Towards a Game Design Patterns Suggestion Tool

The documentation of a computerized textual analysis experiment

Marcos S. O. Almeida, Lucio G. Valentin, Rogério A. Gonçalves
Computer Science Coordination
Federal University of Technology – Paraná (UTFPR)
Campo Mourão, PR, Brazil
{marcossilvano, lgvalentin, rogerioag}@utfpr.edu.br

Flávio S. C. da Silva
Computer Science Department
University of São Paulo (USP)
São Paulo, SP, Brazil
fcs@ime.usp.br

Abstract—**Throughout the last decade, designers and researchers have made attempts to bring improvements to the game design process through proposals of tools and methods. Some studies have focused on the analysis of design elements of popular games in order to understand the factors that may have contributed to their commercial and critical success. Among them, the Game Design Patterns project has been discussed in a number of publications. However, the identification of the patterns of a game is a long term process that must be entirely done by designers. No attempts of using computer tools to aid this process were documented. In this context, this paper presents an experimentation of use of textual analysis techniques to suggest patterns that may be contained in games. The results show that, while not mature enough to be used in real world scenarios, its achievements were positive and have room for improvements.**

Keywords—game design; game design patterns; similarity analysis applied

I. INTRODUCTION

The primary task of game designers can be synthesized as the creation of successful games, whether commercial or critical. Throughout the last decade, designers and researchers have made an attempt to bring improvements to the game design area by proposing new tools and methods. Their objective is to help designers to create better games. Most of such efforts are summarized in the work of [1]. Some of these attempts have focused on recognizing the recurring characteristics of the design of existing games in order to build a structured collection of reusable design concepts. In such works, there is a belief that the analysis of the parts of successful games may help in the creation of new ones, by reusing them. Among these works, the Game Design Patterns (GDP) approach [2] seeks to extract and document recurring patterns of game design in order to provide a framework for analysis and design of games.

The identification of which patterns are used in a game is a long term process that must be entirely done by designers. Currently there are almost four hundred patterns documented in the project wiki[1] but more may be added. What if designers would have to read thousands of patterns before deciding which ones could benefit their project? To facilitate this process, designers could do opposite: take a known game they judge to have interesting features for their project and then, from the collection of patterns for this game, select the desired ones. But to achieve this, we would need a way to suggest which patterns were contained in the game chosen by the designer. No attempts of using computer tools to aid this process were documented. On the other hand, computer textual analysis techniques are frequently used to check the similarity between contents. They are often applied in content suggestion for users in web based services for movies, music and games rental and acquisition. In this context, would be feasible to use these techniques to help designers to identify the patterns used in a game? We believe that is possible to analyze a game description and match it with the patterns in order to discover which ones are used. Throughout this text, we will present an experiment towards this effort.

In this paper we present the initial experiments with computerized textual analysis techniques towards a game design patterns suggestion system. Firstly we present a brief conceptual background of the research field. Next, we describe the textual analysis method implemented and a description of the tool tailored for the experiment. Finally, we discuss the preliminary results obtained.

II. CONCEPTUAL BACKGROUND

Since the early years of game designing, the Game Design Document (GDD) has been widely adopted as the mainstream tool in the industry [3]. However, some authors have pointed a considerable lack of formalization and standardization in the document [1]. They agree that, despite the visible success of the industry, the lack of game design tools hinders the progression of the design process as currently known.

In 1994, Costikyan [4] started a discussion about the need for greater formalism on game design. He suggested the build of a common design vocabulary as a framework that could allow designers to analyze and describe games. According to him, designers should have a way to dissect games, to understand the elements that compose them and to identify those elements that benefit or harm them. His speech has been echoed by many authors that have followed the same research

[1] http://gdp2.tii.se/index.php/Main_Page (visited on 2013/06/24)

field: the constitution of a framework for the analysis and design of games through a collection of reusable design concepts. Among then, the Game Design Patterns (GDP) project [5], seeks to extract recurring design concepts and to structure them as design patterns, much like in software engineering [6]. The project has achieved a collection of almost four hundred patterns, currently documented in the project wiki, becoming the most active work in its field of research.

A. *Related Work*

The game design patterns approach has been subject of analysis and discussion in a number of publications by many authors, which are related in the GDP project wiki. Some of these works have applied the GDP framework to analyze and compare existing games and design concepts.

Loh and Soon [7] applied the GDP framework to analyze and compare the fundamental concepts presented in some board and computer games through the patterns they present. The patterns were identified in the selected games by a reverse engineering approach, in which analysts did observations and experiments over the games. Although a common practice for analyzing games, the resulting set of patterns from the reverse engineering may vary if the analysis is performed by distinct designers, which directly relates to the different interpretations of each designer.

To help addressing this bias, Tolmie, DiPaola and Charles [8] proposed to represent the set of patterns of a game as a graph structure, which graphically highlights the inter-relations between patterns and allow a better overview of a game's patterns, thus facilitating its comprehension. As a result of their work, they have built a software application for interactive visualization of the GDP graph structure which is generated from a previously selected set of patterns. The application does not automatically identify the patterns from the game description: this must be done by the designers.

Samarnggoon [9] tried to diminish the bias resulting from identification of a game's patterns by using graph mining techniques. As an experiment, he used the board game Monopoly as a study case and asked ten designers to create a GDP graph for the game through manual reverse engineering. Then, he took the ten different graph versions and, using graph mining techniques, he converged then into one single version that would represent a consensus between the different visions of the designers.

In the context of those two works, we intent to provide an input to the graph structure that represents GDP map of a game. More specifically, we want to aid the designers in the identification of which patterns are used in a game, a process currently done manually, by providing a suggestion tool based on textual similarity analysis. Fig. 1 illustrates this process.

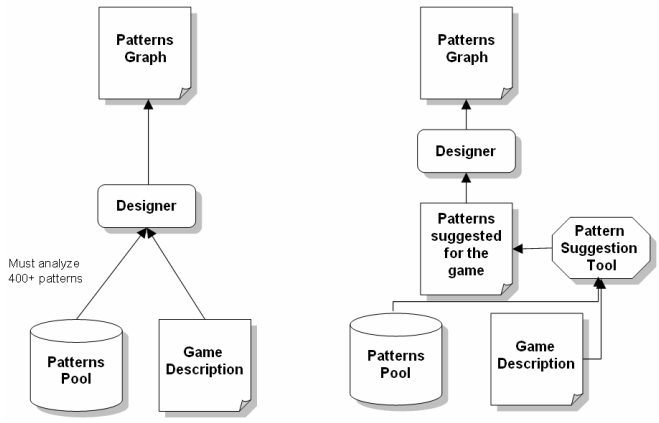

Fig. 1. Pure manual identification of patterns (on the left) and computer aided suggestion (on the right).

B. *Similarity Analysis and Content Suggestion*

Computer science's data mining is the area concerned with information processing and retrieval techniques. Formally known as "knowledge discovering", it comprises various research fields, including similarity analysis and recommendation systems [10]. Those two are specifically important to this paper experiment: the similarity analysis can show how close two documents are and a recommendation system may use it to suggest contents to a user.

Virtually all of today major web sites that rent, sell or advertise any kind of products and services to end users employ some content suggestion, which is done by recommendation systems. According to [10], the main applications of recommendation systems are product suggestions at on-line retailers, movie recommendations and news articles. The key idea behind the technique is to discover the user's preferences and, based on this, which content recommend to him. Blogs and other regular content providers, like YouTube, employ recommendation systems. Netflix has been using computer aided suggestion for a long time. It suggests movies and TV shows to the user based on the type of content he usually watches. Amazon does the same when recommending products to costumers.

The techniques that a recommendation system employs may vary according to the type of data to be analyzed. Rajaraman, Leskovec and Ullman [10] classify two types of recommendation systems: content-based systems and collaborative filtering. The first type examines document contents in order to discover which of them seems to be more related. For instance, a book store system may analyze the summary of a reader's books and, using content similarity analysis, match it with thousands of other book's summaries in order to suggest the most similar. On the other hand, collaborative filtering is based on the user opinion: a movie retail web system may recommend movies to a costumer by matching his ratings of the movies he already watched with the ratings of others users in order to find similar costumers to him, and then, which movies they have already watched that he didn't.

As for the present work, the content-based recommendation systems, which employ similarity analysis techniques, are very usefully to the suggestion of the most related GDP of a game based on the analysis of their content. The patterns wiki contains the full descriptions of the patterns and we can obtain the game descriptions from many sources from the web. The explanation of the techniques and the process employed are described in the following section.

III. THE EXPERIMENT

The experiment consisted of selecting a game, obtaining its description and matching it with all the GDP descriptions in order to find the patterns which have the most similar content, thus suggesting them to the user. To achieve this, a software application was built in order to automate the entire process.

A. Methods and Tools

A software application (Fig. 2) to execute the experiment was built using the Java platform and Apache Lucene [11], a library for text processing. The software was a key element in the experiment once it provides functionalities that allowed us to agilely extract and check the necessary data. By using the app, we were able to get the patterns suggestion for the game by inserting the game's name in the search field. As can be seen in Fig. 2, we could also configure the parameters of the similarity analysis process. Those parameters are: the number of terms of each document that will be used in the similarity matrix, the number of reviews for the selected game that will be processed during analysis and the size of the patterns list to be shown in the result box.

Beyond helping us to collect the necessary data and executing the similarity algorithms, the tool allowed us to reduce the load of checking the correspondence between each suggested pattern and the game by providing the pattern description (this process is described in a later section). Another feature implemented in the software is the ability to plot a relationship map of a game design pattern for quickly visualization (Fig. 3). By looking into the map we can check the suggested patterns and the patterns to which these relate. While not specifically valuable to the experiment of this paper, it will serve in future works.

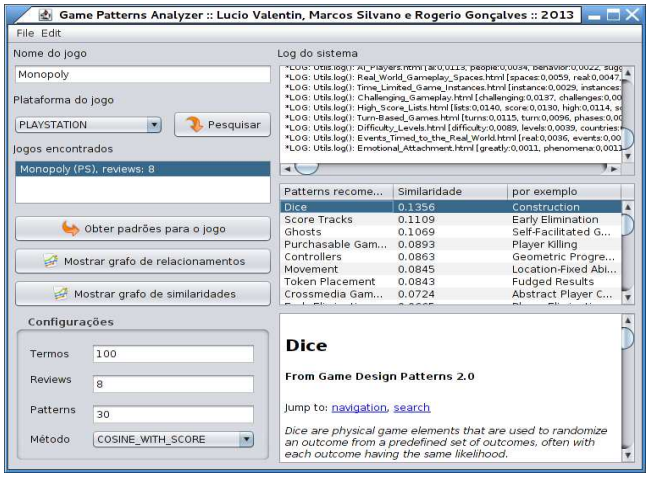

Fig. 2. Screenshot of the suggestion tool built with Java.

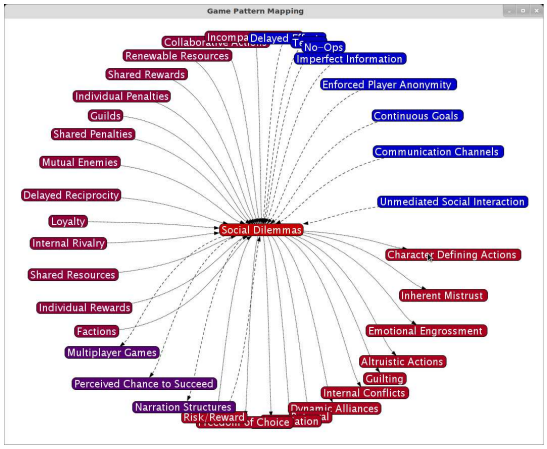

Fig. 3. Relationship map of a game design pattern. The related patterns are positioned around the selected pattern, which is at the center.

The software application downloads all the necessary data for the experiment from specific websites. The patterns descriptions are obtained from the GDP project wiki. The textual data is then cleaned (removes formatting) in order to extract only the desired information to the experiment, which are the patterns description. For the game description, there were two options as source of data: game manuals or game reviews. Game reviews were considered more appropriated to this experiment as they contain game design analysis, whereas game manuals are more concerned in teaching the user on how to play the game. Thus, the website GameFaqs[2] was used as the source of game reviews as it has many reviews for each game.

The entire process performed by the software application to recommend patterns can be summarized in the following steps (see Fig. 4):

1. When first used, the application will download, extract and clean the descriptions of all the patterns from the GDP project wiki. (The information regarding the patterns are indexed with Lucene and stored locally for better performance).
2. The user enters a game name;
3. The software obtains and cleans the game reviews from GameFaqs website;
4. The data is indexed with the Lucene library and stored in a proper data structure.
5. The game description data is compared with the description data of each pattern using a similarity algorithm, which indicates how close the two textual contents are.
6. At the end, the suggestions of GDPs for the game are listed in a descending order of similarity.

The data structure, the indexing process as well as the methods for similarity analysis are described in the next subsections.

[2] http://www.gamefaqs.com/ (visited on 2013/06/24)

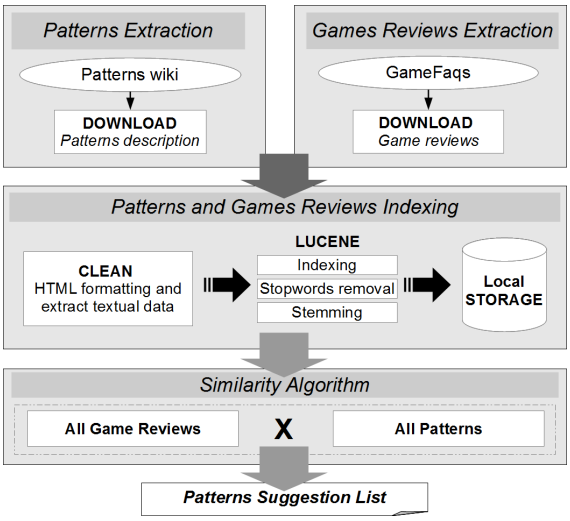

Fig. 4. Operation method of the suggestion tool.

B. *Similarity analysis method employed*

There are some techniques that can be used to denote how similar two documents are. In our case, we want the algorithm to tell us how similar a game description and a pattern description are. After discovering the similarity between each pattern and the game description, we can sort the resulting list in a descending order of similarity: from the most to the least similar.

The first step of the similarity algorithm between two documents employed consists of extracting all the *terms* (words or other strings of characters other than white space [10]) from the document content. This is done with the Apache Lucene library, which removes the *stop words* and *stems* the resulting terms.

According to [10], stop words are the least important terms in the processing of natural language, once they help to build ideas but do not carry any significance by themselves. Examples of stop words are articles and pronouns. In the experiment, we used the same stop words dictionary from MySQL [12], Onix [13] and Ranks [14] systems.

After removing stop words from the text, the Lucene stems the remaining terms. The stemming is the process of reducing inflected words to their root form [10]. As an example, the words "stemmer", "stemming" and "stemmed" are reduced to the root form "stem".

After the processing is completed by Lucene, the (root) terms of a document are stored into an associated one dimensional matrix, which is used to store the terms frequency within the document: the number of times the term appears in the document. Each document has its own frequency matrix. The matrix length is equal to the total number of terms of all documents. Each position in the matrix corresponds to a term frequency inside the document (replaced later in this experiment by the TF.IDF score [10] for better results, as discussed bellow).

When the frequency matrices of the two documents – game review and pattern description – are done, the cosine similarity [10] between them can be calculated. Singhal [15] states that the cosine similarity technique gives a useful measure of how similar two documents are likely to be in terms of their subject matter. This is especially useful for the experiment described in this paper. We are interested in finding documents with similar meaning and not character-level similarity, which wouldn't say much about how close are a pattern and a game. With this objective in mind, we later combined another technique suggested by [10] that is taken as useful for measuring the similarity of meaning between documents: the TF.IDF score.

After the first run of experiments, we soon noticed that the pattern suggestions tool was overvaluing common terms like "game" or "player" in the frequency matrix, as they really were the most common terms in the analyzed documents. However, the importance of these terms was very low to our analysis. Thus, we had to consider the importance of each word within a document to discover its most meaningful terms. For this, we replaced the frequency matrix by a matrix of TF.IDF scores of the terms. The TF.IDF (*Term Frequency times Inverse Document Frequency*) gives a score based on the importance of a word in a document [10], which gave us better results in the experiment (as demonstrated in the next section). The TF.IDF obtainment process is explained hereafter.

C. *The TF.IDF Score*

According to [10], the TF.IDF score can be used to find the terms that best describe the document they belong to. To obtain the score of the importance of a term *i* in a document *j*, we use the equation (1).

$$score(i, j) = TF_{ij} \times IDF_i \quad (1)$$

The TF_{ij} in (1) represents the *Term Frequency* of the term *i* in the document *j*. The IDF_i in (1) represents the *Inverse Document Frequency* of the term *i* over the entire collection of documents under analysis. They are calculated by the equations (2) and (3), respectively.

$$TF_{ij} = f_{ij} / Nterms_j \quad (2)$$

The equation (2) shows us how the term frequency is calculated. In (2), the frequency of the term *i* in the document *j* (f_{ij}), which is the number of times the term occurs in the document, is normalized by the total number of terms of the document ($Nterms_j$). If a term count is equal to the total number of terms in the document, then the TF for this term will be 1. Otherwise, the TF will have a value between 0 and 1.The more the term appears in a document, the higher will be its TF value.

$$IDF_i = log_2(Ndocs/N_i) \quad (3)$$

In equation (3) we have the obtaining of the inverse document frequency (IDF), which gives us the presence of the term *i* in the entire collection of documents. N_i represents the number of documents in which the term *i* appears and *Ndocs*, the total number of documents in the collection.

D. Running the Experiment

The results of the experiment discussed in this paper represent a work in progress. The software tool built is not complete and mature enough to be used in real world scenarios, but the achievements done so far are very positive and encouraging. We chose to show the gradual results of the experiment instead of only the final (and better) results as a way to report the complete process and justify the decisions made.

The games chosen to be tested were classic games from the third generation of videogame consoles. A total of three games were used to run the experiment: Super Mario Bros. [16], Ninja Gaiden [17] and Tetris [18]. Two of these games were select because they were the best selling games of their platforms [19] [20]: Super Mario Bros. for the NES [21] and Tetris for the Game Boy [22]. The third game, Ninja Gaiden, also for the NES platform, was chosen by a personal preference of the authors.

The experiments were done comprising various trials towards better suggestions of patterns (the results are shown in the next section). By better suggestions we understand having more patterns related to a game in the list showed by the software tool. Instead of using just one game review, we progressively tested the patterns suggestions against all the available reviews for each game. In all the tests, we experimented the two similarity algorithms previously pointed, cosine similarity with frequency matrix and cosine similarity with TF.IDF score matrix. The former was the first implemented in the software and the later, as a measure to improve the suggestion mechanism.

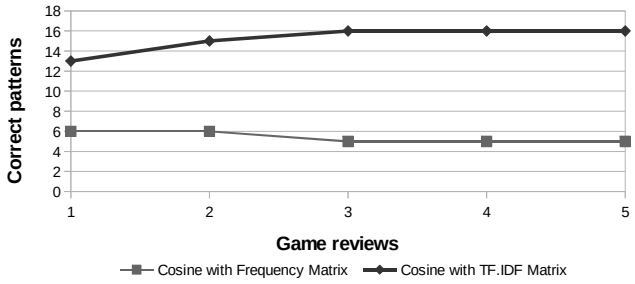

Fig. 5. Progression of the correct suggestions of patterns over the number of reviews for the game Tetris (Game Boy).

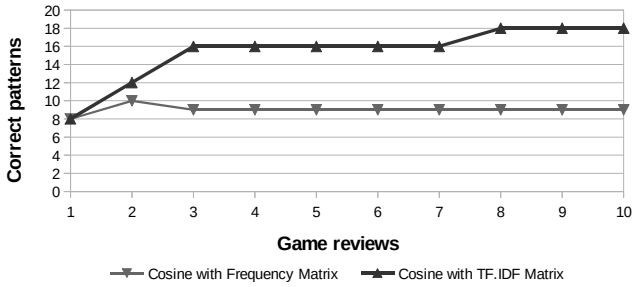

Fig. 6. Progression of the correct suggestions of patterns over the number of reviews for the game Ninja Gaiden (NES).

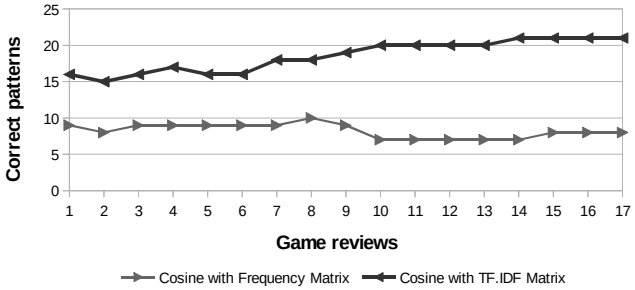

Fig. 7. Progression of the correct suggestions of patterns over the number of reviews for the game Super Mario Bros (NES).

E. Results and discussions

The results for the experiments are synthesized in the graphs shown in the Fig. 5, 6 and 7. We ran the software tool for each one of the chosen games and collected the list of patterns suggested. We then checked each of the patterns to see which were accepted for the games, which means that, we manually verified if the suggested patterns were really contained in the games. This was comprised of a manual task of reading each one of the patterns description and verifying its acceptance to the game's concepts. Then, we counted how many accepted patterns the tool suggested and used this as a measure of evaluation for the experiment. It is important note that, as a recommendation system, the tool doesn't "know" when a pattern is accepted or not, it simply recommend it in order of similarity to the game description, in terms of their subject matter.

Fig. 5, 6 and 7 show the graphs that contain the tests results for the three games selected for the experiment. As can be observed in the graphs, using the cosine similarity mixed with the TF.IDF score matrix gave notably better results. These results help to corroborate with the suspicion that, when combined, the cosine similarity and the TF.IDF score form a useful tool to measure the similarity between two documents in terms of their subject matter.

The analysis of different games with an unequal number of reviews helped us to note another fact: as can be seen in the three presented graphs, the number of reviews has strongly influenced the results. More than that, it was directly related to the number of accepted suggestions of patterns. In most cases, the more game reviews we used in the analysis process, the more patterns were appropriately suggested by the tool.

The suggestion lists obtained for the three games are shown in the Tables I, II and III. The accepted patterns in each table are presented in bold. In these tables, we limited the resulting list of suggested patterns to the first thirty positions (from a total of four hundred). Above this, we couldn't find any accepted patterns in the list, although we don't believe that "thirty" is a rule for all games, which will be subject of study on future experiments. While the perfect result would be having all the correct patterns grouped in the beginning of the list, and not scattered in the first thirty positions, the current result is very positive, if we consider that the suggested

patterns were all positioned between the first thirty positions of a four hundred list. This suggests that, although not perfect, the employed technique is going towards the desired results.

TABLE I. LIST OF SUGGESTED PATTERNS FOR TETRIS

Nº.	Pattern	Nº.	Pattern
1	Line of Sight	16	Crossmedia Gameplay
2	**High Score Lists**	17	Meta Games
3	**Puzzle Solving**	18	**Difficulty Levels**
4	**Challenging Gameplay**	19	**Levels**
5	**Time Limits**	20	Zero-Player Games
6	**Casual Gameplay**	21	Time Limited Game Instances
7	**Geometric Progression**	22	**AI Players**
8	Development Time	23	**Obstacles**
9	Free Game Element Manipulation	24	Speed Runs
10	Game Worlds	25	Score Tracks
11	Non-Player Help	26	**Back-to-Back Game Sessions**
12	**Single-Player Games**	27	**Gameplay Statistics**
13	**Split-Screen Views**	28	Ghosts
14	Extra-Game Consequences	29	Dice
15	**Tiles**	30	**Diegetically Tangible Game Items**

TABLE II. LIST OF SUGGESTED PATTERNS FOR NINJA GAIDEN

Nº.	Pattern	Nº.	Pattern
1	Back-to-Back Game Sessions	16	**Cutscenes**
2	**Levels**	17	Feigned Die Rolls
3	Backtracking Levels	18	Drop-In Drop-Out
4	**Challenging Gameplay**	19	**Avatars**
5	Crossmedia Gameplay	20	**Loot**
6	Difficulty Levels	21	Trick Taking
7	Quick Returns	22	God Fingers
8	**Death Consequences**	23	**Combat**
9	**Lives**	24	**Traverse**
10	**Enemies**	25	**Exaggerated Perception of Influence**
11	**Invisible Walls**	26	**Game Worlds**
12	**Weapons**	27	**Freedom of Choice**
13	**Boss Monsters**	28	Non-Player Help
14	**Split-Screen Views**	29	**Damage**
15	Dynamic Difficulty Adjustment	30	Illusion of Open Space

TABLE III. LIST OF SUGGESTED PATTERNS FOR SUPER MARIO BROS.

Nº.	Pattern	Nº.	Pattern
1	**Levels**	16	Parallel Lives
2	**Warp Zones**	17	**Enemies**
3	**Power-Ups**	18	**Thematic Consistency**
4	Crossmedia Gameplay	19	**Lives**
5	**Boss Monsters**	20	Difficulty Levels
6	Friendly Fire	21	**Diegetic Consistency**
7	Back-to-Back Game Sessions	22	**Avatars**
8	Dice	23	**High Score Lists**
9	**Speed Runs**	24	**Multiplayer Games**
10	**Challenging Gameplay**	25	**Rescue**
11	**Temporary Abilities**	26	Big Dumb Objects
12	**Quick Travel**	27	**Movement**
13	**Privileged Movement**	28	**Diegetically Tangible Game Items**
14	Ghosts	29	Backtracking Levels
15	**Traverse**	30	**Obstacles**

IV. FINAL THOUGHTS AND FUTURE WORK

Recommendation systems have been used successfully in on-line services for consumer's content suggestion, like retailers, movie services and news articles. Even outside this scope, these systems can be applied to measure the similarity between groups of textual contents, thus recommending the contents with higher similarity. This is particularly valuable to the question that driven this work: would be feasible to use these techniques to help designers to identify the patterns used in a game? The results of the experiment described in this paper shows that it is possible, but improvements in the algorithm used may lead to a better outcome.

The similarity method employed in the software tool specifically built for the experiment presented throughout this paper, has showed encouraging results. It helped us to confirm that both the cosine similarity and the TF.IDF score are really useful methods to analyze the similarity between two documents in terms of their subject matter. The cosine similarity method alone was able to achieve some results, but the mixing with the TF.IDF score matrix gave a notably better outcome. While it didn't have achieved the most reliable results, all the appropriately suggested patterns were grouped, with some scattering, at the first positions of the list, which brings a promising initial result.

Another important fact that deserves to be noted is that the reviews used in the similarity analysis during the experiment were mostly written by end users. Although some of them cover important aspects of the game being reviewed, they may lack some more technical language, like the one usually employed by professional game designers or even the specialized media writers. Clearly, the problem is where to find a good source of game reviews with technical quality.

During the evaluation phase of the experiment, when we had to confront each of the suggested patterns with the game by reading the patterns definition, we noticed that some patterns that should have been suggested for the game had relations with the suggested ones. We observed this fact when visualizing the patterns relations through the relationship maps generated by the software tool. In this sense, it's possible that patterns that have some relations may translate them into the games that use these patterns, but this was clearly a suspicion and not an observed fact. However, this would be a probable subject of study and experimentation in the next step of the present work.

As future work, we intent to implement other similarity algorithms in order to compare the results with the ones presented in this paper. Rajaraman, Leskovec and Ullman [10] as well as Singhal [15] present other techniques that may be triable.

Another possibility of a further study is the employment of collaborative filtering techniques in an attempt to enhance the suggestion algorithm. We can create a collaborative tool where designers and users may choose the most probably correct patterns for a specific set of games and, based on the similarity of these games with the ones under analysis, we may suggest the patterns. An experiment of this kind may allow us to combine collaborative filtering with automatic content analysis in order to get more adequate suggestions of game design patterns.

With more solid results, we may have the tool available to public usage and experimentation, allowing a broader collecting and processing of usage data, thus improving the recommendation process via the combined techniques of content similarity and collaborative filtering. When we reach this mark, we will have a fully functional tool that may provide an easy way for big studio designers, independent developers and game specialists to analyze and create games using the game design patterns framework.

REFERENCES

[1] Neil, K. Game Design Tools: time to evaluate. Proceedings of DiGRA Nordic 2012.

[2] S. Björk, S. Lundgren, and J. Holopainen. Game Design Patterns. In DIGRA Conf., 2003.

[3] Kreimeier, B. Game Design Methods: a 2003 survey. Gamasutra, March 2003. http://www.gamasutra.com/view/feature/2892/game_design_methods_a_2003_survey.php.

[4] Costikyan, G. I have no words & I must design. Interactive Fantasy, Eng., n2, 1994.

[5] S. Björk and J. Holopainen. Patterns in Game Design (*Game Development Series*). Charles River Media, December 2004.

[6] Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. Design Patterns: elements of reusable Object-Oriented Software. Addison-Wesley, (1994).

[7] Loh, S. and Soon, S. H. Comparing computer and traditional games using game design patterns. In Proceedings of the 2006 international conference on Game research and development, CyberGames '06, pages 237{241, Murdoch University, Australia, Australia, 2006. Murdoch University.

[8] Tolmie, J.; DiPaola, S.; Charles, A. Towards an interactive visualization of game design patterns. In DIGRA Conf., 2005.

[9] Samarnggoon, K. An analysis of game design using graph-based substructure mining technique. In Computer Research and Development (ICCRD), 2011 3rd International Conference on, volume 1, pages 368 {372, march 2011.

[10] A. Rajaraman, J. Leskovec and J. D. Ullman. Mining of Massive Datasets. 2013, http://infolab.stanford.edu/~ullman/mmds.html.

[11] Lucene. The lucene search engine. http://lucene.apache.org/, 2012.

[12] MySQL: Open Source Database Management System. http://www.mysql.com/.

[13] Onix, Onix Text Retrieval Toolkit, http://www.lextek.com/onix/.

[14] Ranks, Ranks Web Page Analyzer Tool, http://www.ranks.nl.

[15] A. Singhal, Modern Information Retrieval: A Brief Overview. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering 24 (4): 35–43. 2001.

[16] Nitendo, Super Mario Bros. Nintendo Enterteinment System: NES [Cartridge], Japan: Nintendo Co. Ltd, 1985.

[17] Tecmo, Ninja Gaiden. Nintendo Enterteinment System: NES [Cartridge], Japan: Tecmo Co. Ltd, 1988.

[18] Bullet Proof, Tetris. Nintendo Enterteinment System: NES [Cartridge], Japan: Bullet Proof Software, 1989.

[19] List of best selling videogames, Wikipedia, http://en.wikipedia.org/wiki/List_of_best-selling_video_games

[20] Global sales per game. Videogame Chartz, http://www.vgchartz.com/gamedb/.

[21] Nintendo, NES: Nintendo Entertainment System (videogame console). Japan: Nintendo Co. Ltd., 1983.

[22] Nintendo, Game Boy (videogame console). Japan: Nintendo Co. Ltd., 1989.