized numerical classification ratings found in Table 2.

We carried out a direct comparison between the results of the two techniques. In Figure 4, we present the normalized severity ratings for both techniques. The figure shows the severity ratings ordered by the prioritized rating in decreasing severity with the corresponding classification rating and genre rating that a specific problem received. The chart shows that some usability problems are reordered when genre and classification ratings are combined into the prioritized rating, creating a much clearer rating system.

Here are two examples of usability problems from the traditional evaluation:

1. *Description:* Grappling hook rope length uncontrollable
Heuristic violated: 7. Provide controls that are easy to

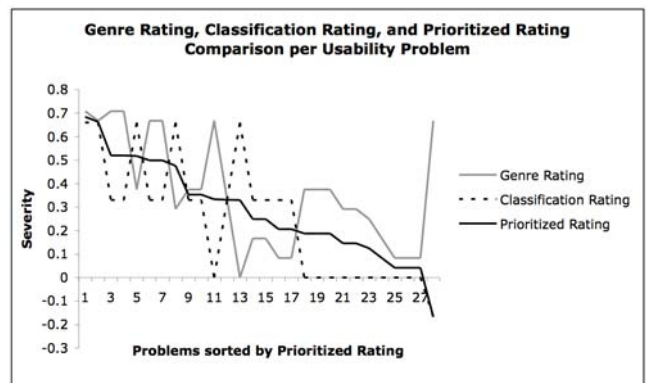


Figure 4: A comparison between severity ratings demonstrating the advantage of the prioritized rating.

manage, and that have an appropriate level of sensitivity and responsiveness.

Severity: moderate (The problem is persistent and occurs frequently, but has a low impact because it can be worked around)

2. *Description:* No option to skip tutorial or story text if you desire, info bubbles are difficult to avoid and cannot be sped up.
Heuristic violated: 5 Allow users to skip non-playable and frequently-repeated content.
Severity: major (This has a high impact, frequency, and persistence, especially when playing a level more than once.)

In this case the first problem is grouped with 11 other moderate problems and the second with 4 other major problems. Also the second problem is considered more severe than the first. If we examine the same problems evaluated using our technique:

1. *Description:* Grappling hook rope length uncontrollable
Heuristic violated: 7. Provide controls that are easy to manage, and that have an appropriate level of sensitivity and responsiveness.
Severity: 0.519 Controls moderate (The problem is persistent and occurs frequently, but has a low impact because it can be worked around)
2. *Description:* No option to skip tutorial or story text if you desire, info bubbles are difficult to avoid and cannot be sped up.
Heuristic violated: 5 Allow users to skip non-playable and frequently repeated content.
Severity: 0.330 Skip Content major (This has a high impact, frequency, and persistence, especially when playing a level more than once.)

Using our technique the moderate controls problem is rated as more severe because in game reviews controls were cited as a problem more often than skip content for action/shooters.

The results of the evaluation were presented to the Capsized development team in a report. Both the classification ratings and the prioritized ratings were presented in the report; however, the prioritized ratings were assigned a numerical value, while the classification ratings were presented in text. No problems were severe enough to be classified as either ‘*game breaking*’ or ‘*catastrophic*’. The 28 problems were grouped into one of the other four categories: five were ‘*major*’, twelve were ‘*moderate*’, ten were ‘*minor*’, and one was ‘*not a problem*’. Problems in the report were ordered from most severe to least severe. The second report contained a list of the same problems; however, they were prioritized using our technique. The prioritized severity ratings were multiplied by 100 to ensure ease of reading, but this has no bearing on the function of the rating system.

4.5 A Developers Reception

Play testing began in October 2009. This build represented 1/3 of the final game, and included most of the game mechanics. Before the evaluation was performed, the heuristic evaluation technique was described and presented to the designers. In response, one designer said, “The list of usability principles was a great read, I look forward to seeing your full review.”

The designer was questioned after he had examined both the traditional and critic-proofed heuristic analysis. When ask which of the two was more valuable he picked our prioritized list:

“I would say the priority list is more important.”

And:

“It’s more developer oriented and task based, and can be constantly updated and items moved around based on what needs priority.”

It was also clear that heuristic evaluation in general was considered to be something valuable that multiple developers could perform, thereby increasing the visibility of usability issues:

“It would be constantly updated and more developers would take notice of these issues.”

And:

“With this development-centric approach we will be able to solve these problems more effectively, and clearly identify more possible issues.”

We also drew a comparison to more traditional playtesting models [5]. The designer clearly felt that our technique provided greater benefit compared with traditional playtesting. Suggesting there would be value in allowing playtesters to perform the evaluations:

“Rather than just a description of what was ‘fun’ or ‘exciting’ (which is of course very valuable), the system clearly identified problems and their severity, making it more development-centric than the evaluation system in place.”

And:

“I would most likely base it on a wiki and host the priority system there once production stage is done and we are focusing on playtesting. So even playtesters could add to the system.”

5. DISCUSSION

The technique we have outlined is a valuable addition to the game-testing process. Our approach incorporates the advantages of traditional heuristic usability evaluation, such as cost effectiveness and small time investment, while also providing designers with a new empirically-grounded method to evaluate the severity of problems. Many inspection methods rely heavily on designer judgment and experience for prioritizing problems; our technique allows another avenue for validation. We do this by providing a metric-based approach to help determine where developer resources should be focused, based on critic reviews from prior games.

Play testing can be performed on a game as soon as it reaches a playable state. This testing is typically done in-house, or out sourced to a dedicated testing company. Actual players don’t test a game until it reaches a later stage of development, often because developers don’t want their game becoming public too early. If a game becomes available too early, players’ and critics’ perceptions about quality and content might be biased before the final product is released.

Although useful for companies of all size, our technique is particularly valuable for a small development team. The process is simple, and heuristic evaluations are quick and cheap. In addition, the organizing and prioritizing of problems that our technique provides helps to mitigate the development risk of limited corporate product memory for new independent studios and inexperienced.

Larger companies and publishers can also benefit from our technique. Our process can easily be added to any existing testing or development framework. By presenting heuristic feedback in

an organized and prioritized manner, we enable larger teams to integrate heuristic testing more tightly with project management systems and tools. Larger development teams can also benefit from the process in many of the same ways that smaller teams do – by performing usability tests cheaply and quickly at many points in the development process. The outsourcing of quality assurance (QA) is a common practice in established studios with publishing partnerships. QA is often performed by the publisher, and in many cases outsourced again. Our technique also fits well in this model, because, prior to being delivered to QA, our technique can quickly be performed by the developers.

6. CONCLUSION

We have presented a modification to traditional heuristic evaluation of games that does not require much effort, time or resources, but provides substantial additional value by creating a genre-specific prioritized list of usability problems. When you compare our technique to traditional play testing, the value becomes apparent. Our technique can be performed early in a project's development, is fast and cheap, and provides developer-centric results. Our technique also incorporates historic genre-specific usability issues collected from game reviews. The evaluation technique may be performed prior to release, but provides feedback based on similar games.

We validated our technique with a case study of a game currently in production. Feedback from developers suggests that the technique provides additional benefit over both traditional heuristic evaluations and standard play testing.

We describe our process as critic-proofing, because the genre ratings are grounded in an empirical analysis of critical reviews. As we described earlier, feedback from critics is extremely valuable in assessing user experience [10][6]. While the technique does not attempt to inform the designer of how to make a perfect game in the eyes of critics, it does incorporate critics' feedback to produce priority ratings of usability problems that can help focus development efforts.

7. FUTURE WORK

The technique that we have outlined is an important first step on the road to developing a more robust overall game testing approach based on user experience. Future studies will examine the use of heuristic evaluations for the validation of system architecture, and the integration of the technique into a higher-level design methodology, such as scrum. To further validate our technique, we will conduct an empirical study comparing our results to critics' reviews. Our study explored a limited set of heuristics focusing exclusively on usability. Future studies will explore the use of our technique with other heuristics exploring topics such as playability.

8. ACKNOWLEDGMENTS

We would like to say a big thank you to Lee Vermeulen who provided us with substantial feedback, even during development crunch time on the Capsized project. Without his input our evaluation would not have been possible, or nearly as successful.

9. REFERENCES

- [1] Desurvire, H., Caplan, M., & Toth, J. A., Using heuristics to evaluate the playability of games. *Ext. Abstracts of CHI 2004*, 1509-12.

- [2] Desurvire, H., & Wiberg, C.. Game Usability Heuristics (PLAY) For Evaluating and Designing Better Games: The Next Iteration. *Proc. of HCI Conference 2009*.
- [3] Dumas, J.S., & Redish, J.C., *A Practical Guide to Usability Testing*, Ablex Publishing Corporation, Norwood, 1993.
- [4] Federoff, M. *Heuristics and Usability Guidelines for the Creation and Evaluation of Fun in Video Games*. Indiana University Master of Science Thesis, 2002.
- [5] Fullerton, T., *Game Design Workshop, 2nd Edition: A Playcentric Approach to Creating Innovative Games*, Morgan Kaufmann Publishers, Boston, 2008.
- [6] Koeffel, C., Hochleitner, W., Leitner, J., Haller, M., Geven, A., & Tscheligi, M., Using Heuristics to Evaluate the Overall User Experience of Video Games and Advanced Interaction Games. in Bernhaupt, R. ed. *Evaluating User Experience in Games*. Springer, 2009.
- [7] Korhonen, H., Koivisto, E.M.I., Playability Heuristics for Mobile Games. *Proc. of MobileHCI 2006*, 9-16.
- [8] Korhonen, H., Koivisto, E.M.I., Playability Heuristics for Mobile Multi-Player Games. *Proc. of DIMEA 2007*, 28-35.
- [9] Korhonen, H., Paavilainen, J., & Saarenpää, H. Expert review method in game evaluations: comparison of two playability heuristic sets. *Proc. of MindTrek 2009*, 74-81.
- [10] Larsen JM (2008) Evaluating User Experience – how game reviewers do it. In: Evaluating User Experiences in Games, *Workshop at CHI 2008*.
- [11] Nacke, L. From playability to a hierarchical game usability model. *Proc. of FuturePlay 2009*, 11-12.
- [12] Nielsen, J. How to Conduct a Heuristic Evaluation (2007). Retrieved March 4, 2010, from UseIt: www.useit.com/papers/heuristic/heuristic_evaluation.html.
- [13] Nielsen, J. 1992. Reliability of severity estimates for usability problems found by heuristic evaluation. *Proc. CHI 1992*, 129-130.
- [14] Nielsen, J. Severity Ratings for Usability Problems (2007). Retrieved March 4, 2010, from UseIt: <http://www.useit.com/papers/heuristic/severityrating.html>.
- [15] Nielsen, J. Ten Usability Heuristics (2005). Retrieved March 4, 2010, from UseIt: www.useit.com/papers/heuristic/heuristic_list.html.
- [16] Nielsen, J., and Mack, R.L. *Usability Inspection Methods*, John Wiley & Sons, New York, 1994.
- [17] Nutt, C Back in Space: BioWare On Mass Effect 2. Retrieved March 4, 2010, from Gamasutra: www.gamasutra.com/view/feature/4251/back_in_space_bio_ware_on_mass_.php, 2010.
- [18] Pinelle, D., Wong, N., Stach, T., Heuristic Evaluation for Games: Usability Principles for Video Game Design. *Proc. of CHI 2008*, 1453-1462.
- [19] Pinelle, D., Wong, N., Stach, T., Gutwin, C. Usability Heuristics for Networked Multiplayer Games. *Proc. of GROUP 2009*, 169-178.
- [20] Pinelle, D., Wong, N., Stach, T., Using Genres to Customize Usability Evaluations of Video Games. *Proc. of FuturePlay 2008*, 129-136.