Synchronising Live Second Screen Applications with TV Broadcasts through User Feedback

Pedro Centieiro[1,2], Teresa Romão[1], A. Eduardo Dias[1,2], Rui Neves Madeira[1]

[1] NOVA-LINCS, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Caparica, Portugal
pcentieiro@gmail.com, tir@fct.unl.pt, rmadeira@est.ips.pt
[2] Viva Superstars Digital Media Lda, Madan Parque, Caparica, Portugal
edias@bviva.com

Abstract. A common issue on live sports TV broadcasts happens when a viewer hears a neighbour screaming a goal before watching it on his TV. Similarly, viewers will also have a disruptive experience if a second screen application, designed to present information synchronised with the live TV broadcast, displays information beforehand. This paper presents a simple and universal synchronisation mechanism for second screen mobile applications, which does not rely on automatic content recognition, but rather on user feedback to help users achieve a pleasant and effective second screen experience. Results from user tests helped validate our approach, as users were able to synchronise a mobile application with a simulated live TV broadcast, with different delays.

Keywords. Second screen, delays, live TV broadcasts, user experience, sports.

1 Introduction

In recent years, many media devices have become a regular presence in our living rooms. This space, once used mostly to watch television, has evolved into a shared space where we use different devices like laptops, tablets, smartphones or gaming handhelds. Within this refurnished space, an innovative concept has become quite popular: the possibility to interact with the content watched on television through an additional electronic device. This concept, defined as second screen, provides several functionalities that improve the viewers' experience, usually by providing additional show-related information and access to social networks or interactive experiences, such as polls and quizzes, synchronised with the program content. In a survey conducted by Nielsen on the first quarter of 2013 [7], it was showed that nearly half of smartphone (46%) and tablet owners (43%) used their devices as second screens while watching TV every day, which depicts how this interaction concept is becoming widely adopted by TV viewers. Thus, it is not surprising there are mobile applications solely focused on exploiting this concept. For instance, in the world of sports there are several examples of applications that seek to enhance the viewers' experience during live broadcasted sports, usually by predicting what will happen next [8, 11].

However, since different TV providers have diverse types of connections and hardware, it is common for some viewers to get events on second screen applications that are not synchronised with the TV broadcasts. Although this issue could be solved

using automatic content recognition (ACR), like audio watermarking or audio fingerprinting, and inter-destination media synchronisation (IDMS), both approaches may not work on all scenarios. There is not a universal ACR solution that allows developers to synchronise any given TV show, and TV service providers with a lower broadcast delay are not interested in synchronising with others with a higher delay. These facts motivated us to implement a universal and simple interaction mechanism to allow users to synchronise second screen applications with TV broadcasts.

This interaction mechanism is named SMUF (Synchronisation Mechanism through User Feedback) and is based on the feedback given by users on how the applications' events are synchronised with the corresponding TV broadcasts. We developed a prototype called WeSync to evaluate how users interact with SMUF and to analyse if they are able to synchronise a second screen application with a TV broadcast using SMUF. WeSync is a mobile application that prompts users to interact on key events of a soccer match TV broadcast. When these events are not synchronised with the TV broadcast, users can give feedback on how the application is presenting the events: before or after they appear on TV. Each subsequent event is presented taking into account the feedback previously given, in order to have the application and the TV broadcast synchronised. Results from user tests allowed us to gather important insights, which will contribute for future research on this area.

2 Related Work

In the context of second screen interaction, content enrichment [4] refers to the content manipulation that users can perform on either the TV or the second screen device. Several studies were conducted within this context, with second screen applications tackling different areas such as newscasts [1], TV series [6], and sports [3]. These applications seek to deliver in-sync additional content-related media, such as trivia questions, key information and predictions about future events.

When watching a TV broadcast, out of sync multimedia streams can spoil the viewer's experience. In the context of sports, Mekuria et al. [9] presented empirical evidence that relative delays encountered in digital TV degrade the soccer watching experience, especially, when there are fans close to each other (i.e., neighbours) watching non-synchronised TV feeds. Furthermore, the study conducted by Kooij et al. [5] depicts a variation of the playout delay up to 6 seconds in TV broadcasts, and more than one minute in some web based TV broadcasts. Within the second screen context, this can affect the user experience of viewers, particularly when receiving key events on an application before watching them on the TV broadcast. In fact, a study conducted by Centieiro et al. [3], in which users were prompted to guess if a goal would happen in the next seconds during a soccer match, showed that some users were frustrated or stressed for not being able to perform this action. This happened since the match on TV was delayed relatively to the real match and consequently to the match events on the second screen application.

It is possible to solve this issue in either two ways, without having to add or replace hardware: by using IDMS to synchronise all the viewer's televisions, or by using ACR on the TV broadcast audio to synchronise each viewer's mobile

application. The goal of IDMS is to deliver the same stream for all the individuals of a group at the same time. There have been several approaches to achieve this goal, either based on reporting the media stream arrival time and buffering at the end-points [2], or doing it so at the network itself [10]. However, viewers are dependent on the TV service providers to implement these solutions, and if they do not do it the synchronisation problem will remain. Regarding ACR, there are two main methods to identify a given TV show: audio watermarking and audio fingerprinting. Audio watermarking works by adding a well-defined sound to an audio stream, in order to be detected by an algorithm. Audio fingerprinting (popularly used by Shazam) works by comparing an audio fragment to a database of unique audio fingerprints of millions of audio files. However, when there are not third-party APIs for ACR that can detect a given TV show, viewers will not be able to synchronise their applications with the TV broadcast. Furthermore, the development of an ACR system for a given TV show may result in a complex and lengthy process that developers may not wish to go through.

3 Synchronisation Mechanism through User Feedback (SMUF)

As we mentioned before, it is easy to think of a scenario where solutions like the IDMS algorithms and ACR are not able to guarantee a synchronised experience between the TV and the second screen applications to all users. Since different users might have different TV broadcast and network delays, we argue that the solution needs to go through them. Users should be involved in the solution process, by providing feedback on their experience. Thus, we implemented a simple, universal and low-cost synchronisation mechanism called SMUF that developers can quickly add to any kind of second screen application and by which users can easily synchronize them with the TV. To evaluate how users interact with SMUF and to analyse if they can synchronise an application with a TV broadcast, we developed a prototype called WeSync.

WeSync is a mobile application that prompts users to guess the outcome of corner kicks, penalty kicks and freekicks during a soccer 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 after each event occurrence (Figure 1a). Through SMUF, users can rate their experience, providing feedback on how the application is presenting the events. Each subsequent event is presented taking into account the previously provided feedback, in order to achieve the synchronisation between the application and the TV broadcast.

An interaction walkthrough is presented to show how SMUF works on WeSync:

1. A screen appears asking users to rate their experience after completing a given interaction, such as predicting the outcome of a corner kick.
2. Users may specify that the application is synchronised by simply clicking on the "Confirm" button, or they can adjust the slider in order to select how the application behaved in comparison to the television.

3. If the user starts to adjust the slider, a text appears describing the experience rating currently selected (e.g., “Good! App is 1.5 seconds ahead.”). Moreover, the slider changes its colour (e.g., red indicates a higher delay) and an animation starts on the upper half of the screen indicating how that given delay corresponds to the user experience. Users with a high delay can move the slider all the way to the right (max = 4s) and, once there, if they keep their finger on the slider, the slider scale will increase allowing for a higher maximum value (4s are added to the maximum value every 0.5s a finger is kept on the slider). Users can touch the “Confirm” button once they are satisfied.
4. The next time a key event happens, the application will delay the request by the number of seconds chosen by the user, so it can be closer to the TV broadcast timeline. A screen will appear once again asking for his feedback (Figure 1a) after the user finishes interacting with the key event. At this time, users can slide to the left, decreasing the delay that was set (again, a finger kept on the slider increases the minimum value, which is the delay value previously selected). Users can also slide further into the right (Figure 1b), increasing the delay (up to a maximum of 4s). The process will repeat until the user states that the application is synchronised, or close to it (delay < 1s).
5. Once the users state that the application is synchronised, the application stops asking them about their experience. Then, a popup appears explaining that from now on they can adjust their experience whenever they wish, by clicking on the top-right button that just appeared at that point (Figure 1d).

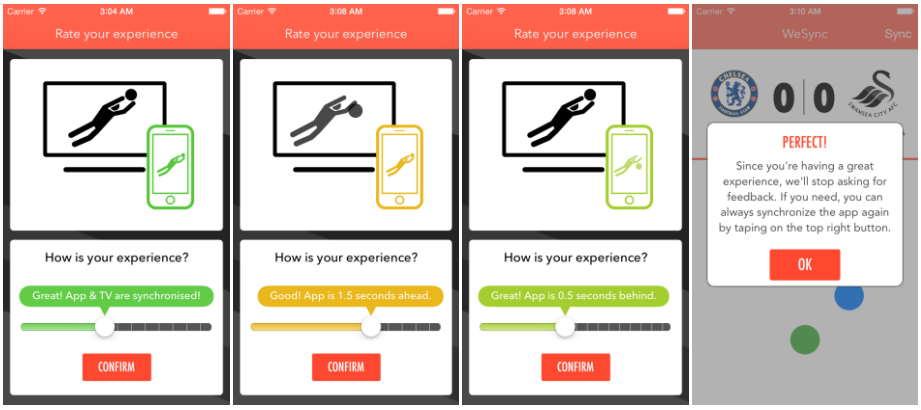

Fig. 1. (a) Iterative process of setting a delay. After setting a 3s delay in a previous interaction. (b) Second try to set the right delay. (c) Third try to set the right delay. (d) Notification after the application is synchronised.

We designed SMUF by taking into account several facts. First, we assume that the events triggered on the second screen application are always synchronised with their real counterparts occurring on the venue where the live event is taking place. This is not just for the sake of simplicity, but it is a guideline for similar works: live second screen interactions that rely on tight synchronisation will not work correctly if they are not synchronised with the real time of the live action, or at least, the fastest TV broadcast available. If this is not the case (e.g. the human operator triggering the events on the application is watching a delayed TV broadcast), users with a TV

broadcast delay lower than the operator's will get the events on the application after they have occurred on the TV broadcast, with no possibility of synchronising them.

Second, we introduced four interaction cues to facilitate the user interaction: the temporal indication, the illustrative animation, the overall colour, and the experience rating. The temporal indication aims at giving precise information on how the TV delay compares with the mobile application events (e.g. "App is 3.0 seconds ahead"). Some users may not be able to initially identify their exact delay, but they will gradually get closer. The remaining cues are defined according to the temporal indication value. The illustrative animation simulates the situation corresponding to the delay specified by the user, aiming at providing the user with a better perception of his choice: the key events on the mobile device are appearing synchronised with the TV, ahead, or behind it (Figures 1a, 1b and 1c). The animation reproduces the exact temporal indication value selected by the user up to a maximum of 4s. The use of colours allows the user to immediately recognize the current situation: green for synchronized, red for not synchronised at all and yellow for in-between situations. In order to complement this cue, we also added a popup text describing the experience rate currently selected by the user ("Great!", "Good!", "Fair.", and "Poor…"). The way that values change on both of these interaction cues were based on the work done by Mekuria et al. [9], who studied how viewers are disturbed by different TV delays.

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 application match time to synchronise both feeds - since we wanted to have a universal synchronisation mechanism that could be deployed on any broadcast. Thus, considering that it is not certain that a well-defined cue is present on the TV broadcast to help users synchronising their applications, we designed SMUF without relying on any extra information besides the users' feedback regarding their experience.

At this stage, WeSync simulates a soccer match broadcast on a TV, by presenting different videos with several highlights. Running on an iPhone, the application presents a screen with information about the match, while waiting for events to occur. Each highlight video is streamed from the iPhone to an AppleTV connected to a TV.

4 Evaluation

We carried out a user study based on the WeSync prototype in order to evaluate the SMUF's usability and usefulness. The user tests were conducted with 30 voluntary participants (28 male and 2 female) aged 20-36 ($\bar{x}$ = 23.3). The tests took place in a room at our University campus. Participants were briefed before each test session.

In each session, participants watched three 7-minute highlight videos from Chelsea 4 - 2 Swansea match that took place on September 13, 2014. None of the participants had seen the match before. We edited each video to contain the same number of events. Since several TV providers have different delay values, it was set a different delay (low, medium and high, as random values in the range of 0-2s, 2-4s and 4-6s, respectively) for each highlight video, based on the work by Kooij et al. [5]. A within-subject experimental design was used to evaluate the three delay scenarios and the sequence of videos and delay scenarios was counterbalanced to minimize learning

effects. After watching each video, we asked participants to rate two statements in a questionnaire regarding their user experience at the start and at the end of the video. Lastly, at the end of each test session, users were asked to fill in the remainder of the questionnaire. This questionnaire was based on the USE questionnaire and was comprised of nine statements regarding general feedback about the activity, and the SMUF's interaction cues. Users rated these statements by using a five-point Likert-type scale, which ranged from strongly disagree (1) to strongly agree (5). Users were also free to write down any further suggestions and comments. Finally, we also registered all the users' interactions during each test, such as number of times SMUF was accessed, response times, delays set by users, among other important data.

Users accessed SMUF 240 times during the whole evaluation process, averaging 2.677 times per user/scenario (σ = 1.43), with 1.566 times during the low delay scenario (σ = 0.78), 2.633 times during the medium delay scenario (σ = 1.16), and 3.833 times during the high delay scenario (σ = 1.31). These data show that when the delay is lower users need to perform fewer interactions with SMUF to get the application synchronised with the TV. Since each scenario had 6 events where users could compare the application to the TV broadcast, these results are very positive. Only in high delay scenarios users needed to set a delay more than 3 times on average.

We also analysed the time duration of each user interaction with SMUF. We were interested in finding out if the users' interactions with SMUF got shorter as the number of interactions performed by them increased and they became more familiar with SMUF. In Figure 2a, we can see that on average, during the first interaction, users took around 7 seconds to set a delay, but it lowered over time reaching 3.5s on the sixth interaction (in this case we could only analysed the medium and high delay scenarios, as users did not accessed SMUF more than 4 times during the low scenario). Only on the third interaction the overall response increased, but we did not find any data that could explain these results in an objective manner. Although, we presume that, at the third interaction, users were wondering why the application was still not synchronised with the broadcast, hence the time they took to set a new delay.

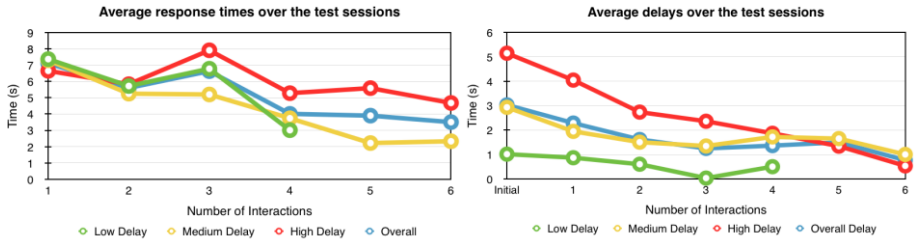

Fig. 2. (a) Average response time for all scenarios. (b) Average delays for all scenarios.

An important topic that we also investigated was the difference between the initial delays of the scenarios and the final delays achieved by the users at the end of the videos. We wanted to analyse which delay values the users were satisfied with, and whether they managed to set the application as synchronised at the end of a video. We verified that on average users ended the low delay scenario, the medium delay scenario, and the high delay scenario with 0.6s (σ = 0.63), 1.38s (σ = 0.70), and 1.61s (σ = 0.98) delays. Overall, the average final delay was 1.20s (σ = 0.89). We presume

that these values could change slightly if the videos were longer – the delay difference was converging to values near or below 1s over time as presented on Figure 2b – as users may still try to do fine adjustments in order to reach a perfect synchronisation (ending up decreasing the average difference). In addition, we found out that on overall, only 23.76% of users changed the delay after setting the application as synchronised, trying to achieve a perfect synchronisation.

Regarding the questionnaire, results were extremely positive when we analysed the reported user experience at the beginning and at the end of each video. As Table 1 shows, users always had a better experience by the end of the video, after using SMUF.

Table 1. Summary of the questionnaire results regarding the user experience. Low delay in green, medium delay in yellow and high delay in red (higher scores are highlighted).

Statements	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
In the beginning, I had a good experience.	0%	3.3%	16.7%	43.3%	36.7%
In the end, I had a good experience.	0%	0%	0%	16.7%	83.3%
In the beginning, I had a good experience.	0%	6.7%	33.3%	40%	20%
In the end, I had a good experience.	0%	3.3%	0%	20%	76.7%
In the beginning, I had a good experience.	0%	36.7%	40%	16.7%	6.7%
In the end, I had a good experience.	0%	0%	6.7%	46.7%	46.7%

The remainder of the questionnaire also showed positive results. In general, participants stated that they liked to use SMUF (56.7% agreed and 36.7% strongly agreed) and found it easy to learn (46.7% agreed and 50% strongly agreed) and to use (33.3% agreed and 56.7% strongly agreed). The majority of the participants disagreed with the statement "I had difficulties to understand how the key events should be synchronised" (40% strongly disagreed and 26.7% disagreed). Finally, almost all participants stated that SMUF was useful to synchronise the application with the TV broadcast (46.7% agreed and 46.7% strongly agreed). These results give us a lot of confidence in applying SMUF to other kinds of broadcasts.

Next, we evaluated the different interaction cues to ascertain whether they helped the users to synchronise the application with the TV broadcast. Table 2 shows that the temporal indication and the overall colour proved to be very helpful for users to interact with SMUF ($\overline{\square} = 4.47$, $\sigma = 0.57$, and $\overline{\square} = 4.4$, $\sigma = 0.81$, respectively). Users also rated the illustrative animation and the experience rating as helpful ($\overline{\square} = 4.0$, $\sigma = 1.08$, and $\overline{\square} = 3.63$, $\sigma = 1.09$), although getting lower scores than the previous ones.

Table 2. Summary of questionnaire results regarding the questions presented at the end of each test session. Higher scores are highlighted.

Statements	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The temporal indication helped to synchronise the application.	0%	0%	3.3%	46.7%	50%
The experience rating helped to synchronise the application.	3.3%	13.3%	23.3%	36.7%	23.3%
The overall colour helped to synchronise the application.	0%	3.3%	10%	30%	56.7%
The illustrative animation helped to synchronise the application.	3.3%	3.3%	26.7%	23.3%	43.3%

Finally, one third of the participants mainly suggested that: the device should vibrate on whistles marking the key events; and it should be possible to access SMUF after making a prediction and while waiting for its outcome, which we agree.

5 Conclusions and Future Work

This paper presents an interaction mechanism called SMUF, which allows developers to quickly add a simple, universal and low-cost synchronisation mechanism to their second screen applications. In order to evaluate SMUF, we implemented a mobile application called WeSync, where users were prompted to synchronise different kinds of events during a soccer broadcast match. The user tests were extremely positive, validating SMUF's design and concept. Overall, participants managed to set a delay near 1s, making WeSync almost perfectly synchronised with the TV broadcast. One of the most important results was that participants stated that they had a better user experience by using SMUF to synchronise WeSync with the videos.

In the future, we aim to solve the issues reported by the users in order to improve the user experience. We will also study how to prevent cheating during a real second screen competition. Currently, users that set up a delay higher than the real one of a TV broadcast get an advantage in predicting key events, since they see what happens on the TV before interacting on the application (happens both with SMUF and ACR). We hope that by analysing usage patterns we can prevent users from exploiting this flaw.

References

1. Blanco, R., Morales G.D.F, Silvestri F.: Towards leveraging closed captions for news retrieval. In Proc. WWW '13 Companion, pp. 135-136 (2013)
2. Boronat, F., Lloret J., García M.: Multimedia group and inter-stream synchronization techniques: A comparative study. Information Systems, 34, 1 (2009)
3. 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)
4. Cesar, P., Bulterman, D., Jansen, J., Geerts, D., Knoche, H., Seager, W.: Fragment, Tag, Enrich, and Send: Enhancing Social Sharing of Video. Journal of ACM Transactions on Multimedia Computing, Communications, and Applications, 5, 3 (2009)
5. 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)
6. Nandakumar, A. and Murray, J.: Companion apps for long arc TV series: supporting new viewers in complex storyworlds with tightly synchronized context-sensitive annotations. In: TVX '14, 3-10. ACM (2014)
7. Nielsen: How Second Screens are Transforming TV Viewing, http://bit.ly/1grK4R3
8. MLB Preplay, http://bit.ly/1BKVyUV
9. Mekuria, R., Cesar, P., Bulterman, D.: Digital TV: the effect of delay when watching football. In: EuroITV '12, pp. 71-74. ACM (2012)
10. Stokking, H.M., van Deventer, M.O., Niamut, O.A., Walraven, F.A., Mekuria, R.N.: IPTV Inter-Destination Synchronization: A Network-Based Approach. In: ICIN'10, pp. 1-6. IEEE (2010)
11. Viva Ronaldo, http://bit.ly/1LTsplD