Heuristic Evaluation of the EvoGlimpse Video Game

Bianca-Cerasela-Zelia Blaga[1], Selma Evelyn Cătălina Goga[2],
Al-doori Rami Watheq Yaseen[3], Dorian Gorgan[4]
Technical University of Cluj-Napoca
Computer Science Department
Cluj-Napoca, Romania
[1]zelia.blaga@cs.utcluj.ro, [2]selma.goga@cs.utcluj.ro,
[3]ramy@uob.edu.iq, [4]dorian.gorgan@cs.utcluj.ro

ABSTRACT
The evaluation of the interface of a video game is essential for its development. In this paper, a heuristic evaluation is proposed, from the perspective of an interactive application. The goal is to estimate the level of usability. The game is tested by evaluators who follow a series of scenarios and relevant actions, with the purpose of answering questions that can determine if it respects the usability requirements. Specific evaluation criteria are established, and solutions are proposed for the found problems.

Author Keywords
evaluation criteria; heuristic evaluation; interactive applications; video games; usability.

ACM Classification Keywords
empirical studies; HCI design and evaluation methods; interactive games.

INTRODUCTION

Video games are a very popular type of interactive application, with a large number of objectives. For example, they can be used in educational purposes [1], to offer useful information to the user, in an enjoyable way. The games are also used because they are recreational and a preferred pastime for children and teenagers. They can help develop fast problem-solving skills, with applicability in real life too. This can be better observed in the case of strategy games, where the player has to use the existing environment, resources, and characters to efficiently win the game.

The main motivation behind the concept of usability of an interactive application, so implicitly of video games, relies on its capacity of establishing the success or failure rate of a software product. Therefore, numerous companies have strict evaluation criteria, with some of them presented in this paper. The evaluation of usability can be done during the implementation and development stages of a game, which is highly recommended. Evaluation is an iterative process, by intercalating it in the stages of developing a video game, and because it has the advantage of highlighting the design and implementation flaws, errors and specific deficiencies, which can only be observed during testing. There are various evaluation methods, but in general, they are done by testing some scenarios with necessary actions that need to be done when executing the project, establishing what outputs are expected, what needs to be avoided, the execution speed, and also the number of errors. These methods of estimating the usability are preferred by developers and are done by experts in the domain, which are familiar with such systems.

Figure 1. The game interface of EvoGlimpse, whose usability will be evaluated

The evaluators are focusing on finding the problems and creating thorough and helpful reports. Usability can be established by cognitive or pluralist evaluations, inspecting the consistency, standards, and characteristics of the system or by heuristic evaluation [3]. In the current paper, they are combined in order to reach solutions to solve the problems which are found.

The interactive application that will be evaluated in this paper is the game called EvoGlimpse. It started with the aim to give to players a glimpse into evolution from the perspective of an exterior observer, who can travel at different points in time of Earth's existence. This game is heavily inspired by the movie and the book „2001: A space odyssey" [2], in which a civilization of advanced beings helps humans that are in different stages of evolution by presenting to them ways that can aid in their survival.

A series of worlds would be available, starting from the first appearance of life – the fusion between RNA and an enzyme, then at different stages of the evolution of species – underwater life, transitioning to land and dinosaurs, moving on to the human history – from the ancestors until today, and for a plus of entertainment, will continue with a science fiction view of humankind – the union of human-machine and the exploration of the universe. The player would be able to travel in these worlds in different specific shapes: atoms, energy, swimming, walking, riding animals,

driving the cars, flying with the flying cars, and exploring space in spaceships.

Each stage has as objective finding the knowledge source, represented by the monolith, which has an imposing shape, tall, black, created by a superior entity and which holds superior information about the current state of the world. For example, in the stone age, this can offer to the monkeys the idea of creating weapons that represent an advantage in the fight for survival.

As a world is explored, different obstacles appear, and the player must overcome them with the current set of skills. This is enhanced each time the monolith is found. Once the world has been completely observed and the enemies are defeated, the monolith appears to present the way of going from the past to the future. Using visual and auditory information, the player will know if he/she is close to the location of the monolith, and when this will be found, an educational video about evolution will be presented. The player will also be able to see all finished phases and all the discovered videos in a library, to which he/she can return at any time.

For the actual game implementation, the goal was to create only a world, a futuristic one, on a planet covered by water, in a developed society, with modern architecture and flying cars. The main enemies will be planes guided by artificial intelligence. The player will have to protect itself from them by shooting, for example with bullets, plasma or laser. The main plot of the game follows 3 stages. In the first one, the player will have some time to get used to the planet and the controls, being able to peacefully explore and observe the world scene. In the second stage, the player will have to protect the planet from some invaders; as the game advances, the abilities of the player increase. In the last stage, since an advanced technology state has been reached, the monolith will appear in an unknown location and will have to be found by following its sound signals. An in-game image can be seen in Figure 1. Here there can be observed the game scene composed of water, building and a separator ring, the player's vehicle, and the dynamic object with which the player will interact (enemies and power-up boxes).

We want to heuristically evaluate this game, which is a technique that helps determine the usability problems of a user interface. This is done by a small number of evaluators (two), using a specific set of heuristics, proposed by the developer. Afterward, the evaluation results are centralized, and the noticed problems are marked out, and solutions are proposed. The chosen criteria come from the 10 heuristics of Nielsen [4]: the visibility of system status, match between system and the real world, user control and freedom, consistency and standards, error prevention, recognition rather than recall, flexibility and efficiency of use, aesthetic and minimalist design, help users recognize, diagnose, and recover from errors, and help and documentation.

This paper is structured as follows: in Section **Related Work** will be presented a literature review of this domain, together with some evaluation methods. In Section **Theoretical Considerations**, the exact methodology that was taken into account for the heuristic evaluation will be explained. In Section **Experimental Considerations**, the stages of the evaluation are discussed, and the observations are explained; there are presented the requirements, the evaluators, the heuristic evaluation details, the scenarios and the tasks that need to be followed for testing, and the means of recording the results. Then, in Section **Result Analysis**, the outcomes are analyzed after the independent and group evaluations, the errors discovered are highlighted and solutions are proposed. The final observations are written in Section **Conclusions**.

RELATED WORK

For the evaluation of the usability of an interactive application, there are various methods, each one specific to the type of application, and the main goals of its developers. In general, there are used usability questionnaires like SUMI [5] or QUIS [6], from which standard information from the domain of usability can be extracted.

In [7], a series of steps are defined for evaluating the usability: data gathering – by collecting information related to how the application should be used, data analysis – summarizing the statistics that were done and pointing out the flaws and coming up with ways of improving them.

In virtual reality applications, for example, there are 6 stages [8]: the exploratory one – where similar applications are analyzed and bibliographic material, related to the domain and the evaluation heuristics, is collected, the descriptive one – where the conclusions from the first stage are synthetized, and specific evaluations are formalized, the correlative one – where the principal characteristics of the usability heuristics are identified, and representative case studies are presented, the explanatory one – where the heuristics are established following five characteristics (identifiers, explanation, example, benefits, and problems), the validation one – where the evaluators inspect the application based on the previously mentioned heuristics, and the refinement one – after which three types of problems are found and need to be solved.

The developer is the one who proposes game scenarios, and him/her describes how these can be done by the evaluators,

Table 1. The developer and the usability evaluators

Name	Specialization and year of study	Domain
Developer	Artificial Vision and Intelligence 1st year master's student	Researcher in the image processing group; medium experience with video games
Eval1	Artificial Vision and Intelligence 1st year master's student	Researcher in the image processing group; little experience with video games
Eval2	Artificial Vision and Intelligence 1st year master's student	Experience in designing interactive applications; medium experience with video games

as the execution of specific actions. Thus, there can be observed how these actions can be done, how easily they are understood, their difficulty level and the differences between the expectations and the actual implementation can be seen. An evaluator has to test the game while keeping in mind the requirements and will write reports which will point out the discovered flaws.

Another evaluation method is AOP (Aspect Oriented Programming) [9], a recent technique with satisfying results, and which is easy to use. In [10], the same authors propose the use of agents that can automatically do the evaluation. Based on an initial set of knowledge, they have the capacity to learn how to use the environment in which they are placed and know what tasks to execute.

The heuristic evaluation proposed by Nielsen [4] asks the evaluators to establish the usability level based on 10 criteria. This is done by a small number of evaluators, based on a detailed set of scenarios and materials. The tasks have to be executed twice on the application's interface, with each element being inspected (button, object, control element etc.), followed by the evaluation of the implementation techniques and the interaction with them. The main goal is to find design and implementation errors and solutions to them.

THEORETICAL CONSIDERATIONS

Usability is defined by Shackel [11, 12] as the capacity of a system to be easily understood and efficient to use by a specific category of users, which received instructions and assistance in the usage of the application, by executing tasks defined for a system. The emphasis is on efficiency, ease of learning, flexibility, and attitude. A similar way of defining usability is the one devised by Preece [13], which measures it as the ease with which a system can be used, together with its efficiency and security.

In the ISO 9241-11 standard [14], usability is defined as being: "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use". Efficiency is the ratio between the used resources and the accuracy with which they can be used, efficacy is the accuracy and correctness of the system, while satisfaction is a more subjective measure that refers to the user's comfort.

From Dix's perspective [15], usability depends on three factors: the ease of learning – how fast can the new users use the system correctly and at a high level of performance, flexibility – how easy it is to use the controls, together with their correctness and robustness – the help that the user has to fulfill the specific actions. These represent a starting point for creating evaluation tools.

Usability evaluation has three main objectives, which are highly correlated to the previously mentioned factors:

1. establishing the degree of functionality of the interactive application;
2. assessing the suitability of the interaction of the user with the interface;
3. identifying the system's problems.

Table 2. Scenarios and actions that will be executed by the evaluators to test the game

Scenario	Actions
S1. Navigation in the 3D scene	T1. controlling the vehicle using the mouse movements T2. increase speed by pressing space T3. zoom in and out using the scroll wheel
S2. Attacking and avoiding enemies	T1. observing the enemies T2. flying towards enemy T3. player attacks by pressing the left button of the mouse T4. the enemies attack when the player gets in a certain range and in a certain field of view T5. observing the enemies reaction T6. avoiding enemies
S3. Monolith	T1. the player should understand the objective, by reading the message shown on the screen T2. successfully navigating in the scene T3. observe the monolith T4. fly towards objective T5. message of winning the game
S4. Repair power-up box	T1. recognizing the object T2. flight towards the objective T3. colission with the object T4. object destroyed T5. life health increased
S5. Immunity power-up box	T1. recognizing the object T2. flight towards the objective T3. colission with the object T4. object destroyed T5. enemy attack canceled for 20 seconds
S6. Display relevant messages	T1. message with the game objectives T2. toggle help option T3. quit button T4. player health information T5. message of collecting repair power-up box T6. message of collecting immunity power-up box T7. message of destroying enemy T8. message of losing the game T9. message of winning the game

Functionality refers to the degree of correctness the implementation of the application has, while the interface is what the user sees and a way of sending inputs and getting outputs. It has a big impact, especially in video games, because the interaction is more visual and based on metaphors specific to the game genre. On it depends the ease of learning and the usage flexibility, but also the ability to recognize not recall, which doesn't load the memory of the user with too much information.

The planning is done together with the evaluators, after an implementation phase of the system. The used concepts are defined in order to avoid misunderstandings. Afterward, a

series of criteria are defined, which are clear and specific to the evaluated interactive application. Then the evaluation is done based on them, highlighting the errors, and finally, the results are evaluated, and solutions are proposed to improve the system. In the next section, these steps will be shown on the game EvoGlimpse.

EXPERIMENTAL CONSIDERATIONS

In the heuristic evaluation done on the proposed video game, the main goal was to find the implementation errors of the proposed scenarios. First, the evaluators were chosen, whose information can be seen in Table 1. Next, six usage scenarios have been set, which contain the game scene navigation by controlling the player's vehicle using the mouse, interaction with the enemies by attacking them, collecting the power-up boxes etc. Each scenario can be executed by following a set of tasks, which result in feedback from the system, and that can be instantly seen by the user. This information is contained in Table 2.

Afterward, the evaluation criteria were established. Nielson's 10 usability heuristics for user interface design were chosen [4], with supplementary explanations that will

Table 3. Evaluation criteria

Nb.	Questions and requirements
1.	**Visibility of system status** • Is the state of the system visible at all times? • Is the feedback offered by the system suitable? • Is the response time appropriate, without unacceptable delays? • The game scene will be observed, as well as the interaction with the objects and elements specific to each game scenario; attention will be payed to movement of the vehicle, attack, collection of the power-ups, the display of messages and particle effects.
2.	**Match between system and the real world** • Does the game correspond to the mental model that the user has from a real-world game? Is it what you expected or similar to other games? • Are the language, words, and phrases used familiar to the user? • Is there a natural way in displaying the information? • Is this a suitable shooter game? Is the game scene realistic? • Are there any uncertainties?
3.	**User control and freedom** • Can the user execute the necessary actions to fulfill the scenarios? Is their functioning correct? • Can the user exit an unwanted state? For example, is there a need for an undo/ redo button? • How does the vehicle control, attack, collection, and buttons feel?
4.	**Consistency and standards** • Is the user surprised by different words, situations or actions that have the same meaning? • Is there consistency in the use of colors and symbols? • Is the meaning of the objects from the scene understood?
5.	**Error prevention** • What is the functional correctness level of the game? • Are the errors eliminated or are there methods to prevent situations that favor the apparition of errors? • For example, notice what happens if the player tries to get too close to the water, at the collision with different objects etc.
6.	**Recognition rather than recall** • Can the player recognize the objects and their usage? • Are there elements that require storage in the memory of the user?
7.	**Flexibility and efficiency of use** • What is the level of flexibility and efficiency of the game usage? • Is the user bothered by certain aspects? Or are some of them missing?
8.	**Aesthetic and minimalist design** • What is the quantity of relevant information? • Is there any redundant information? • Is the information presented clear and easily accessible? • Is the field of view of the player cluttered with too many elements or is it suitable?
9.	**Help users recognize, diagnose, and recover from errors** • Are the messages clear and helpful for the player? • Should there be any additional error prevention cases?
10.	**Help and documentation** • Is the help menu complete? • Does it contain clear, simple, and easily accessible information? • Is the documentation clear, does it contain sufficient information for the player? If not, what should be added?

Table 4. Evaluation stages

Nb.	Name	Evaluation technique
1.	Individual evaluation	• done independently by the 2 evaluators, by filling in separate tables for each scenario • a mark between 0 and 100 is assigned to each evaluation criteria, and at the end the average is taken • at the end of testing, reports are written with the encountered problems • the developer proposes solutions to solve the errors
2.	Group evaluation	• done by the 2 evaluators together with the game developer • tables with the most important questions are written, together with the found answers

Table 5. Individual heuristic evaluation results of the first scenario

Scenario	Criteria	Eval1	Eval2	Average
S1. Navigation in the 3D scene	1. Visibility of system status	100	100	100
	2. Match between system and the real world	90	90	90
	3. User control and freedom	70	80	75
	4. Consistency and standards	100	90	95
	5. Error prevention	50	90	70
	6. Recognition rather than recall	90	100	95
	7. Flexibility and efficiency of use	100	90	95
	8. Aesthetic and minimalist design	100	100	100
	9. Help users recognize, diagnose, and recover from errors	50	95	72.5
	10. Help and documentation	80	100	90
Eval1's report	The collision with objects such as buildings is an enormous problem, as you have probably observed. After colliding with a building, I was simply floating in space, without being able to reposition myself. I know why this is happening, I have the same issue in my game, but an inexperienced user will not understand a thing.			
Eval2's report	The game starts abruptly, without a start menu, but the movements of the vehicle are very smooth. It is easier to move left-right than up-down.			
Solutions	There is a problem at the level of materials that are attached to the objects, in particular to the vehicle and the buildings. This can be solved by changing the bounce value in the phycis property of the materials. The creation of a menu will be taken into consideration for the next implementation iteration.			

Table 6. Individual heuristic evaluation results of the second scenario

Scenario	Criteria	Eval1	Eval2	Average
S2. Attacking and avoiding enemies	1. Visibility of system status	70	100	85
	2. Match between system and the real world	20	95	57.5
	3. User control and freedom	90	80	85
	4. Consistency and standards	100	90	95
	5. Error prevention	90	80	85
	6. Recognition rather than recall	100	100	100
	7. Flexibility and efficiency of use	80	90	85
	8. Aesthetic and minimalist design	100	100	100
	9. Help users recognize, diagnose, and recover from errors	90	80	85
	10. Help and documentation	100	100	100
Eval1's report	The match between the virtual world of the game and the real world has such a small mark because it is not very easy to see when someone is shooting you or when you are attacking someone. I was expecting to see a laser or a bulltet that would appear. Instictually I want to get close to attack objects because I know that a bullet shot at a closer distance is more accurate that one shot at a higher distance. Here it does not matter. Another severe issue is that it is too easy to destroy an enemy. It would have been useful to add a life-bar on top of each one, and to be necessary at least 2-3 shots to take down an object. When an enemy shots you, there is not enough information. You expect to see a particle effect on the car or at least to hear a specific sound. That is why I gave it only 20 points.			
Eval2's report	The enemies are easy to attack and avoid, but their answer is too slow.			
Solutions	The enemies have attack particle effects, but those can not be observed since they are behind the player. This can be changed by adding effects on the car, and adding sounds that would help the player know if he / she is shot. Also, the user attacks in the center of the screen, where the crosshair is displayed. The player should experiment with the attacks, and thus it can be seen that the enemies can be shot only at a certain distance. The enemies do not die instantly, as it can be seen on the particles displayed on the player's vehicle, multiple shots are needed. A health bar should be added to the enemies to aid in this problem. Also, the enemies only attack if the user is at a certain distance from them, and in a certain field of view. To make their response faster, I can increase their movement speed.			

Table 7. Individual heuristic evaluation results of the third scenario

Scenario	Criteria	Eval1	Eval2	Average
S3. Monolith	1. Visibility of system status	90	100	95
	2. Match between system and the real world	100	90	95
	3. User control and freedom	100	100	100
	4. Consistency and standards	90	80	85
	5. Error prevention	100	90	95
	6. Recognition rather than recall	60	100	80
	7. Flexibility and efficiency of use	100	100	100
	8. Aesthetic and minimalist design	90	100	95
	9. Help users recognize, diagnose, and recover from errors	100	100	100
	10. Help and documentation	100	100	100
Eval1's report	Being just a prototype version of the game, it is alright to put the monolith always in the same place, but I admit it would have been more fun to compute its position randomly at each run of the game so I wouldn't know where it is when a new game begins.			
Eval2's report	At the first run of the game, I destroyed all the enemies and I collected all the power-up boxes, and afterwards I found the monolith and the game stoped. A little too repetitive.			
Solutions	I chose the option of fixing the position of the monolith because the game scene is small and the objective would have been too easy to find. If the scene was bigger, then yes, the position of the monolith would be randomly computed at each run of the game. The same is available for the enemies and the power-up boxes - if the game scene is bigger, more objects can be inserted, thus making the game more entertaining.			

Table 8. Individual heuristic evaluation results of the fourth scenario

Scenario	Criteria	Eval1	Eval2	Average
S4. Repair power-up box	1. Visibility of system status	100	100	100
	2. Match between system and the real world	90	100	95
	3. User control and freedom	100	90	95
	4. Consistency and standards	100	100	100
	5. Error prevention	100	90	95
	6. Recognition rather than recall	80	100	90
	7. Flexibility and efficiency of use	100	100	100
	8. Aesthetic and minimalist design	100	100	100
	9. Help users recognize, diagnose, and recover from errors	100	90	95
	10. Help and documentation	100	100	100
Eval1's report	These look nice, but I expected them to disspear before passing through them. I do not notice if the disappear from the game scene for example, because it is difficult to turn the vehicle around.			
Eval2's report	It is a very good game object, but not always necessary, especially because the enemies do not represent a big threat.			
Solutions	I choose the option of making the boxes dissapear after the collision because it would have been confusing otherwise. It can be noticed that they do dissapear instantly after we touch them, and the interaction with them is correct since their effect is immediately observed and a feedback in the form of a system message is displayed. Their necessity can be increased by adding different abilites to the enemies or making them smarter.			

Table 9. Individual heuristic evaluation results of the fifth scenario

Scenario	Criteria	Eval1	Eval2	Average
S5. Immunity power-up box	1. Visibility of system status	100	100	100
	2. Match between system and the real world	90	100	95
	3. User control and freedom	100	90	95
	4. Consistency and standards	100	100	100
	5. Error prevention	100	90	95
	6. Recognition rather than recall	80	100	90
	7. Flexibility and efficiency of use	100	100	100
	8. Aesthetic and minimalist design	100	100	100
	9. Help users recognize, diagnose, and recover from errors	100	90	95
	10. Help and documentation	100	100	100
Eval1's report	Same observation as above. These look really nice, I love the graphics. The concentric circles look really good.			
Eval2's report	I think more enemies are needed in order to increase the game difficulty.			
Solutions	I admit that I focused more on the functional correctness of the game rather than on the level of entertainment. This can be changed by increasing the game scene and adding variety to the enemies.			

Table 10. Individual heuristic evaluation results of the sixth scenario

Scenario	Criteria	Eval1	Eval2	Average
S6. Display relevant messages	1. Visibility of system status	100	100	100
	2. Match between system and the real world	80	90	85
	3. User control and freedom	100	90	95
	4. Consistency and standards	90	80	85
	5. Error prevention	50	90	70
	6. Recognition rather than recall	70	100	85
	7. Flexibility and efficiency of use	100	90	95
	8. Aesthetic and minimalist design	100	100	100
	9. Help users recognize, diagnose, and recover from errors	100	100	100
	10. Help and documentation	100	100	100
Eval1's report	I took points for error prevention because the user should receive a feedback when he / she is colliding with the buildings or a message should be displayed with how to solve this issue or something similar.			
Eval2's report	The messages are clear and correctly displayed			
Solutions	I think the best solution is to do error prevention at the collision with the buildings, so the user won't have to worry about it.			

Table 11. Group heuristic evaluation results – first part

Scenario	Eval1	Eval2
S1	**Q1**: The problem of the collision with the building is pretty serious. Did you think how you could solve it? **A1**: Yes, by altering the paramethers of the physics materials from which the objects are made. These have a bounce value that determines how they react when the collision takes place. **Q2**: The navigation using the mouse is very good, that's all I'm going to say. Much better than before, it is even fun to play. **A2**: Indeed, I took into consideration your early evaluations where you have observed that controlling the vehicle by the keys W, A, S, D is not intuitive, so I worked to improve it.	**Q1**: The vehicle now moves with the help of the mouse, not with the keys W, A, S, D? **A1**: Indeed, I modified this interaction to increase the usability of the game.
S2	**Q1**: I have noticed that there is no indicator saying that you hit an enemy or if you were hit, other than a message that I do not always have enought time to read. Is there a minimum distance needed to shoot an enemy? Did you think of using bullets, sounds and a life bar? **A1**: There is a certain distance that both the player's and the enemies attack can take place. If you are too far, the attack won't have any effect. I want to add power-up boxes with new types of attacks (laser, plasma etc.) and to add more useful effects to them, like particles, rays and sounds. And also the enemies should receive a health bar to aid the player.	**Q1**: It is too easy to take down enemies, and they rarely attack you. How can you improve these aspects? **A1**: I can change the life amount of enemies, the damage of both the player and the enemies, and the speed for the later, so they would move faster.
S3	**Q1**: Did you think to randomly compute the monolith's position randomly with each new game? Then I could really say from the beginning of the game that it is true that you "find the monolith" and not "recall where you last saw the monolith". **A1**: Yes, I did think about it, but because the game scene is too small, I had to position it at the furthest and most difficult to see position. Otherwise, there would have been the risk of it to appear right next to the player and the game would have ended instantly.	**Q1**: I was expecting the monolith to be more colourful and smaller, now it is just a black object. We can not differentiate between it and the buildings from far away. What is its purpose? **A1**: Actually, I am really happy that it is harder to notice, and I am glad that it is a correct representation from the monolith in „2001: A Space Odyssey" [2], it even respects the 1:4:9 proportions. I left it bigger to help in testing, but I can scale it to a smaller size. In the broader picture of the game, it has the purpose of displaying an educational video to the player, so it would cease to be just a big black box. **Q2**: The monolith is in the same place all the time and I can just avoid the enemies and fly directly towards it to win the game. How can you change this? **A2**: For the first issue, see answer to S3, Q1, Eval1. For the second one, I can make the monolith appear only after the player has destroyed all the enemies in the game, by setting it as active at a random position on the scene.

Table 12. Group heuristic evaluation results – second part

Scenario	Eval1	Eval2
S4	**Q1**: Do you think it is possible to identify the moment when a player entered too much in collision with a building and lost control, and therefore restart the game or display a relevant message? A1: Yes, I can write a method that would check when it enters in collision with an object and have a certain functionality - restart or display message. But I believe that in this case it is better to prevent this error.	**Q1**: Why is it so big and easy to obtain? **A1**: So it can be noticed from far away. I do not think it should be difficult to obtain. Inded, right now we can't see its true importance, but if more enemies were present, and you had a limited amount of resources (for eg amo), the player would have to pick the right time to use each power-up box. This would bring the game more on the strategy type.
S5	-	**Q1**: Why is it so big and easy to obtain? Also, did you think of informing the player when it expires? **A1**: Same as before, with the addition that For now the player has no information when the power-up expires, so a suitable message should be added when this happens.
S6	-	-

aid the evaluators to focus on the desired elements. Each person who tests the game will complete a table for each scenario, in which he/she will give a mark between 0 and100 for each of the 10 heuristics, and then will write a report with the found problems. These aspects are detailed in Table 3 and Table 4. There are two stages, one where the evaluation is done independently by each person, and one where it is done together with the developer to discuss the found issues directly on the game and to assess how suitable are the proposed solutions.

RESULT ANALYSIS

In this section, tables that contain the usability evaluation will be presented. The results of the evaluation were gathered in tables to keep track of the scenario that is tested, together with the problems it contains, and specific solutions that are proposed to remove the errors. Table 5 to Table 10 contain the results of the individual evaluation for each proposed scenario, together with a report on the discovered errors, and also with the solutions found by the developer. In Table 11 and Table 12, the most important questions that were asked during the group evaluation are taken apart and answered. Thus, we have successfully identified the drawbacks of the user interface design and implementation, and we were able to find solutions to correct them.

After the two evaluations, both evaluators found different types of errors and problems. Mostly, it had to do with functionality flaws, related to the interaction between the player's vehicle and the buildings. It was also noted that the user is not fully satisfied with the game, because it is repetitive and does not bring a lot of excitement with the low variety of tasks it had to do, and with the objects it has to interact with. After the heuristic evaluation, we reached the conclusion that the game has a level of usability of 92.6%.

CONCLUSIONS

This paper focused on heuristically evaluating a video game. The two evaluators had to execute a set of specific scenarios, to follow a group of tasks, and to write down reports with the problems that they found while testing the interactive application. Two main types of tables resulted, one containing the individual evaluations, and one with the group evaluation discussions. For each mentioned error or deficiency of the game, solutions were proposed by the game developer. It has been found that the game has a high level of usability. This evaluation represents a big help for a creator of interactive application because it is an extremely helpful way of finding in a fast and efficient way what the problems are, which speeds up the process of improving the system.

REFERENCES

[1] C. Pribeanu, D. D. Iordache, V. Lamanauskas, R. Vilkonis, "Evaluarea utilizabilităţii şi eficacităţii pedagogice a unui scenariu de învăţare bazat pe realitate îmbogăţită," presented at the Conferinta Nationala de Interactiune Om-Calculator - RoCHI, 2008.

[2] A. C. Clarke, S. Kubrick, *2001: a space odyssey*, 1968.

[3] E. d. Kock, J. v. Biljon, and M. Pretorius, "Usability evaluation methods: mind the gaps," presented at the Proceedings of the 2009 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists, 2009.

[4] J. Nielsen, *Usability Engineering*: Morgan Kaufmann Publishers Inc., 1993.

[5] K. Jurek and C. Mary, "SUMI: the Software Usability Measurement Inventory," *British Journal of Educational Technology*, vol. 24, pp. 210-212, 1993.

[6] J. P. Chin, V. A. Diehl, and K. L. Norman, "Development of an instrument measuring user satisfaction of the human-computer interface," presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 1988.

[7] M. Y. Ivory and M. A. Hearst, "The state of the art in automating usability evaluation of user interfaces," *ACM Comput. Surv.*, vol. 33, pp. 470-516, 2001.

[8] D. Gorgan, C. Rusu, D. Mihon, V. Colceriu, S. Roncagliolo, V. Rusu, "Euristici specifice de utilizabilitate pentru aplicaţiile paralele şi distribuite," presented at the Revista Română de Interacţiune Om-Calculator, Vol.4, Nr.2, 2011.

[9] A. M. Tarta and G. S. Moldovan, "Automatic Usability Evaluation Using AOP," *2006 IEEE International Conference on Automation, Quality and Testing, Robotics*, vol. 2, pp. 84-89, 2006.

[10] A. M. Tarta, G. S. Moldovan, G. Serban, " An Agent Based User Interface Evaluation Using Aspect Oriented Programming Techniques," presented at the ICAM5, 2006.

[11] B. Shackel, "Usability - context, framework, definition, design and evaluation," in *Human factors for informatics usability*, Cambridge University Press, 1991, pp. 21-37.

[12] B. Shackel, "Usability - Context, framework, definition, design and evaluation," *Interact. Comput.*, vol. 21, pp. 339-346, 2009.

[13] B. D. Preece J., Davies G., Keller G., RogersY, "A Guide to Usability," 1990.

[14] ISO9214-11, "Ergonomic Requirements for office Work with VDT's – Guidance on Usability," 1991.

[15] A. Dix, J. E. Finlay, G. D. Abowd, and R. Beale, *Human-Computer Interaction (3rd Edition)*: Prentice-Hall, Inc., 2003.