Improvement of a Web Browser Game Through the Knowledge Extracted from Player Behavior

João Alves, José Neves, Sascha Lange and Martin Riedmiller

Abstract In this article, we describe how a web browser game can be improved through the knowledge obtained from the analysis of the behavior of its players. The analysis of player behavior is a technique that has been used with success in traditional computer games. These kind of analyses have been proven to help the developers in creating games which provide a more engaging and enjoyable experience. Currently, there is an interest in trying to replicate this success in a less-conventional genre of games, normally called browser games. The defining characteristic of browser games is the fact that they are computer games which are played directly on the web browser. Due to the increased ease of Internet access and the growth of the smartphone market, this game genre has a promising future. Browser games regarding the area of game mining have an advantage over traditional computer games in that their characteristics player behavior is relatively easy to collect.

Keywords Knowledge extraction · Data mining · Machine learning · Browser games · MMOG

J. Alves (✉) · J. Neves
Department of Informatics, University of Minho, Braga, Portugal
e-mail: joao.mpereira.alves@gmail.com

J. Neves
e-mail: jneves@di.uminho.pt

S. Lange
5DLab, Freiburg, Germany
e-mail: sascha@5dlab.com

M. Riedmiller
Department of Computer Science, University of Freiburg, Freiburg, Germany
e-mail: riedmiller@informatik.uni-freiburg.de

A.M.J. Skulimowski and J. Kacprzyk (eds.), *Knowledge, Information and Creativity Support Systems: Recent Trends, Advances and Solutions*, Advances in Intelligent Systems and Computing 364, DOI 10.1007/978-3-319-19090-7_5

1 Introduction

Browser games are computer games that have the unique characteristic of being played directly on the web browser without the need of additional software fittings. The web browser is then responsible to provide the player interface and the access to the game world.

Typically, games in this genre offer a multiplayer environment where users can interact with each other. Attending to the way this genre works, the current game status of the players needs to be stored by the game provider. This fact makes it easy to collect information about the players, since companies already need to store it in order for their games to work.

There is an increasing awareness in the game developers community about browser games with the rise of smartphones with web access [3] and the growing popularity of social platforms [9].

Traditionally, the business model for computer games involves selling a game package to the customer that enables him to enjoy the game. However, by definition, in browser games there is no package to be sold. To be able to gain popularity and to attract new players, due to fierce competition, these games are available free of charge. Because of this constraint, developers have to resort to other sources of revenue. Commonly these sources are advertising, which may be displayed easily within the player interface, or selling in-game features, which typically enhance the player's game experience [16].

In this paper, we will focus on the behaviorial analysis of the players registered in a browser game developed by 5DLab[1] called Wack-a-Doo[2] and the improvements made in result of this analysis. Wack-a-Doo is a strategy browser game which requires players to develop economical and research structures so that they can maintain an army able to resolve military conflicts with other players and to secure in-game points of strategic interest. Regarding this game, some work has already been published; however, the focus was more on enumerating important behaviorial characteristics and less on improvements that can be created taking into account the information obtained from those characteristics [1].

In browser games, the playing mechanics involve the player making a limited set of actions, for example moving armies or founding cities that have an associated time cost for completion. As a result, the player can leave the game and return only when the time associated to these actions is over or he can perform more in-game actions. Because of these playing mechanics, the typical user performs short playing sessions various times a day [10]. The mechanics of Wack-a-Doo, while having their own original flavor, follow also this generic formula.

[1] http://www.5dlab.com.

[2] http://www.wack-a-doo.com.

2 Related Works

The growth in popularity of free-to-play games like Wack-a-Doo has created a new research perspective on behavior analysis. Because the major revenue stream for the developers of these kind of games is selling in-game content through microtransactions, the analysis of the players preferences and their development is truly crucial for the game's financial success [16]. The games developed under this game model require a constant analysis of the player behavior in order to maintain their financial sustainability [12].

The dependency that games have on obtaining knowledge from the player behavior lead to the creation of many fairly recent companies that provide solutions which enable the game developers to perform all kinds of in-depth analysis of the data collected by the game. Some companies present these characteristics namely Honeytracks[3] or GameAnalytics.[4]

Unfortunately, the amount of studies publicly known about knowledge extraction in MMOGs is limited, mainly due to confidentiality issues. The available knowledge for free-to-play games is more comprehensive; however, it generally comes from articles or blog posts [6].

One of the techniques used for analyzing player data is calculating metrics that serve as key performance indicators (KPIs); examples of KPIs in MMOGs could be the session times, the churn rate or if applicable even tutorial completion information [7]. Because of the more mature state of the field of web analytics, there are some techniques that were adapted from this field and used in the context of MMOGs. Examples of this adaptation could be conversion rates analysis, user acquisition cost analysis, or cohort analysis [8].

There are also data mining techniques used on more traditional computer games that have objectives such as behavior prediction [14], classification of user behavior [2, 5], and retention modeling [8] that are interesting and can potentially provide very useful information to the MMOGs developers [6].

3 Motivation

The sources of revenue in Wack-a-Doo are the selling of extra features to the players and the selling of bonus that give a temporary game advantage to the players who buy them. There is then, of course, a huge incentive for the developers to create a game that incentives the consumption of those products. To improve their game experience it was decided to analyze the player's behavior.

The company behind Wack-a-Doo divides the players in categories according to a hierarchy of states. This classification is made in such a way that when a player advances to the next conversion state, he can never posteriorly be classified at a

[3] www.honeytracks.com.

[4] http://www.gameanalytics.com.

lower position in this hierarchy. From the lowest to the highest hierarchic level, the conversion states found in this study are *registered*, *logged in once*, *played 10 min*, *logged in 2 days*, *active player*, and *paying player*.

Aside from the conversion state, there are other indicators that the Wack-a-Doo developers suggest as important, such as the time played or of course the very common churn ratio. Our objective in this study is to analyze the player behavior and extract knowledge that can help developers improve or understand these indicators. The indicators used in this study will be primarily the time played and the conversion state.

There could be multiple ways to make the analysis of the player behavior, the approach chosen for this study was to extract as much knowledge using as source only the first playing days of each player. As such we decided to use the information from the first 3 days of each player and make a short-term prediction. The day used for the prediction value was the 10th day.

Given the long-term characteristic of the games in this genre [16], the study focus on only some days can be confusing. The fact is that the analysis is not only on some days of the game history; what happens is that we used not all the days of the game history but only the first 3 days of the player history. This short period is relevant due to three factors. The former relies in the fact that it is in the first days of playing that we can find the information about all the users experimenting the game for the first time. The second point is that the first days of each player are very important in defining their behavior for the rest of the game; for example, in Wack-a-Doo after the 10th day and not counting inactive users 81 % of the players maintain their conversion status. The latest is a consequence of the high short-term churn ratio that is verified on Wack-a-Doo.

This short-term analysis can also be very useful to help the companies quickly decide on the effectiveness of a marketing campaign. For example, they can study the players that registered from a certain marketing campaign in their first few days, and predict whether or not there is a good percentage of them that are going to bring any added value to the game.

4 Data Collection and Preprocessing

For this study to be relevant, the data needs be extracted automatically from the deployed Wack-a-Doo database. As such, the first task was to analyze the database and decide which of the stored data could be used to describe the player's behavior. Fortunately, the developers had a very-well-designed database and stored various player attributes useful for this study.

The approach taken while selecting the attributes to collect from the database was a comprehensive one. The final list contained 33 features for each player that varied from the gameplay related such as the number of armies or the number of settlements founded, to the more social oriented such as number of messages received/sent

or number of likes received/sent, to the obligatory more financial oriented such as amount of money spent.

One of the characteristics of the Wack-a-Doo database is that it does not have a time-related attribute associated with its data. In practise, only the current state of the player is registered and there is no information about when certain changes in the state of the player occurred. Because of this it would not be possible to study the evolution of the player using Wack-a-Doo's database alone.

To surpass this shortcoming we developed a system for collection and preprocessing of player status that we call SysCOPPE. This system is capable of giving us the necessary knowledge about player's evolution. With SysCOPPE we are easily able to collect all of the selected attributes from the Wack-a-Doo database at regular intervals. For the reasons of this study, we decided to opt to use an interval of 24 h between each data collection (Fig. 1).

SysCOPPE runs every day and stores its daily data collection in a text comma-separated values (csv) file. After that, the data in the csv file is preprocessed and then inserted in the SysCOPPE database. The data used for this study was collected during a period of 108 days with the first collection data being 23 of March the first data collection date.

During the preprocessing of the csv's SysCOPPE performs data filtering and cleansing tasks. For example, sometimes players use stolen credit cards to make purchases and when that happens their registered money gross will revert back but the conversion state will not. In this case SysCOPPE will update the conversion state to the value it was before. Another example more related to the developer Wack-a-Doo implementation could be the negative values for time played when the player has registered but never logged in, when this happens SysCOOPE changes the value to zero.

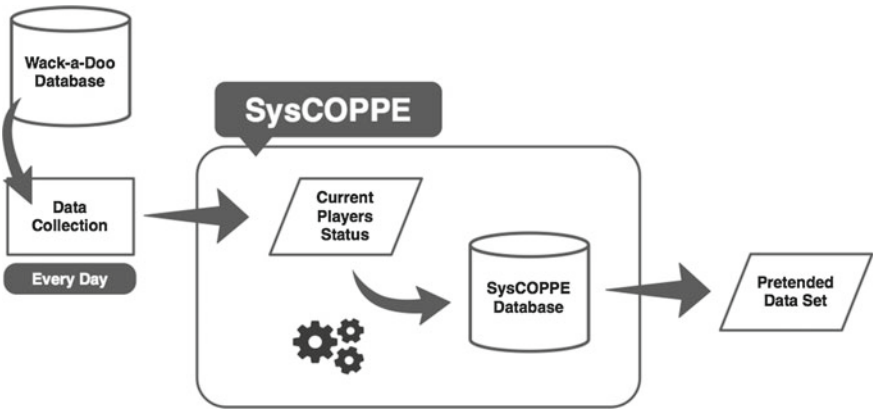

Fig. 1 SysCOPPE

Another of the features of SysCOPPE is the ability to generate datasets according to user specified requirements. The user can specify e.g., the time interval, the filtering inactive users or of course the target feature of his pretended dataset.

5 Methodology

According to the definition, a time series is a series of timely sequential observations $X_i(t)$; $[i = 2, \ldots, N; t = 1, \ldots, T]$, where T refers to each time in point and i refers to the number of variables in the time series. In our case, the number of observation by each point in time is equal to the number of players and the variables registered are the features that we record for each player. Because our number of observations per point in time is bigger than one, we can say that the data registered on the SysCOPPE database allows the construction of a multivariate time series (MTS).

However, when creating this time series we need to be careful when choosing what values do we want to use for our time axis. We could make a time series using the collection data as the time value, but from an analysis point of view this would be highly debatable. What makes sense is to position the time axis in relation to the days since the player registered in the game; with this when we select the time position number 1, we will only see players in identical game development stages. With the necessity cut our time series to only three points in time, the traditional MTS mining techniques are inadequate [15]. Because of this constraint imposed by our objectives and to avoid confusions, we will stop treating our data as a time series.

The first step to prepare for our experiments was to create the two datasets with the 33 collected features during the first 3 played days with one having as target the conversion state and the other the time played. Because the datasets comprise of data collected during three days plus the prediction target value on the tenth, the number of features for each recorded player was not 33 plus the prediction target but 33 for each day (e.g., for each player we have recorded the number of logins on day 1, on day 2, and on day 3. The number of players used for this study were 4212.

The number of attributes collected are already considerable; however, we were not totally satisfied because our datasets consisted only of absolute values that did not revel directly the player evolution. To overcome this problem, we decided to do some data transformation adding new attributes.

One of the measured features where we made some data transformation was the number of armies (where we calculated the difference of the number of armies between each of the 3 days). This gives a more direct measure of how fast the player is creating armies. Another example could be the information that we store each day if the player has joined or not an alliance under the form of a Boolean and we added a new attribute saying specifically in which day he joined an alliance. We also added some attributes that are not game related but could prove to be relevant such as the day of the week when he registered. With this new approach we added 59 new attributes, having this new dataset version a total of 158 attributes. This addition should make the task of the prediction algorithms easier.

During our experiments we used the Repeated Incremental Pruning to Produce Error Reduction algorithm [4] for rule learning and the C4.5 algorithm [13] for decision tree generation. This particular decision to use these kinds of algorithms is justified by the fact that we do not want to create the best prediction model possible but to extract knowledge from the data. Decision tree and rules are one of the most direct ways to accomplish this objective [11].

6 Experiments and Results

6.1 Conversion State

For the analysis focused on the conversion state, we used the RIPPER and the C4.5 algorithms. For this classification problem, the decision tree model had a general accuracy of 90.2 %, really close to the 91.0 % general accuracy obtained with the model created by RIPPER (Table 1).

The confusion matrix associated to the model created by the RIPPER algorithm shows that there is some difficulty distinguishing between the *active player* and the *logged 2 days* conversion states. Actually this observation also holds when analyzing the model resulting from the C4.5. According to the RIPPER model, the precision percentages of the *active player* conversion state and the *logged 2 days* states are, respectively, 77.9 and 89.25 %, being the former the worst class precision of both models. The recall for these two states is, in the same order, 82.25 and 55.77 %.

These results show that there is a very big difficulty predicting the *logged 2 days* state, and the class most affected by this is the *active player* state. In fact this is one of the reasons why we can say that the investment in adding new attributes to the dataset paid of. With these new attributes, the confusion between those two classes decreased around 20 % in both of the algorithms used. It was interesting to verify also that the attributes that made most difference in decreasing this confusion were the ones directly measuring player evolution, e.g., the difference between the time played in one day and the time played in the next one.

Table 1 Conversion state classification RIPPER confusion matrix

Pred. →	Ten minutes	Logged once	Active	Logged 2 days	Registered	Paying
Ten minutes	1317	0	4	0	0	0
Logged once	0	911	0	0	0	0
Active	14	1	292	46	0	2
Logged 2 days	174	26	72	382	31	0
Registered	1	0	0	0	896	0
Paying	0	0	7	0	0	36

This difficulty to distinguish between active players and players who logged in twice in a row can probably be explained by the nature of our analysis. When using only the information about the first 3 days the difference of a player who has logged in twice in a row and an active player should not be very noticeable. However the active players are crucial to the prosperity of Wack-a-Doo, this is why we dedicated some time to the analysis of how the models behaved in distinguishing these two classes.

From the learned rules, the one that is most helpful in distinguishing between the *active player* and the *logged in twice* states is the one that dictates that on day 3 the player played the game less then 24.3 min. This value is now useful for the developers as a goal to reach for each player and a performance indicator.

Another useful finding that helps distinguish between players from these two states is that if by day 2 the player has not raised an army, then he will never be an active player. What was concluded from this finding was that the military aspect of the game was an engaging factor for the players and as such it should be advertised. As a result from this finding, a slight change in the tutorial tasks of the game was introduced. In this new version of tutorial, the military mechanics are explained not only earlier but also more extensively.

It is also curious to observe again that the victory ratio of a player is positively related to the time he will be playing in the future. This means that the players who have a bigger win/loss ratio will end up playing the game more. This is a somewhat expected because it is only natural that a player finds a game more enjoyable when he is winning. However, boosting the victory ratio of the players is tricky because if one player wins another has to loose. The solution that was created for this ratio problem was the introduction of computer-controlled armies in the game. This new kind of armies not only helps boosting the players victory ratio but also, especially in the early game, provides more action for the players.

In the RIPPER model, there was only one rule that was used to classify the players belonging to the paying state. This rule can be interpreted as stating that a customer who was a paying customer in the first 3 days is classified as a paying customer in the 10th day. This is a trivial rule; however, what surprises is that it is able to classify almost 84 % of all the paying customers. This high percentage probably means that the game developers are doing a good job on encouraging the new players to spend money in their very first days.

From the models created it was evident that a higher number of logins per day indicates a likely higher retention of the player. It was that if a player only logged in once in the second day, then by the 10th day he would not login any more. To try to lure the players into the game and motivate them to come back, a temporary bonus feature was added. This feature consists of small “bubbles” of resources that appear around the player’s villages and when popped give the resources to the player. These bubbles are temporary and need to be popped within a certain time.

6.2 Time Played

Every time a player is playing the game, Wack-a-Doo keeps track of the time they spend and when their session is over it updates the amount of registered playtime. We can say that Wack-a-Doo keeps a cumulative counter of each players playtime. The metric unit used to register the time played is seconds. Because this data is stored in a continuous scale we had the need to discretize it.

When we tried to create bins of the same range size, we faced a problem because the number of users in each of the bins is very unbalanced. The most common bin has the range $(-\infty, 63385]$ and contains 98.5 % of players. This happens for two reasons the first is that it is expected for the cluster with low values to contain more players because of the users who register and never play or just try the game for one unique short session. The second is that there are some few players who spend a big amount of time playing, for example there are 18 players that are distributed between 126769 and 316923 s of playtime. And because of this distribution of players the range size for the bins needs to be big in order to each bin to have at least one player and to accommodate all players. Because of all this, the upper boundary of the first bin is really high which again contributes to unbalance of the player distribution across the bins.

With this last failure to apply an automatic discretization algorithm, we learned that it would make more sense for us to create the bins according to our analysis of the Wack-a-Doo data. After our analysis, we propose five profiles of players according to time playing patterns found in the Wack-a-Doo data. We should note that while the specific times used in the ranges of each profile are very specific to the Wack-a-Doo game, the theoretical notion behind each of these profiles is probably useful in every game of this genre (Tables 2 and 3).

The frequency of each of the classes still varies greatly from a minimum of 68 players to a maximum of 1773; however, the class imbalance is not as huge as the one observed in the previous discretization attempt. But more important than that, the clusters created make sense from a game point of view and should allow us to extract much more relevant knowledge from the Wack-a-Doo data.

After this preprocessing of the dataset, we used the rule learning algorithm RIPPER and the C4.5 decision tree generator algorithm. Both of these algorithms

Table 2 Profiles for players according to their time played

Name	Time range	Frequency
Not playing	[0, 1]	552
Experimenting players	(1, 100]	330
Casual players	(100, 10000]	1489
Interested players	(10000, 60000]	1773
Hardcore players	(60000, $+\infty$)	68

Table 3 Time classification C4.5 confusion matrix

Pred. →	Not playing	Experimenting	Casual	Interested	Hardcore
Not playing	552	0	0	0	0
Experimenting	8	319	2	1	0
Casual	3	3	1482	1	0
Interested	5	2	59	1687	20
Hardcore	0	0	0	29	39

performed very well with the decision tree having a general accuracy with 96.84 % and the rules a general accuracy of 95.75 %.

Because of the slight imbalance of our classification, general accuracy can be a very misleading measure; however, after studying both of the confusion matrices, we can conclude that the models created are relevant. In both of the created models, the worst precision results are obtained for the casual and hardcore players classes. In the particular case of the decision tree, the class precision of casual players is 96.05 % and of hardcore players 66.10 % with a respective recall of 99.53 % and 57.35 %. With this analysis, we can also conclude that this model does not perform well when differentiating between hardcore players and interested players when they are hardcore players.

The *completed tutorial* attribute in the decision tree model is one of the major factors distinguishing between the interested and hardcore players. This means that a more appealing game tutorial is very likely to bring more players to classes related to higher playtime. This led to a bigger focus on the development of a better tutorial and resulted in a rework of the tutorial design and functioning. Among the new tutorial features are an automatic pop-up system that details every assignment with pictures and themed graphics or an helping arrow that guides the players throughout the assignments. This helping arrow at first points to every location where the player needs to click but as the tutorial is getting to an end it starts to get less common and only indicating how to access recently introduced game features. Another change was the inclusion of relatively advanced game features in the tutorial that were previously left out.

Other important factor distinguishing between the interested and the hardcore players is the already mentioned success in battle. For the players who played more then 355 min, the ones who had a number of defeats bigger than zero in the first day are likely to be interested players and the ones who had more victories than defeats are likely to be classified as hardcore players. The solution created for this was already mentioned in the previous section where an artificial victory ratio boosting mechanism is discussed.

Example of the positive effect of the social features

```
(diff_time_d3d1 >= 10302) and (time_played_day1 >= 6680)
and (overlal_score_day3 >= 315) and (diff_ovrscore_d2d1 >= 75)
and (day_join_alliance <= 1) => result=hardcore players
```

(Example rule created with the RIPPER algorithm)

In the above example, *diff_time_d3d1* and *diff_ovrscore_d2d1* represent respectively the difference of the cumulative values of time played and overall score between days 3 and 2 and between days 2 and 1. If the player has joined an alliance the *day_join_alliance* attribute has a numerical value indicating in which of the 3 days that happened. This particular rule shows how the fact that a player has joined an alliance on day 1 positively influences his time spent on the game.

The creation of these two models led to the discovery that the social aspect of the game has a positive effect on the time played. The players who have sent messages or likes to other players or joined an alliance are always classified as one of the classes who spend more time playing. With this knowledge some effort was made in order to not only make the social part of the game more appealing but also to make the game itself more social. Some of the features that resulted from this effort are the addition of more channels for the in-game chat system and the introduction of better bonus for the players who brought their friends to the game.

It was added to Wack-a-Doo an interesting characteristic that makes it a more social game. What was changed was that the game when deciding the positioning of the player in the Wack-a-Doo map will take into account the real geographic location of the players. In practice, this means that players from the neighboring cities will be located in close proximity to each other. This change increases the probability that the player's motivation to play will come from real-life social factors. An example of such factors could be a rivalry between two cities that motivates the development of a war inside the game.

7 Conclusions

The future of browser games is predicted to be very promising with the rise of smartphones with web access and the growing popularity of social platforms. This creates an increased awareness in the game developers community, which in turns raises the competition in the browser games market. The usage of data mining techniques to extract knowledge about their game can be the deciding factor to achieve financial success.

In this paper it was described SysCOPPE, a system that was created to collect and preprocess data from a browser game. This type of solution is very versatile and can be easily adapted to a browser game created by a different developer. Future work on SysCOPPE should pass by adding the capability to make an automatic analysis of predefined features every time the SysCOPPE database is updated.

It was presented a case study using the data from a browser game currently in the market. In the experiments section it was suggested five profiles according to playtime patterns that were identified in the Wack-a-Doo data. These profiles are a contribution that may help classify players from other games of this genre.

This article focused on making very short-term predictions of features that the developers considered relevant. It is expected that this article will make a contribution

to comprehend which characteristics of the player's behavior inside the game are important to be taken in consideration, and which parts of the game should be object of a more careful elaboration.

An example of an important player's characteristic with impact on the players behavior is the *victory ratio*. Another example could be the discovery that social-related game features, such as alliances or a like system, have also a positive effect inside the game.

There was also an indication of some changes or features in Wack-a-Doo that focused each of these important player characteristics. These improvements to the game can also be reproduced or at least used as guidelines by other companies that work on the web browser game market.

5DLab has recently released a mobile application of Wack-a-Doo for smartphones. This version of the game is not focused on this study. However, a study of the differences between the mobile players and the web browser players could yield interesting results.

Although it has been concluded that there was useful knowledge extracted using the current approach based only on the player behavior in their first days, we should note that to identify complex behavior patterns it is necessary to use longer time spans.

References

1. Alves, J., Neves, J., Lange, S., Riedmiller, M.: Knowledge extraction from the behaviour of players in a web browser game. In: Proceedings of the International Conference on Knowledge, Information and Creativity Support Systems, pp. 519–530. Progress & Business Publishers, Krakow, Poland (2013)
2. Anagnostou, K., Maragoudakis, M.: Data mining for player modeling in videogames. In: 2009 13th Panhellenic Conference on Informatics, pp. 30–34 (2009)
3. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 20102015 (1 February 2011) by Cisco Systems Inc, http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html, last access September 2014
4. Cohen, W.: Fast effective rule induction. In: Proceedings of the Twelfth International Conference on Machine Learning, pp. 115–123 (1995)
5. Drachen, A., Sifa, R., Bauckhage, C., Thurau, C.: Guns, swords and data: clustering of player behavior in computer games in the wild. In: Computational Intelligence and Games, 2012 IEEE Conference on Computational Intelligence and Games, pp. 163–170 (2012)
6. Drachen, A., Thurau, C., Togelius, J., Yannakakis, N., Bauckhage, C.: Game Analytics, Game Data Mining, pp. 205–253. Springer, London (2013)
7. Fields, T.V.: Game Analytics, Game Industry Metrics Terminology and Analytics Case Study, pp. 53–71. Springer, London (2013)
8. Fields, T., Cotton, B.: Social Game Design: Monetization Methods and Mechanics. Morgan Kauffman Publishers, Waltham (2012)
9. Kirman, B., Lawson, S., Linehan, C.: Gaming on and off the social graph: the social structure of facebook games. In: 2009 International Conference on Computational Science and Engineering, pp. 627–632 (2009)

10. Klimmt, C., Schmid, H., Orthmann, J.: Exploring the enjoyment of playing browser games. Cyberpsychology Behav. **12**, 231–234 (2009)
11. Langley, P., Simon, H.A.: Applications of machine learning and rule induction. Commun. ACM **38**, 54–64 (1995)
12. Mellon, L.: Applying Metrics Driven Development to MMO Costs and Risks. Versant Corporation (2009)
13. Ross, Q.: C4.5: Programs for Machine Learning, vol. 1. Morgan Kaufmann Publishers, California (1993)
14. Shim, K., Srivastava, J.: Sequence alignment based analysis of player behavior in massively multiplayer online role-playing games (MMORPGs). In: Data Mining Workshops ICDMW 2010 IEEE International Conference, pp. 997–1004 (2010)
15. Swift, S., Kok, J., Liu, X.: Learning short multivariate time series models through evolutionary and sparse matrix computation. Nat. Comput. **5**, 387–426 (2009)
16. Vanhatupa, J.: Business model of long-term browser-based games—Income without game packages. In: 2011 7th International Conference on Next Generation Web Services Practices (NWeSP), pp. 369–372 (2011)