Adaptation to TV Delays Based on the User Behaviour towards a Cheating-Free Second Screen Entertainment

Rui Madeira, Pedro Centieiro, Nuno Correia

Adaptation to TV Delays based on the User Behaviour towards a Cheating-free Second Screen Entertainment

Rui Neves Madeira[1,2], Pedro Centieiro[2] and Nuno Correia[2]

[1] Escola Superior de Tecnologia de Setúbal, IPS, Setúbal, Portugal
rui.madeira@estsetubal.ips.pt
[2] NOVA-LINCS, DI, FCT, Universidade Nova de Lisboa, Monte da Caparica, Portugal
pcentieiro@gmail.com, nmc@fct.unl.pt

Abstract. Recent advances in technology created new opportunities to enhance TV personalization, providing viewers with individualized ways to watch TV and to interact with its content. Second screen applications are promising vehicles to enhance the viewers' experiences, but researchers need to take into account the effect that the TV delay has on viewers, in particular when watching broadcasted live events. In this paper, we propose a software-based solution to deal with TV delays. It is mainly directed for a gaming context in which the user has the means to control the synchronisation between the second screen application and the TV content. Taking this scenario into account, we implemented a cheating-detection mechanism to cope with the potential exploitation of the system by its users.

Keywords. Second screen, TV delays, broadcast live events, adaptation, ubiquitous personalization, user profile, game cheating.

1 Introduction

The quest for a personalized TV follows a path started with multichannel TV and has led us to Personal Video Recorder, Video-On-Demand and multiscreen TV [12]. Moreover, the proliferation of mobile devices, such as smartphones and tablets, allows TV viewers to become accustomed to "doing their own thing" [12]. So, if users have "individualized" ways for watching TV and for interacting with TV shows then this can be an opportunity to apply personalization to TV. Second screen applications (apps) are promising vehicles to accomplish personalization since they can provide show-related information and support "social viewing" through dedicated social networks, and even interactive experiences, through polls and quizzes, synchronised with the show content.

However, it is common for some viewers to get events on second screen apps that are not synchronised with the corresponding key moments in the TV broadcasts. The users' engagement can be ruined when a situation like this one occurs, since the app content may be presented too early (spoilers), or too late (obsolete content). Thus, second screen synchronisation is an important step towards TV personalization [12].

In order to address this issue, we use the Synchronisation Mechanism through User Feedback (SMUF), which relies on the feedback given by the users on how apps' key events are synchronised with the corresponding TV broadcasts' key moments [3]. SMUF provides a universal and simple interaction control to the users who become able to adjust their second screen experience according to how they are perceiving the

TV delay. However, this mechanism presents a negative counterpart within gaming scenarios since users can easily find out how to exploit it. If users set up a delay higher than they actually have they will end up receiving the content on the second screen app after watching the corresponding content on TV. This can also happen in situations in which the synchronisation is made by automatic content recognition (ACR) methods, such as audio fingerprinting, where users can synchronise the app with the TV box at a past moment. On both cases, users will be able to win more points since they can, for instance, answer more questions correctly and within shorter time periods. In our context, these users are cheaters since they exploit the system to get an unfair advantage or achieve a target that they are not supposed to [13].

This paper presents a software-based solution that takes into account the users' interaction data, towards a cheating-free gaming experience while using SMUF. In order to achieve this goal, we need to counterbalance SMUF through the implementation of a Cheater-Detection Mechanism (CDM). We followed the generalized model for personalization provided by P^2MUCA [9] in order to build profiles for each user according to her/his gaming and syncing behaviours based on interactions data. If the user is detected with an "abnormal pattern" (e.g., several correct answers in a row in a very short time period) then s/he might be marked as cheater and the app delay can be automatically readjusted according to the app's game mechanics.

In order to understand how users interact with SMUF to find out the appropriate patterns that can tell us who might be a cheater, we collected users' interactions data while they were testing SMUF in WeSync, a mobile app that prompts users to interact on key events related to football TV broadcasts. Results from this study allowed us to gather important insights, which are very promising for future research on this area.

2 Background Research

Synchronisation can affect the user experience of the viewers, particularly when receiving key events on an app before watching them on the TV broadcast. A study by Centieiro et al. [2], in which users during a football match TV broadcast were prompted to bet if a goal would happen in a few seconds, showed that some users were frustrated or stressed for not being able to perform the action. This happened since the match on TV was delayed relatively to the real match and consequently to the match key events on the second screen app. It is in-line with the study of Kooij et al. [7], which states that a variation of the playout delay can go up to 6 seconds in TV broadcasts in the same country, and more than one minute in some web based TV broadcasts. Therefore, there are different solutions to synchronise a second screen app with TV content [3]. However, regardless of which type of synchronisation mechanism developers implement, whether based on ACR or SMUF, users can overcome and exploit it by setting up a false delay. The issue here is that users can manage to set a delay higher than they actually have in order to receive the content on the second screen app after watching the associated content on TV.

In our context, users that try to exploit the system are considered cheaters. Sometimes it is almost impossible to distinguish smart play, e.g., good use of tactics, from cheating, which is defined as "Any behaviour that a player may use to get an

unfair advantage, or achieve a target that he is not supposed to be" [13]. Moreover, evolutionary psychology (EP) observes human nature as the result of a universal set of evolved psychological adaptations to recurring problems in the ancestral environment. EP considers the hypothesis that people have a cheater-detection module [4]. Cheating is a violation of a particular kind of conditional rule that goes along with a social contract. Social exchange is a collaboration system for mutual benefit and cheaters violate the social contract that governs social exchange [5]. The selection pressure for a dedicated cheat detection module is the presence of cheaters in the social world. Therefore, a CDM is needed as an adaptation complement arising in response to cheaters, ensuring the correct implementation of social exchange, which is essential for a game based social experience such as the one we are addressing here.

There are several approaches for cheating detection in games, but they are not directed to second screen gaming. Usually, those solutions are focused on online multiplayer environments, such as the one proposed by Laurens et al. [8], which monitors the player behaviour for indications of cheating play, and the one presented by Yeung et al. [14], which proposes a scalable method to detect whether a player is cheating, or not, based on dynamic Bayesian networks. Alayed et al. present another behavioural-based cheating detection in online first person shooters games using machine learning techniques [1], whilst Ferretti and Roccetti proposed an algorithm based on the monitoring of network latencies [6].

3 SMUF and the WeSync App

WeSync is a mobile app that prompts users to guess the outcome of corner kicks (Figure 1a), penalty kicks and free kicks during a football match. Users can also check their predictions' outcomes, as well as their friends' scores. Furthermore, WeSync also notifies users when a goal is scored (allowing them to quickly share their thoughts on social networks), or when a half starts or ends. However, when these events are not synchronised with the TV broadcast, users need to synchronise them by using SMUF, which can be done by adjusting a slider in a screen that appears right after each key event occurrence (Figures 1b-1c). Users rate their experience through SMUF, providing feedback on how the app is presenting the events. Each subsequent event is presented taking into account the previously provided feedback, in order to achieve the synchronisation between the app and the TV broadcast. Once users state that the app is synchronised, the screen with the slider stops from appearing right after each key event. A popup is shown explaining that from now on they still can adjust their experience whenever they wish by clicking on the top-right button that just appeared (Figure 1d).

We introduced four interaction cues to facilitate the user interaction: the temporal indication (e.g. "app is 3 seconds ahead"); the illustrative animation above the slider which behaves accordingly to the delay set, the overall colour of the slider and the animation, and the experience rating (e.g. "Great!", "Good!", "Fair.", and "Poor…"). The colour intervals and the experience rating values were based on the work done by Mekuria et al. [11]. Finally, we did not include any information regarding the live sports event, such as the match time - which could be used to compare the TV broadcast match time with the app match time to synchronise both feeds - since we wanted to have a

universal synchronisation mechanism that could be deployed on any broadcast, regardless of the information presented on the TV screen.

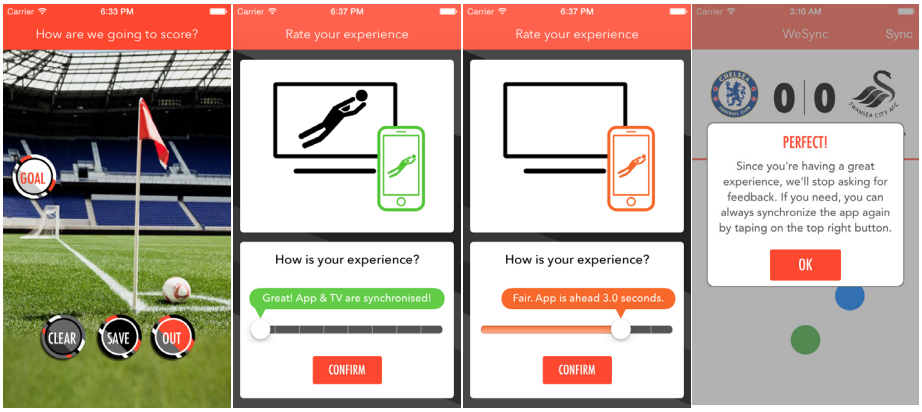

Fig. 1. (a) A key event that prompts users to predict a corner kick; (b) Initial screen of SMUF; (c) User setting a 3s delay; (d) Notification when the application is synchronised.

4 A Cheater-Detection Mechanism

We needed to implement a Cheater-Detection Mechanism (CDM) in order to cope with potential dishonest users (cheaters) as they can freely control the app delay with SMUF. Thus, app delays placed by a user should be personalized (automatically adapted) to benefit the users experience and its social exchange as a whole, persuading and preventing users from cheating. The implementation of CDM follows the generalized model for personalization provided by P^{2}MUCA and named X-Users [10], which can help us expressing what can be a cheater behaviour according to the users' gaming and syncing profiles. P^{2}MUCA is a personalization platform for multimodal ubiquitous computing applications that provides tools and services to help developers in the implementation of personalization solutions. The core of this platform is the personalization model that can be applied to different applications from different domains. The personalization model allows that user interactions with an application A may be used to personalize the experience of that same user when s/he starts interacting with a new application B. For instance, if a new app similar to WeSync is implemented and registered on P^{2}MUCA then a user with a cheater profile in WeSync can be already known by the new app, which allows from the beginning a fast adaptation to the delay set by the user, not requiring initial user interactions for the effect.

In order to use P^{2}MUCA and apply X-Users, we should decide a priori, at design time, what to personalize in WeSync. It is important that we know the app's potential end-users, mainly the specific ones that drive the personalization requirements. According to [10], the careful specification of personas can help to determine what to personalize (instances of personalization, i.e., the app delays in the case of WeSync) and the different options for each personalization. So, in X-Users, each personalization instance (Personalization) can have N different Personalization Options according to the number of identified Personas (Figure 2, left side). A Persona can drive N

Personalization Options since an application can implement different Personalizations. The same Persona can even be the driver for Personalizations of different applications.

Fig. 2. The Personalization core entities of P²MUCA's X-Users model (from [6]).

We want to automatically adapt the app delay set by users (TV viewers) according to their behaviours in the second screen gaming experience. This is a personalization that can be used by any app of this kind that is registered on P²MUCA. So, we had to select a set of resources (interaction data variables) in the context of WeSync (Table 1), combining them to obtain two parameters, which combined correspond to the specified personas (Figure 2, right side).

Table 1. WeSync's Resources (interactions data used for personalization).

R	Resource	Description
1	**keyEvents**	(integer) Total of key events (questions) answered.
2	**totalResponseTime**	(double) Sum of the response time used for all answers.
3	**totalCorrectResults**	(integer) Total of correct answers.
4	**currentUserDelay**	(double) Current value of the app delay after last key event, placed by user.
5	**totalDeltaUserDelay**	(double) Sum differences between successive app delays at each key event.
6	**totalUserDelay**	(double) Sum of the app delays at each key event.

Each parameter is given by a mathematical expression that results in a numeric value representing a specific user behaviour in terms of the resources included. At each key event, the user will therefore have two values: the GamerProfile represented by equation (1), and the SyncerProfile represented by equation (2).

$$0.3 / (\text{totalResponseTime}/\text{keyEvents}) + 0.7 * (\text{totalCorrectResults}/\text{keyEvents}) \quad (1)$$

$$0.5*\text{currentUserDelay} + 0.25*(\text{totalDeltaUserDelay} /\text{keyEvents}) + 0.25 * (\text{totalUserDelay}/\text{keyEvents}) \quad (2)$$

These equations were determined after several iterations of simulations based on empirical data and domain knowledge. Furthermore, the weights were slightly refined with the user experiments study and the results confirmed the equations are appropriate for our goal (Table 4). We were mainly interested in discovering if at each key event (moment) a user should be marked as a cheater, or not. So, initially we decided to represent only two personas (cheater and non-cheater) and, for that, it was only needed a parameter (called TVViewerProfile). However, we found out that two parameters would make much more sense with the separation of resources into Gamer and Syncer profiles, giving us a greater granularity to work on. This way, we defined two options for each parameter (profile): High and Low (they revealed themselves as being appropriate as demonstrated by the results in Table 4). This way, P²MUCA's clustering algorithm should divide users into two clusters for each parameter, receiving as input a parameter's vector composed of the parameter's values representing each user [10]. The permutations of these two options (clusters) of each parameter correspond to four personalization options (Table 2), the considered personas in our scenario: 1) Non-Cheater Tier-1; 2) Non-Cheater Tier-2; 3) Non-Cheater Tier-3; and 4) Cheater.

Table 2. Combinations of the two parameters options result in four personalization options.

GamerProfile / SyncerProfile	High Gamer Value	Low Gamer Value
High Syncer Value	Cheater	Non-Cheater Tier-3
Low Syncer Value	Non-Cheater Tier-2	Non-Cheater Tier-1

In our case study, we consider that a user should only be marked as a potential cheater (final profile) when both parameters (sub-profiles) result in high values. The three tiers of non-cheater should be treated equally by the app if it is not necessary to distinguish between the different sub-profiles of non-cheater users.

5 User Experiments and Data Collection

We carried out user experiments with the WeSync app in order to evaluate the SMUF's usefulness and viability in a gaming scenario and, more important, to collect important interactions data on gaming and syncing aspects. The user tests were conducted with 15 voluntary participants (11 male and 4 female) aged 23 to 45 ($\bar{x} = 31.5$; $\sigma = 7.4$). The tests took place in a lab at our University campus and included two test sessions with participants being briefed before each one. In each session, participants watched an 8-minute highlight video from a football match, containing six key events to bet what the outcome would be. Since several TV providers have different delay values, it was set a random TV delay (values in range 0-6s) for each participant and video. Each participant had two initial moments to become familiar with SMUF, before the key events start appearing, allowing her/him to better understand how to set a delay.

After the first session, we asked participants to rate (with a 5-point Likert-type scale) six statements (Table 3) in a questionnaire regarding their SMUF experience and their cheating perception. Users were also free to write down any further comments and, at the end, we conducted a brief interview. Additionally, during each session, we also registered all the users' interactions data for the resources described in Table 1.

Table 3. Summary of the questionnaire results regarding: the SMUF experience (first 3 rows) and the cheating perception (last 3 rows).

Statements	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I had a good synchronisation experience.	6,67%	0,00%	13,33%	33,33%	46,67%
It is easy to use the synchronisation mechanism.	0,00%	13,33%	13,33%	33,33%	40,00%
This mechanism is useful to synchronise the application with the TV.	0,00%	6,67%	0,00%	20,00%	73,33%
This mechanism allows you to have an unfair competition.	6,67%	6,67%	33,33%	6,67%	46,67%
It is easy to cheat using the synchronisation mechanism.	0,00%	0,00%	20,00%	20,00%	60,00%
The mechanism motivates me to be honest when synchronising.	6,67%	6,67%	33,33%	26,67%	26,67%

The questionnaire's results were very positive regarding the users experience with SMUF. Table 3 shows that, in general, participants had a good synchronisation experience ($\bar{x}$ = 4.1; σ = 1.13) and found it easy to use ($\bar{x}$ = 4.0; σ = 1.07) and useful to synchronise the app with the TV ($\bar{x}$ = 4.6; σ = 0.83). The remainder of the questionnaire also showed very interesting results. Although the majority of participants have agreed (6.67% agreed and 46.67% strongly agreed) that SMUF allows us to have an unfair competition ($\bar{x}$ = 3.8; σ = 1.32), there are a considerable number of neutral opinions and two of them even stated that "the competition would not be unfair since everyone has the same conditions to cheat". Participants really found it easy to cheat using SMUF ($\bar{x}$ = 4.4; σ = 0.83), generally after a few key events. Nonetheless, results showed participants felt motivated by SMUF to be honest ($\bar{x}$ = 3.6; σ = 1.18). We believe this happens because of social and peer-pressure. It is interesting to refer that one participant told us s/he would be dishonest if s/he would perceive that others were cheating.

Furthermore, after the first session, we asked participants to indicate (using a 5-point Likert-type scale) how they behaved in terms of honesty when setting up the app delay for each key event. This was important to link gathered interactions data to users' perception on cheating. For example, no participants stated that they behaved like cheaters in this session and our results actually show no cheaters. Moreover, the results show that participants 1, 2 and 15 had a behaviour progression almost into cheater profile during session 1 and they confirmed that to us. Table 4 shows data of only six participants since these present the most interesting values to understand global results. Data are relative to the moment right after the conclusion of all 6 key events.

Table 4. Data results of 6 key-participants: final profiles, parameters and resources values.

P#-S#	TV Delay	Final Profile	GamerP. (Eq. 1)	SyncerP. (Eq. 2)	R2/ R1	R3/ R1	R5/ R1	R6/ R1	R4
1-S1	2,4	non-cheater tier-2	0,59	4,31	4,56	0,67	0,75	2,00	4,5
1-S2	0,2	non-cheater tier-1	0,21	3,17	3,84	0,17	0,17	1,75	2,0
2-S1	1,6	non-cheater tier-3	0,34	18,17	4,24	0,33	4,00	7,33	24,0
2-S2	4,5	non-cheater tier-1	0,52	5,92	5,95	0,33	0,67	2,67	6,0
11-S1	5,9	non-cheater tier-1	0,30	7,69	2,22	0,33	0,08	4,25	4,5
11-S2	3,3	**cheater**	0,53	16,88	2,54	0,67	1,67	8,92	12,0
12-S1	4,0	non-cheater tier-1	0,19	4,10	2,75	0,17	0,08	2,25	2,5
12-S2	4,4	**cheater**	0,67	14,50	3,42	0,83	2,00	7,33	12,0
14-S1	3,9	non-cheater tier-1	0,22	7,69	4,07	0,17	0,08	4,25	4,5
14-S2	5,7	**cheater**	0,71	26,13	4,60	0,83	1,58	12,83	21,5
15-S1	5,1	non-cheater tier-3	0,24	12,56	4,50	0,17	0,92	6,50	9,5
15-S2	0,8	**cheater**	0,56	12,73	3,79	0,67	0,92	6,83	8,5

In the second session, we asked each participant to set up an app delay for each key event according to a "behaviour guide" with indications (using a 5-point Likert-type scale: 1-"100% dishonest" and 5-"100% honest") on how we would like them to behave. This way, we would gather interactions data that would correspond to different profiles, having participants behaving according to our four options. As we have indicated, only four of them behaved like clear cheaters. It was easy for participant 15

as s/he almost did it in session 1. With the two sessions we got data corresponding to 30 users instead of only 15, which was better for testing purposes. The final profiles confirm the parameters equations previously presented. The final equations were obtained after several simulations with the collected data in which we were refining them in terms of the weight given to each resource.

6 Conclusions and Future Work

We described a new concept based on personalization to enhance cheating-free second screen experiences. This work provides a technology-independent and persuasive solution for competitive gaming contexts. We used a mobile app prototype called WeSync to test it. User tests were very positive and validated our ideas. The results showed that our approach is able to categorize the users based on profiles built from diverse resources combined to create parameters. We are already planning a thorough user study in a real environment to validate the resources and further refine the parameters. We will include a higher number of participants and mechanisms for cheating penalization and appealing (case of false positive cheaters) will be evaluated.

References

1. Alayed, H., Frangoudes, F., Neuman, C.: Behavioral-based cheating detection in online first person shooters using machine learning techniques. In: CIG'13, pp. 1-8. IEEE Press (2013)
2. Centieiro, P., Romão, T., Dias, E. A.: From the Lab to the World: Studying Real-time Second Screen Interaction with Live Sports. In: ACE'14, article 14. ACM (2014)
3. Centieiro, P., Romão, T., Dias, E. A., Madeira, R.N.: Synchronising Live Second Screen Applications with TV Broadcasts through User Feedback. In: 15th IFIP TC.13 INTERACT'15. Springer (2015)
4. Cosmides, L.: The Logic of Social Exchange: Has natural selection shaped how humans reason? Studies with the Wason Selection Task. In: Cognition, 31, pp. 187–276 (1989)
5. Cosmides, L., Tooby, J.: Neurocognitive Adaptations Designed for Social Exchange. In: The Handbook of Evolutionary Psychology, D. Buss (ed.), pp. 584–627. Wiley (2005)
6. Ferretti, S., Roccetti, M.: AC/DC: an algorithm for cheating detection by cheating. In: NOSSDAV'06. ACM Press (2006)
7. Kooij, W., Stokking, H., Brandenburg, R., Boer, P.: Playout Delay of TV Signals: Measurement System Design, Validation and Results. In: TVX'14, pp. 23-30. ACM (2014)
8. Laurens, P., Paige, R.F., Brooke, P.J., Chivers, H.: A Novel Approach to the Detection of Cheating in Multiplayer Online Games. In: ECCS'07, pp. 97-106. IEEE Press (2007)
9. Madeira, R.N., Santos, P.A., Correia, N.: Building a platform for pervasive personalization in a ubiquitous computing world. In: MOBIQUITOUS'14, pp. 345-346. ICST (2014)
10. Madeira, R.N., Santos, P.A., Vieira, A., Correia, N.: Model-based Solution for Personalization of the User Interaction in Ubiquitous Computing. In: 11th IEEE International Conference on Ubiquitous Intelligence and Computing (UIC'14). IEEE (2014)
11. Mekuria, R., Cesar, P., Bulterman, D.: Digital TV: the effect of delay when watching football. In: EuroITV '12, pp. 71-74. ACM (2012)
12. Videonet: Making TV more personal. Industry Report, http://www.v-net.tv/making-tv-more-personal. Accessed 11th May 2015
13. Yan, J.J., Choi, H.: Security Issues in Online Game. The Electronic Library, vol. 20 (2), pp. 125-133 (2002)
14. Yeung, S.F., Lui, J.C.S., Jiangchuan Liu, Yan, J.: Detecting cheaters for multiplayer games: theory, design and implementation. In CCNC'06, vol. 2, pp. 1178-1182. IEEE Press (2006)