Wersync: a web platform for synchronized social viewing enabling interaction and collaboration

Fernando Boronat[1] (ORCID: 0000-0001-5525-3441), Mario Montagud[2] (0000-0002-2398-1505), Pau Salvador[3] (0000-0002-2197-4942), and Javier Pastor[4] (0000-0001-6693-8489)

Universitat Politècnica de València – Immersive Interactive Media R&D Group

Calle Paraninfo, 1, 46730, Grao de Gandia, Valencia (SPAIN), +34962849341

{fboronat@dcom[1],pasallla@epsg[3],fjpastor@dib[4]}.upv.es; mamontor@gmail.com[2]

ABSTRACT. Shared media experiences between geographically distributed users are gaining momentum. When remote users are consuming online media content together while interacting, several challenges still need to be addressed to provide an enjoyable experience (e.g. in terms of delays and synchronization). This paper presents Wersync, a web-based platform that enables the creation of different groups of geographically distributed users for concurrently consuming the same media content in a synchronized manner, while socially interacting via text and/or audiovisual chat channels, and even collaborating. Its main features have been derived via a thorough state-of-the-art analysis, results of a previous survey and user-centric activities. Interestingly, Wersync includes adaptive and accurate media synchronization solutions, which performance has been objectively evaluated, obtaining better results than the ones needed for the kind of applications Wersync is intended for, such as Social TV. Additionally, the features of the platform have been subjectively evaluated (N=60 participants) in a social viewing scenario, obtaining satisfactory results, mainly in terms of perception about performance, originality, usability, attractiveness, usefulness, applicability and awakened interest. These results and the development based on standard web components foresee the potential impact and widespread adoption of the platform in the future.

Keywords: Shared media consumption, Social Interaction, Social TV, Synchronization, WebRTC.

1. Introduction

Co-located media consumption has traditionally been a social habit, e.g., several users watching TV in the living room. However, it is not always possible to enjoy shared experiences in the current global society, in which family members and relatives commonly live apart, or in emergency (e.g. pandemic) situations. The emergence of new technologies, together with the advent of Social Media and conferencing services, bring the possibility of recreating these experiences while being geographically distributed. Nowadays, we are witnessing a transition from physical togetherness towards networked togetherness around media content consumption. Novel forms of shared media experiences are gaining momentum [1], allowing distributed users to socially interact within the context of simultaneous content consumption. Examples of relevant use cases are Social TV (a.k.a. social viewing)[1], multi-player games, tele-work and e-learning. The authors' study in [2], involving more than 1000 participants, provides many interesting statistics and insights regarding the relevance of the Social TV use case. That study is a proof of evidence of the high interest Social TV awakens, and it identifies the need for better technological solutions to

---

[1] Social TV or social viewing scenarios allow different groups of viewers, independently of their location and the network (and the device) they are using, to watch a TV program, or online content, while simultaneously interacting and sharing services.

support those scenarios. Around 80% of the participants showed an interest in Social TV scenarios, but only 20% declared having participated in similar experiences before. Regarding the latter, 30% stated that being able to share the video consumption experiences with other remote relatives is mostly the reason why they are or have been willing to watch the contents, and 90% declared having perceived delay differences in these scenarios, considering it as a serious barrier for satisfactorily interacting while watching TV with other remote users. Likewise, the study identifies many benefits of Social TV, such as the feeling of togetherness, socialization and increased engagement. However, several technological challenges still need to be addressed to enable natural and truly interactive remote shared experiences. Some examples are shared experience modeling, universal session handling, synchronization (abbreviated as *sync*, hereafter), Quality of Service (QoS), scalability, presence awareness, design guidelines, privacy concerns, and social networking integration [3]. Although there exist many solutions for conferencing (tele-meetings), collaborative work and e-learning, they are not designed for a shared synchronous consumption of high quality continuous media (such as TV or video), and some challenges are not solved yet (especially regarding sync, as identified in [2]).

In this paper, authors face some of those key challenges. With the insights of a thorough state-of-the-art (SoA) analysis, a set of user-centric activities to determine the necessary requirements for the envisioned scenarios, and the transformation of the derived requirements into technical features, an innovative solution for efficiently providing such synchronized and interactive social viewing scenarios for remote users is presented. It is called Wersync, and is a web-based platform that enables distributed media consumption, social interaction and/or collaboration between different groups of remote users, in a synchronized manner. It provides many novel and enhanced features compared to other existing platforms (summarized and compared in Sections 2 and 3). Although initially designed for entertainment purposes, mainly social viewing, its applicability for other collaborative scenarios, such as e-learning and tele-work, was also identified in the SoA analysis and conducted user-centric activities.

Most of the participants in the aforementioned study [2] stated that the availability of accurate media sync solutions is essential, and they felt that more efficient solutions were needed. Indeed, previous studies have shown that the magnitudes of delay variability between streams and destinations are much larger than tolerable limits to the human perception, as summarized in [4], chapter 1. These situations lead to users' annoyance and dissatisfaction, reflecting the need of media sync. For instance, imagine a scenario in which various friends are watching an online football match, while apart. Herein, being aware of a goal through the cheering of a friend via the chat channel, before the goal is actually displayed on the local screen, will lead to frustration and will prevent users from participating in such social viewing scenarios. To avoid these situations, on the one hand, Wersync provides the smooth sync of media playout for different groups of users (a.k.a. Inter-Destination Media Sync or IDMS [1, 45]), which is a key technological enabler to support coherent shared media experiences. It also enables the sync of the distributed execution of the playout or VCR-like control commands, such as play, pause, skip to another scene, or fast forward. On the other hand, it also includes an inter-stream sync solution between video and chat streams to time-align their presentation, avoiding spoiling situations. The correct performance of both types of sync solutions implemented in Wersync has been objectively evaluated, obtaining better results than the ones needed for the kinds of applications Wersync is intended for, such as Social TV. Additionally, the features of the platform have been subjectively evaluated (N=60 participants) in a social viewing scenario, obtaining satisfactory results, mainly in terms of perception about performance, originality, usability, attractiveness, usefulness, applicability, and awakened interest.

The main contributions of this paper can be summarized as:

- The identification and selection of key aspects and features (requirements) to efficiently enable interactive and collaborative social viewing scenarios. They have been derived by thoroughly reviewing the SoA, and by conducting user-centric activities with experts (Sections 2 and 3).
- Based on the insights from the previous contribution, the design and implementation of a new platform (main contribution), called Wersync (Sections 4 and 5). Even acknowledging the relevance and usefulness of the existing platforms, it was decided that the development of a new one was the appropriate solution to efficiently meet the newly gathered requirements. Multiple arguments contributed to this decision: 1) not all the platforms were still available, and not all of them were open-source; 2) some platforms did not use modern/efficient technological components; 3) the modularity of some platforms did not always allow for the accommodation of extra features and/or improvements; 4) the availability of a new platform enables higher efficiency and accuracy for conducting research-oriented tasks; etc. In conjunction, Wersync provides many innovative and enhanced features compared to SoA platforms, and initial versions of the platform. The following key features can be highlighted: the use of HTTP-based adaptive streaming (MPEG DASH, Dynamic Adaptive Streaming over HTTP); ad-hoc adaptive and accurate Inter-Stream and IDMS solutions (without needing any external clock sync mechanism or technology); and low-latency and synchronized social interaction and/or collaboration tools, through secured communications. Moreover, Wersync includes a responsive and modern Graphical User Interface (GUI), and allows the (optional) recording of important data about the shared media session for further analysis and research.
- Comprehensive objective and subjective evaluations of the platform (Section 6), showing its satisfactory performance and confirming its benefits, and getting valuable insights about the interest and preferred options in shared media consumption scenarios. The obtained results are also discussed.

The rest of the paper is structured as follows. Existing platforms for social viewing are reviewed in Section 2, comparing them with Wersync. Relevant studies related to social viewing / Social TV are also summarized in this section. Section 3 presents the process followed to gather the key requirements for social viewing, which in turn have determined the design choices, and the rationale on using web-based technologies and components for developing Wersync. Then, the main functionalities of Wersync are described in Section 4. Section 5 includes the involved technologies, the architecture and some implementation details. The evaluation scenario, the followed methodology and the obtained results are presented and discussed in Section 6. Finally, the conclusions and some future work ideas are summarized in Section 7. Two additional Appendixes with complementary technical information are also provided.

2. Related work

This section firstly reviews related platforms to enable shared video watching or social viewing. A qualitative comparison between these platforms is also provided, with the goal of comparing their features, but most importantly to compare them with the Wersync ones. Likewise, as Social TV is a key use case of these platforms, relevant studies in that field are additionally reviewed in this section.

2.1 Related Platforms for Shared Video Watching / Social Viewing

In [3], a shared video watching platform, implemented by using native applications, is presented. As interaction features, it integrates both text and voice chat channels, and allows sharing the navigation control commands. An IDMS solution, originally proposed in [6] for gaming, is adopted and adapted to synchronize the playout across the involved devices. It can make use of both M/S (*Master/Slave*[2]) and DCS (*Distributed Control Scheme*) sync control schemes [1] for choosing the sync reference (the receiver with larger delays), and employs NTP (Network Time Protocol, RFC 5905[3]) for time-aligning the clocks of the involved entities. Its performance was assessed when using both sync control schemes, obtaining similar results. The asynchrony levels were kept below 500 ms in a WAN scenario, and below 200 ms in a LAN scenario. The platform was also used in [7] to subjectively assess the tolerable asynchrony levels in shared video watching scenarios. It was concluded that delay differences around 1 s might be noticed (and even of 500 ms in specific situations), but differences over 2 s really became annoying for users.

In [8], a web-based platform enabling a synchronized playout of digital pictures and video clips across separate users is presented. It allows sharing the navigation control commands and includes interfaces with voice conferencing tools and social networking sites (Facebook) to enable interaction between users. It also provides IDMS capabilities, by adopting an SMS (*Synchronization Maestro Scheme*) sync control scheme [1], with a centralized server in charge of the exchange of sync information. The platform has been exclusively developed by using web technologies, and it yields a straightforward, relatively loose sync, in which "*timing discrepancies amounting up to a handful of seconds might occur across participating sites*". Their authors argue that such sync levels may be acceptable for digital entertainment purposes and recreational applications for residential users, according to a qualitative user-centred research study conducted in [9]. However, they recognize the limitations of the adopted solution and that more accurate sync levels should be provided in order to use that platform for real-time interactive scenarios.

In [10], another framework to enable social viewing is proposed. It makes use of an IDMS-enabled engine developed in [11] to enable shared media consumption of document-based media. That IDMS solution consists of synchronizing the document clock across separated clients and making use of either SMS or M/S sync control schemes. It does not employ globally synchronized clocks, but makes use of a virtual clock sync algorithm to compensate for the drift between the involved clocks by estimating the RTT (Round-Trip Time) between them. The engine also includes a feature to allow a shared execution of the navigation control commands. The framework integrates a native testbed developed in [12] to enable interaction via a multi-party audio/video conferencing tool.

In [13, 14], a platform to enable shared video watching using MPEG DASH is presented. It consists of a communication model to establish and manage different groups of clients, and an IDMS solution to synchronize the playout timings within each group. The solution adopts a DCS sync control scheme to exchange sync information, uses NTP for clock sync and an AMP (Adaptive Media Playout) technique for adjusting the playout processes.

---

[2] Master/Slave is the terminology used traditionally in related works and, therefore, we have maintained it in this section. As it has been discussed to be troublesome (https://www.theserverside.com/opinion/Master-slave-terminology-alternatives-you-can-use-right-now), in this paper, although we propose M/S-based sync schemes, we use the Master/Follower terminology.

[3] https://tools.ietf.org/html/rfc5905 (last access: July 2020)

The work in [15] presents a similar web-based platform to the one in [13] to enable shared video watching for independent groups of clients by also using DASH. Its IDMS solution adopts an M/S sync control scheme, makes use of global clock sync solution (NTP or the timing mechanism proposed in [16], described later), and also adopts AMP for playout adjustments.

The work in [16] proposes relevant web-based components to support the required sync and interaction features in platforms and scenarios for social viewing. These components include a programming model and JavaScript APIs (Application Programming Interfaces) based on an external timing approach to provide a so-called *shared motion* in both single- and distributed multi-device web-based scenarios. The shared motion is provided by an online central timing reference source, which not only provides a common timeline to the connected elements but also enables shared navigation control commands. It relies on an M/S sync control scheme. Results showing sync levels of around a few tens of milliseconds for multi-device playback scenarios are presented, but no description of those distributed scenarios is provided.

Finally, it should be mentioned that other public and commercial platforms/tools have provided, or currently provide, media sharing and interaction features. On the one hand, four public tools providing shared video watching capabilities are highlighted: *Yahoo! Zync*[4], *Watchitoo*[5], *Synchtube*[6] and Facebook's *Watch Party*[7]. The first three tools are not currently available. Nevertheless, their features could be easily found on the Internet and in previous publications. *Yahoo! Zync* [17, 18] was a shared video watching tool developed as a plug-in for *Yahoo! Messenger*, thus providing access to an implicit social network. It enabled synchronized viewing and sharing the navigation control commands, while interacting via text chat. *Watchitoo* was a web-based application for shared video watching, while enabling text chat, transmission of sticky notes, and audio/video conferencing tools. It also allowed sharing the navigation control commands and the volume level settings. It has been rebranded as *newrow*[8], a commercial service for e-learning purposes. *Synchtube* enabled synchronized playback of video clips and slides, while sharing the navigation control commands, and additionally providing text and video chat tools. *Watch Party* enables users to watch videos together in a synchronized manner, while text chatting, but only for videos hosted on Facebook. In *Watch Party*, only the host of each session can add videos to the watch queue, control the playback and suggest additional videos, although co-hosts can be added.

Similarly, other commercial and open-source tools for media sharing and interaction exist. With respect to commercial solutions, six of them are highlighted: *Cisco's WebEx* platform[9], *Adobe Connect*[10], *Gotomeeting*[11], Google Meet[12], Microsoft Teams[13] and Zoom[14]. With respect to open-source solutions, Big Blue Button[15] is highlighted. Most of them provide media sharing options, such as screen sharing, while enabling interaction via text

---

[4] *Yahoo! Zync*. https://bit.ly/36A3aRA (in Spanish) (last access: July 2020).
[5] The features of *Watchitoo* can be found in YouTube videos, as in https://youtu.be/vRoVOCmTghM (last access: July 2020).
[6] *Synchtube*, https://angel.co/synchtube, last access: July 2020).
[7] Facebook Watch Party https://techcrunch.com/2018/01/17/facebook-watch-party/ (last access: July 2020).
[8] *Newrow*: https://www.newrow.com/ (last access: July 2020)
[9] *Cisco's Webex:* http://www.webex.com/ (last access: July 2020).
[10] *Adobe Connect:* https://www.adobe.com/products/adobeconnect.html (last access: July 2020).
[11] *Gotomeeting*: https://www.gotomeeting.com/ (last access: July 2020)
[12] Google Meet: https://meet.google.com/ (last access: July 2020)
[13] *Microsoft Teams*: https://www.microsoft.com/en/microsoft-365/microsoft-teams/ (last access: July 2020)
[14] *Zoom*: https://zoom.us/ (last access: July 2020)
[15] *Big Blue Button*: https://bigbluebutton.org/ (last access: July 2020)

and audio/video chat channels. However, these media sharing features are not primarily targeted for highly interactive distributed video watching, but for online meetings and collaborative tasks, in which media sync solutions become less crucial. For all these platforms, it is unclear whether and how IDMS and Inter-Stream Sync solutions are provided, and if so, the accuracy levels that are achieved, as it is not specified in their references/websites. They also provide limited support for social presence and integration with Social Networks.

Table 1 provides a summary of, and comparison between, the features and functionalities provided by the aforementioned related research-oriented and public platforms identified in the SoA review. Likewise, as the provided IDMS solution is a key element of the social viewing platforms, Table 2 summarizes the IDMS related features provided by the existing platforms, which are also compared to the improved ones provided by Wersync (described in Section 5). On the one hand, these two tables provide an overview of key features that are missing or need to be improved in existing platforms (e.g. support for IDMS, integration with Social Networking sites, shared volume level, etc.), thus revealing the need for further research. On the other hand, the features listed in these tables were then used as discussion input in the conducted user-centric activities to validate their need, and thus decide on their adoption and/or refinement for Wersync, as well as a starting point for the identification of new required features (e.g. shared pointer/board, exchange of files, etc.). This is further elaborated in Section 3.

2.2 Studies on Social TV

Previous studies ([18, 19-23]) have investigated and highlighted the benefits that Social TV scenarios can provide to users, such as the feeling of being together, an improved sense of connection and improved relationships. To successfully provide them, the availability of communication channels between users and of IDMS functionalities have also been identified as key requirements in [19, 21], and in [7, 18], respectively. Regarding the availability of chat communication tools and their modality in Social TV scenarios, several studies have been conducted in the past [7, 20, 22]. In [20], it was found that text chat can result in distraction for users with low typing ability, who prefer to use voice chat tools. However, text chat is preferred by skilled users, as they can chat and watch/hear simultaneously, without the interference of other participants' voices, which may result in some parts of the content being missed, and thus becoming annoying. In [22], it was also concluded that text chat is the preferred communication modality, as there is no need to respond immediately and it takes less energy. In [7], it was also highlighted that both text and voice chat provide high levels of togetherness.

As for the relevance of IDMS in Social TV use cases, it was found in [19] that the interaction between remote users tends to happen during silent periods and/or scene changes, and that the conversations mostly evolve around the TV content. Accordingly, if significant delay differences exist, the naturalness of the conversations will be lost, resulting in users' annoyance. In this context, controlled experimental setups have analyzed the effect of the existence of different delay levels on the QoE (Quality of Experience) in Social TV scenarios, in which remote users can interact via text and/or voice chat ([7, 24]). In [7], couples interact while watching a TV quiz show. In the conducted experiments, various sync conditions, with different asynchrony levels, ranging from 0 s to 4 s, in steps of 500 ms, are forced and presented to participants in a randomized order, by enabling one of the two interaction channels in each test. The results indicated that asynchrony levels up to 1 s might not be perceptible by users while communicating using audio chat, although a low percentage of participants noticed the lack of sync when setting the lowest delay offset of 500 ms. However, asynchrony levels above 2 s became annoying, regardless of the chat modality. It was also detected that users with high text chat activity notice sync differences similar to those using voice chat. Similar results were obtained in [24] in a shared football watching experience. In addition, a subjective quality assessment was conducted in [25] to determine the tolerable asynchrony levels between interactive media clips played out by remote users. It was concluded that asynchrony levels of 400 ms do not have an impact on the QoE, but levels of 750 ms are already noticeable and can degrade it in these scenarios.

As summarized in [4], chapter 1, the thresholds determined in those studies are much lower than the delay variability ranges in current networks, which reflects the need for accurate media sync solutions.

3. Selection of Required Features

Before proceeding with the design of the platform, an initial phase was devoted on accurately determining the required features to be implemented in Wersync. This section firstly details the followed methodology that led to the design options, highlighting the novel aspects of Wersync, and then the rationale on using web-based technologies and components for its development is provided.

3.1 Methodology and Process

Basically, the selection of features of the presented platform was driven by four input sources:

i) Thorough review of the state of the art (Section 2), taking also into account the insights from these works, if experiments were conducted.
ii) Results from the study conducted by authors in [2] to obtain insights regarding the consumption habits, needs, preferences and/or expectations of consumers in different interactive scenarios, including social viewing.
iii) Assumptions and suggestions from our own experience in the field, by re-shaping some existing features and identifying other potential ones.
iv) User-centric activities (focus groups and interviews) with (up to 14) experts to get valuable feedback from them, and to validate the gathered insights up to then, as well as to refine and/or possibly extend the list of required features.

The user-centric activities were in turn divided into three rounds of focus groups and interviews:

i) With 4 international experts on Social TV (meeting held at a related international conference in June 2017).
ii) With 6 researchers from the authors' university with expertise on Social TV (2), audiovisual communications (2), UX design (1) and Social Media (1) (meeting at our university in July 2017).
iii) With 4 experts in teaching innovation (meeting at our university in July 2017). The meeting was organized due to collected insights about the applicability of the platform for synchronous and collaborative e-learning.

Features / Functionalities	**Vaishnavi et al. [3]**	**Wijnants et al. [8]**	**Jansen et al. [10]**	**Rainer et al. [13, 14]**	**Pauwels et al. [15]**	**Arntzen et al. [16]**	**Wersync**	**Selection/ Adoption Reason**
Support for Independent Groups	N	-	-	Y	Y	-	**Y**	SoA Review, User-Centric Activities
IDMS	Y	Y	Y	Y	Y	Y	**Y**	
Session Information	N	Y	-	N	N	N	**Y+**	
Shared navigation control commands	Y	Y	Y	Y	Y	Y	**Y+**	SoA Review, Authors' suggestion, User-Centric Activities
Shared volume level	N	-	N	N	N	N	**Y**	SoA Review, User-Centric Activities
Ad-hoc private text chat	Y (non-secure)	N	N	N	N	N	**Y (secure)**	SoA Review, Results from study in [2], and User-Centric Activities
Integration with Social Networking sites	N	Y (Facebook)	N	N	N	N	**Y (Twitter)**	
Audio chat	Y	Y	Y	N	N	N	**Y**	
Audio + Video chat	N	N	Y	N	N	N	**Y**[16]	
Dynamic Role Switching	N	-	N	N	N	N	**Y**	*User-Centric Activities*
Shared pointer	*N*	-	*N*	*N*	*N*	*N*	**Y**	
Shared board	*N*	*N*	*N*	*N*	*N*	*N*	**Y**	
Exchange of Files	*N*	*N*	*N*	*N*	*N*	*N*	**Y**	
HTTP Adaptive Streaming (HAS) / DASH Support	N	N	N	Y	Y	-	**Y**	SoA Review, Authors' suggestion, User-Centric Activities
(Exclusive) Use of web-based technologies	N	Y	N[17]	Y	Y	Y	**Y+**[18]	See Section 3.3

a) Comparison between research-oriented platforms and Wersync

Features / Functionalities	***Yahoo Zync!***	***Watchitoo***	***Synchtube***	***Facebook Watch Party***	**Big Blue Button**	**Presented Platform (Wersync)**
Support for Independent Groups	Y	Y	Y	Y	Y	**Y**
IDMS	Y	Y	Y	-	-	**Y**
Session Information	-	Y	-	Y	Y	**Y+**
Shared navigation control commands	Y	Y	Y	Y	Y	**Y+**
Shared volume level	N	Y	N	N	N	**Y**
Ad-hoc private text chat	N	Y	Y	N	Y	**Y (secure)**
Integration with Social Networking sites	Y	-	N	Y	N	**Y (Twitter)**
Audio chat	N	Y	Y	N	Y	**Y**
Audio + Video chat	N	Y	Y	N	Y	**Y**
Dynamic Role Switching	-	-	-	*Y*	*Y*	**Y**
Shared pointer	*N*	*N*	*N*	*N*	*Y*	**Y**
Shared board	*N*	*N*	*N*	*N*	*Y*	**Y**
Exchange of Files	*N*	*N*	*N*	*N*	*N*	**Y**
HAS / DASH Support	N	N	N	Y	N	**Y**
(Exclusive) Use of web-based technologies	N	Y	-	Y	Y	**Y+**[19]

b) Comparison between other public platforms and Wersync (same adaption reasons as in the previous table)
[-] Means not specified in the references; [+] Means improved performance and/or extra options (see footnotes)
NOTE: Rows with text in *italics* refer to novel required features identified in the user-centric activities

Table 1. Features included in related platforms and in Wersync

---

[16] In Wersync, modern and standard web-based technologies, such as WebRTC, are used for the integration of the audio-video conferencing technologies, unlike in the platform in [10], which makes use of native tools.

[17] The framework in [10] only makes use of web-based technologies for shared media consumption, but not for its embedded audio-video conferencing tools.

[18] Wersync makes use of modern and standard web components, compared to the ones in [8] and [10] that make use of obsolete components, which may lead to incompatibility and inter-operability issues.

[19] Wersync makes use of modern and standard web components, compared to the ones used in *Yahoo! Zync* and *Watchitoo*, which are obsolete and non-standard components, which may lead to incompatibility and inter-operability issues. Compared to *Watch Party*, not only does Wersync provide support for videos hosted on the platform, but also for external ones.

Features / Functionalities	Vaishnavi et al. [3]	Wijnants et al. [8]	Jansen et al. [10]	Rainer et al. [13, 14]	Pauwels et al. [15]	Arntzen et al. [16]	Presented IDMS Solution
Support for Independent Groups of Destinations	N	-	N	Y	Y	-	**Y**
Clock Synchronization	Y (NTP)	N	Yes (Ad-hoc Solution)	Y (NTP)	Y (NTP or [16])	Yes (Ad-hoc Solution)	**Ad-hoc Virtual Clock (+NTP)**
Control Scheme	DCS and M/S Schemes	SMS	SMS and M/S Schemes	DCS	M/S Scheme	M/S Scheme	**Dynamic M/S Scheme (Extensible)**
Dynamic Role Switching	N	-	N	N	N	-	**Y**
Shared navigation control commands	Y	Y	Y	Y	Y	Y	**Y**
Adaptive Media Playout (AMP)	N	N	N	Y	Y	-	**Y**
(Exclusive) Use of web-based technologies	N	Y	N	Y	Y	Y	**Y+**[20]

[-] Means not specified in the references; [+] Means improved performance and/or extra options (see footnotes)

Table 2. Features of the IDMS solutions integrated in the related platforms and in Wersync

The methodology to extract and validate the requirements in all these focus groups was as follows:

i) The context and objectives of the research work, together with the associated challenges, were introduced to the experts.

ii) The results from [2] and the conducted SoA analysis (including mockups from other existing platforms) were summarized to the experts.

iii) The preliminary list of potential features for social viewing, extracted from the SoA analysis (Table 1), was presented and described to the experts. Note that these required features were mostly non-functional high-level requirements that directly or very easily determine the required technological features to be provided (e.g. support for multiple groups, shared volume level, etc.). Note also that the researchers previously validated the feasibility of developing/integrating these features in the platform to be developed.

iv) A demo of a preliminary, but fully functional, version of the platform, was showcased to the experts.

v) Mockups illustrating additional potential features and GUI design alternatives were shown to the experts.

vi) A semi-structured interview, driven by a script, was conducted with the experts. The sessions were recorded (with the experts' consent) for posterior analysis, but also the two researchers in charge of the focus groups made annotations on the key comments and insights. Also, the experts were encouraged to suggest further features and applicability scenarios.

vii) Finally, the key discussions and lessons learned during the focus group were reviewed with the experts to ensure they were aligned with them, and thus with the extracted and validated requirements.

In general, the experts on Social TV mostly agreed with the presented requirements and related features, and no one was concerned with the addition of these features. The experts also provided useful feedback on the "session information" to be provided and on the "integration with Social Media platforms" and "social presence" features. They also suggested extra features, like "shared volume control" and "shared pointer". These suggestions were welcome by most of the experts.

Given the rising relevance of collaborative media sharing platforms, and the potential of Wersync to additionally accommodate such use cases, meetings with experts in this field were additionally organized. They also confirmed

[20] See previous footnote 16.

the appropriateness of most of the features discussed with the Social TV experts, but additionally suggested the ability to have a “shared board” over the video region, and to “exchange files” through the chat tool.

The experts on both disciplines also pointed out on the necessity of providing cross-platform support and a responsive GUI. These most low-level requirements were at the end transformed into the “use of web technologies” for developing the platform (Section 3.3).

3.2 Derived Requirements for Social Viewing, and Adoption in Wersync

As a result of the followed methodology and process (Section 3.1), a set of key requirements (i.e. required features) for social viewing has been gathered. These requirements are listed and briefly described below, also including suggested improvements. They are divided into requirements from existing platforms, with or without suggested improvements, and new requirements gathered via the user-centric activities.

Requirements for social viewing derived from the SoA analysis and conducted survey [2]:

- *Support for Independent Groups*. It refers to the ability of creating and managing independent shared sessions, and controlling them independently. This feature is included in many of the existing related platforms, and its need was confirmed in the user-centric activities.
- *IDMS*. It refers to the sync between the video playout processes in all the participants’ devices to enable coherent shared media sessions. This feature is provided in some of the related platforms, but they do not generally achieve accurate sync levels (e.g., [6, 8]), which is key to efficiently support interaction and collaboration in social viewing. This requirement was observed from the SoA review, from the results in [2], and its need was confirmed in the user-centric activities.
- *Session information.* For each active session, useful information needs to be provided (e.g. names or nicknames of its members). This feature is provided by few related platforms (e.g. [8]), and its need was confirmed in the user-centric activities. Indeed, the experts recommended providing extra session information, like the members’ roles, the session name, and the content being watched. They also suggested to include some feature to be able of dynamically showing and hiding this information in the GUI of the platform.
- *Shared navigation control commands of the media player*. This feature refers to the ability of issuing a playout control command (e.g. play, pause, seek), and that command being also issued in the rest of the players of the shared session. It is supported by many of the related platforms, and its need was confirmed in the user-centric activities. In addition, the experts highly recommended to integrate this feature with the IDMS one, so that highly accurate sync can be kept, even after the execution of these commands. It was also recommended to enable this feature only for one client at a time (Master client) to prevent from chaotic situations (discussed later).
- *Shared volume level settings*. Similarly to the previous requirement, it refers to the ability to transfer the volume level settings of the Master client to all other clients. This feature was also provided by *Watchitoo*, and it was recommended in the user-centric activities.
- *Text Chat Channels*. This feature enables the interaction between users via text chat, which can be provided via an ad-hoc and private chat channel and/or via the integration of a Social Network chat tool, like Facebook or Twitter. This feature is provided in related platforms, but they differ in the type of chat channel they integrate. The reason of including an additional private chat is because various limitations and constraints of using Social Media platforms, such as Twitter, in media sharing applications have been identified in [26, 27], such as: i) high

end-to-end delays; ii) low flexibility for embedding and retrieving sync metadata (e.g., timelines); iii) high dependence on third-party components and infrastructure; iv) non-guaranteed scalability and availability; v) need for filtering and refresh mechanisms, etc. On the one hand, a private chat tool provides a set of advantages, such as: i) lower latency; ii) higher flexibility for adding and interpreting timestamps (i.e., for achieving sync); iii) *"private"* chat rooms for each group (instead of having a *"public"* chat room when using e.g. Twitter); iv) higher flexibility to secure the communication channel. On the other hand, chat tools from Social Networks are well-known by most users, they allow to filter messages by words (e.g., by using hashtags in Twitter) and to interact with external participants during the session. The surveys and interviews in [2], and the user-centric activities, did not reveal clear differences between the preferences regarding the use of the above types of text channels. That is the reason why both approaches are considered valid options for social viewing, and can even be jointly integrated (as in this work). Twitter was the preferred platform in [2] and in the user-centric activities (e.g. experts highlighted that it can be used for sending automatic invitations), so it has been integrated in Wersync.

- *Audiovisual (AV) chat channels*. They allow to more naturally interact than text based chat channels. Other platforms, such as the one in [10], also provide such a functionality, but make use of native tools. Likewise, the platform in [8] only enables interaction via voice. The convenience for this requirement is supported by recent studies (see Section 2.2), by the results from [2], and by the user-centric activities.

<u>(New) Requirements for social viewing derived from the User-Centric Activities:</u>

- *Dynamic Role Switching*: If some of the previously identified features (e.g. shared playout control commands) were available to all participants in a shared session at any time, it might result in chaotic situations. Therefore, it was suggested in the user-centric activities to only enable those features to the Master User (e.g., the moderator or instructor). In order to let all the participants make use of them, a dynamic role switching policy needs to be provided.
- *Shared pointer over the video player*. It refers to the ability of displaying a (mouse, touch) pointer over the video player of the Master User, and also displaying the pointer and its exact instantaneous position in all other players. That way, the user can e.g. point at a specific regions of the scene while the video is being played out for a richer interaction.
- *Shared board over the video player.* At a specific moment, the video can be paused (at all players) and a shared board can be activated to draw, make annotations over the video screen or to point to specific parts of the video. That specific requirement was suggested by the experts in teaching innovation in the user-centric activities.
- *Exchange of files*. It refers to the ability to exchange files (e.g. related pictures and/or documents), e.g. via the text chat tool. The need for this requirement was highly recommended by both the experts on social viewing and on teaching innovation in the user-centric activities.
- *Upload of content*. It refers to the ability to allow only authorized users uploading their video files in order to watch them together with the group they wish, and not just rely on available video on other online platforms (e.g. Youtube). The uploaded content pieces need to be listed once a new session is being created. The need for this requirement was also highly recommended by both the experts on social viewing and on teaching innovation.

As far as authors know, all those requirements derived from the user-centric activities are not provided by any of the analyzed existing social viewing platforms, but their need has been identified in this work. Two extra requirements were determined from the insights from [2] and from the user-centric activities:

- Need to *maximize the support* in terms of network connections, consumption devices and platforms (i.e. Operating Systems). This has been met in this work by relying on web-based technologies for both media delivery and the platform development (Section 3.3).
- The need to provide *personalization options* for a best-fitted and comfortable experience.

Overall, up to 14 requirements for social viewing have been determined via the insights from a set of research activities. These requirements can serve as a guideline for any researcher and developer interested in social viewing platforms and scenarios, and have specifically driven the development of the platform presented in this work. The experts from the user-centric activities raised concerns about potential cognitive load, distraction and usability issues that the implementation of all requirements, including the personalization options, might involve. Due to the optional nature of most of the identified features, the possibility to hide the associated controls and the potential benefits they can provide, all of them have been integrated in the platform. This also increased the relevance of a further key requirement: the availability of an *appropriate layout and GUI*. The users' acceptability towards these required features in Wersync have been evaluated via subjective testing (Sections 6 and 7).

As a summary, Table 1 lists all derived requirements, indicating the ones provided by existing social viewing platforms, the ones previously available but with room for improvement, and the newly derived ones, also reflecting how their need and/or convenience was observed. This table also serves to identify missing aspects in the existing platforms, comparing them with the one presented in this work that meets all the derived requirements, even including improved implementations for many of the requirements provided in the existing platforms (e.g. ad-hoc secured chat channels, availability of session information, support for media sync, etc.).

3.3 Use of web-based technologies and components

Two main options can be chosen for the development of media platforms: the use of either native (or platform-specific) or web components. The implications of adopting each option, in terms of key factors, such as support, ubiquity, development, maintenance, and usability issues can be found in [26]. That study by the authors provides the main arguments that support the decision of having developed a web-based platform, instead of a native application. In summary, despite of the slight advantages of using native applications in terms of more complete support for specific platforms, performance and usability, the differences are not so significant, especially regarding the development of media consumption platforms, which is the scope of this work. Besides, the use of Web applications can contribute to reduce the costs in terms of development, maintenance and distribution (principle of '*build and update once, run anywhere*'). Moreover, it can guarantee a more flexible and successful cross-platform, cross-device and cross-browser support, and also cross-network support, as the Web traffic is not typically sensitive to firewall blocking policies and NAT traversal issues. This last aspect contributes to a better ubiquity of the media applications and services to be deployed. In relation to this, the presented platform not only allows the integration of HTML5 videos, but also of videos delivered via DASH, which has become the dominant technology for adaptive video delivery. As indicated in Table 1, just few other platforms support this standard technology, or a similar HTTP-based Adaptive Streaming (HAS) one.

Finally, although web components have not been typically conceived for providing precise timing and timely responsiveness [16], the recent advances in web technologies allow developers to overcome these issues, as proved in [14, 15, 16] and later in this work.

All the above considerations support the decision of choosing web-based components for developing Wersync.

4. Wersync Platform

In this section, firstly, an overview of the main web pages of Wersync is presented, and, then, the main functionalities of the platform are briefly described in different subsections. A motion graphics showing the research problem and the provided solution with animations, and a demo video showing the capabilities of Wersync can be watched at https://youtu.be/YGJRwUisSyU, and https://youtu.be/hc0Kgu6zr74, respectively. Several GUIs have been previously designed for preliminary prototypes ([28, 29]), and, finally, an attractive, intuitive and responsive GUI has been professionally designed and implemented. Figure 1 presents the home, log-in, session creation and joining web pages. The home page includes buttons to create sessions, join ongoing sessions, and upload (only authorized users) video files (automatically converted into the MPEG DASH format, with configurable parameters). Figure 2 presents an example of the main page for the Master User once he/she has joined a shared session with three other (Follower) participants. On the upper part, session information is provided, such as the name of the session, participant's nickname, a dropdown list with all participants' names, participant's role (Master or Follower) and, if the participant is a Follower, the nickname of the Master User. This block can be hidden to enlarge the area of the video player.

4.1 Session management

4.1.1 Session creation, joining and leaving

The use of *dedicated* rooms for each session contributes to guarantee privacy in Wersync. An authorized user can create new sessions (Figure 1), simply providing the username (nickname), a name for the session, and selecting the video to be watched (from a dropdown list with all the available videos), and becoming its Master User. Other users can join active shared sessions, by only providing the name of the ongoing session and their nicknames (Figure 1), and, then, all the other fields are automatically filled. A message is sent to the Master User of that session, who can accept or reject them. If accepted, all the other members will be notified with a text message in the private chat window. When a participant leaves the session, it will also be notified to the other members.

In each session, a participant can request a change of its role, from Follower to Master (*Dynamic Role Switching* feature). Then, the current Master User will receive a notification in order to accept or deny the role change.

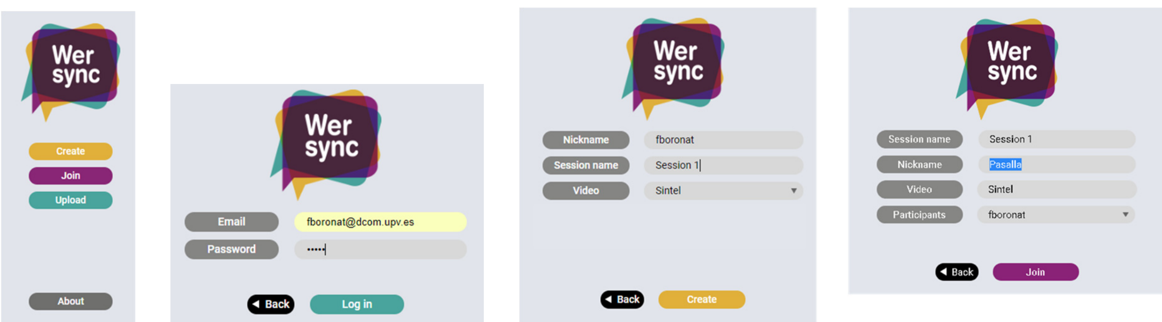


Figure 1. From left to right: Wersync home, log-in, session creation and joining web pages

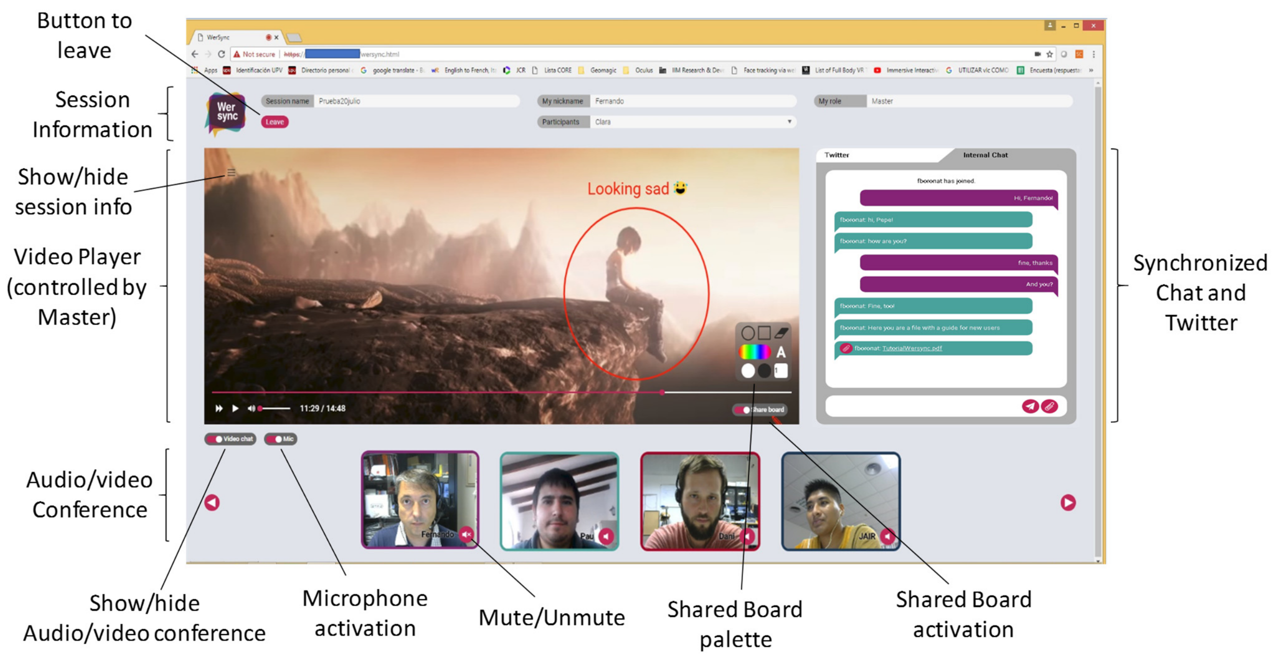


Figure 2. Example of the Media Client's main web page of Wersync (for the Master User)

4.1.2 Social presence mechanisms

Participants in active sessions can send other users both public or private invitations to participate in the session via different methods: e-mail, a social network service or any other communications service. In Wersync, in order to foster the participation of users in active sessions, two social presence [8] mechanisms are provided. Firstly, a list of active sessions, their participants' nicknames and the video being watched are provided. Secondly, an external presence mechanism is provided by linking Wersync with Twitter. It allows external users to be aware of their Twitter contacts' activities on Wersync, encouraging their participation. The Twitter-based chat tool (explained later) has been extended to invite specific followed users by sending them direct messages with invitations (Figure 3, right). In any case, the invitations should include, at least, the homepage of Wersync and the name of the session, or just a direct link to that session. Additional data can also be provided, such as a description of the video content, period of time, type of session, expected type of participants and the hashtag to be used in Twitter. Despite these mechanisms, Wersync takes privacy into account, by controlling the users who can join ongoing sessions, and by using private interaction channels, including the use of data encryption solutions.

4.2 Distributed and synchronized media consumption

On the one hand, in Wersync the playout of the selected video content is synchronized across all the Media Clients by using a developed ad-hoc IDMS solution. The Master User's Media Client is the only one that shows the video playout control options, allowing the Master User to trigger playout control commands (play/pause, skip, fast forward, mute/unmute and volume change). These commands are forwarded to the other (Follower) clients via a low latency secured channel and quickly executed for them. On the other hand, the exchanged private text messages are presented to all the participants in the correct instant to avoid inconsistences and/or spoiling during the watching experience. An Inter-Stream Sync solution between video and private text streams in each Media Client has also been implemented. Both sync solutions are explained in Section 5.2.2.

4.3 Social interaction and/or collaboration tools

In this Section, the additional tools of Wersync to enable social interaction and/or collaboration are presented.

4.3.1 Text tools and Files Exchange

As mentioned, Wersync includes two text-messaging tools: a Twitter-based one and an additional private, secured (encrypted) and synchronized one (Figure 3). In the latter, the messages are shown in different background colors to identify participants. It includes an option (paperclip icon) to send files to other participants in the session.

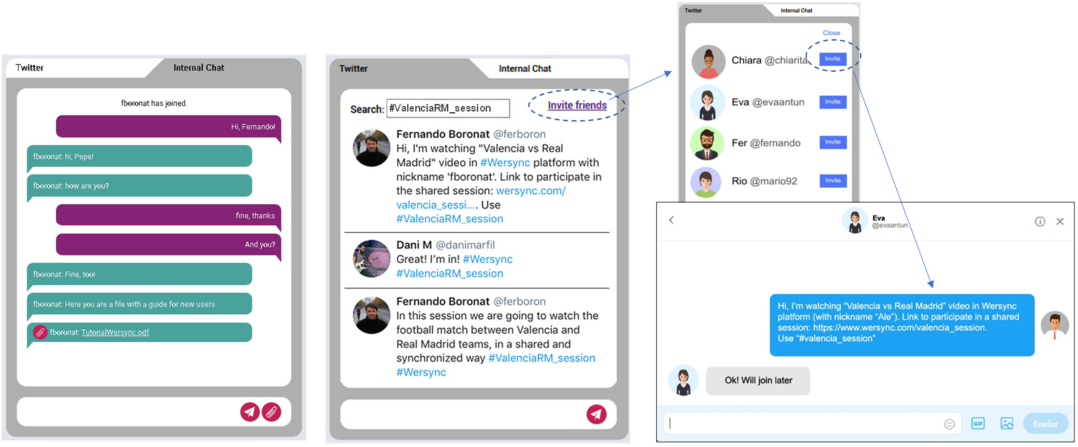


Figure 3. Ad-hoc synchronized (left) and Twitter-based (center) chat tools. Twitter direct messages for invitation (right)

4.3.2 Shared board tool

The Master User can activate and share a drawing board (*Shared board* switch control). When activated, the video is automatically paused and a palette with some options appears, including the possibilities of selecting the color of drawing or text; changing size or thickness, writing text, drawing lines, forms (ellipses or rectangles); or removing them. This tool can also be used to share a pointer. Figure 2 shows an example of its use.

4.3.3 Audio/video conference tool

AV chat channels between the Media Clients are also provided. Although less preferred than text channels [2], they provide more natural, interactive and comfortable communications. Only the Master User can initiate them (*Video chat* switch control) between participants in the session, and the images from the camera of each participant will appear at the bottom of the web page. Participants can switch their microphones on/off, and mute/unmute the audio from specific participants. In order to make the identification of the participants more intuitive, their camera views have an outline with the same color as the one used for the background color of their chat messages.

5. Architecture and implementation details

In this section, the main components of Wersync and some implementation details are provided.

5.1 Architecture

Figure 4 presents an overview of the architecture of Wersync, which includes two types of entities: Media Clients and Servers. On the one hand, *Media Clients*, which are devices with keyboard and mouse (or equivalent), camera and microphone, used to access the web pages of Wersync, include several modules: *Session Client* module, which is in charge of creating, joining and/or leaving sessions, and of exchanging session information with the Session Manager; *Video player* module, which is in charge of playing the video; *Sync Client* module, which is in charge of the playout sync processes; and several modules for user interaction and/or collaboration (text, board, AV conference tools…). On the other hand, the following Server entities are included: *Web Server*, which hosts the Wersync web site and all the associated resources; *Media Server*, which provides all the media content and associated metadata; *Session Manager*, which is in charge of the management of the shared sessions and all the compiled session data from clients, and also provides all the signaling capabilities to manage the used *WebRTC*[21] connections; *Communications Server*, in charge of the exchange of data/control messages between entities; *Data Server*, which is a SQL-based server in charge of managing the databases storing media content, session and users information; and *Clock Server*, which provides a common wall-clock reference for all the Media Client entities.

Server entities can be located in different machines or some of them can also be implemented in the same machine. Media Clients communicate with the Session Manager and Clock Server through the Communication Server by using the *Socket.IO*[22] library. WebRTC *DataChannel* component is used for the exchange of data and control messages between Media Clients. Encryption is mandatory for all the WebRTC components and data channels are secured based on DTLS-SRTP (Datagram Transport Layer Security[23] – Secure RTP[24]).

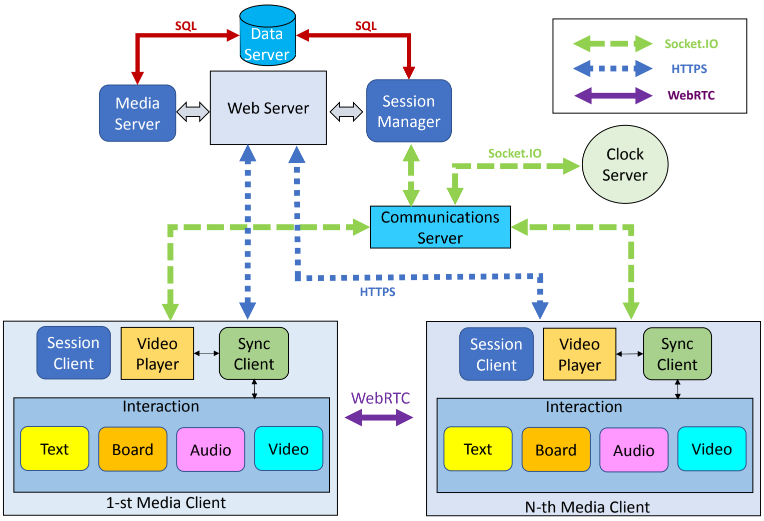


Figure 4. Architecture overview of Wersync

5.2 Implementation details

Some details about the implementation of the most significant features of Wersync are provided in this subsection. All the Servers have been implemented by using *Node.js*[25] except the Data Server that is a MySQL Server[26]. In the Media clients, Dash.js[27] client has been selected for the playback of MPEG DASH content. Several configuration parameters have been defined, which are summarized in Appendix I.

---

[21] *WebRTC* is a free, open project under standardization within Internet Engineering Task Force (IETF) and World Wide Web Consortium (W3C) that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs. https://webrtc.org/ (last access: July 2020)

[22] *Socket.IO* is a lightweight JavaScript library, which enables real-time, bidirectional, reliable and event-based communication between a browser and a (node.js-based) web server. https://socket.io/ (last access: July 2020)

[23] https://tools.ietf.org/html/rfc6347 (last access: July 2020)

[24] https://tools.ietf.org/html/rfc3711 (last access: July 2020)

[25] *Node.js* is an open-source, cross-platform runtime environment, written in JavaScript, for server-side and networking web-based applications. https://nodejs.org/en/ (last access: July 2020)

[26] https://www.mysql.com (last access: July 2020)

[27] Dash.js is a reference client implementation for the playback of MPEG DASH via JavaScript and compliant browsers. https://github.com/Dash-Industry-Forum/dash.js? (last access: July 2020)

5.2.1 Data communications

On the one hand, bidirectional point-to-point encrypted data, audio and video connections between Media Clients are established by using the WebRTC *RTCMultiConnection* library. It needs a signaling server to relate all the users in an RTC session, which has been included in the Session Manager entity. On the other hand, Socket.IO is used for the communications of the Media Clients with the Session Manager, through the Communications Server (Figure 4). The messages related to text chat, shared board, user-generated events (for playout control), and IDMS are exchanged through the *Datachannel* of WebRTC, providing an encrypted communication with very low latency. The first version of the platform ([28, 29]) made this communication through a WebSocket Server, without encryption, and with higher latency (the messages were sent by Media Clients to that server, then processed and forwarded to other Media Clients). There is only one exception: exchanged files are directly uploaded to the Media Server by using *Socket.IO* and only their URLs are sent to all the other Media Clients in the session.

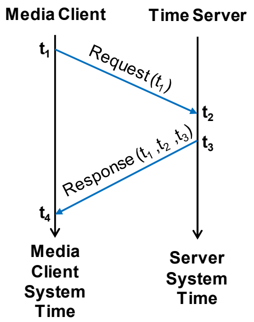
5.2.2 Synchronization

To achieve the two types of sync required in Wersync, three components have been implemented: a virtual clock sync solution, and both Inter-Stream Sync and IDMS solutions.

5.2.2.1 Virtual clock synchronization

A simple ad-hoc virtual clock sync mechanism has been implemented to recreate the functionalities of typical clock sync protocols (e.g., NTP) in order to obtain a common virtual wall-clock reference. It consists of having a reference Time Server and adopting a request-reply protocol [30].Timestamped messages are periodically exchanged between the Media Clients and the Time Server via the Communications Server. These messages are used to estimate the RTT delays and skews between the local clock of the Media Client and the reference clock of the Time Server. Figure 5 illustrates the messages exchange process. Initially, the Media Client sends a *Request* message to the Time Server, including its origination timestamp, t_1 (taken from its local clock). After receiving that message, the Time Server will send a *Reply* message, including t_1, and two additional timestamps: the reception timestamp of the *Request* message (t_2) and its transmission timestamp (t_3), both taken from the clock of the server. Finally, the Media Client will register the reception time of the *Reply* message, t_4 (taken from its local clock). With this information, the Media Client can measure: the end-to-end RTT, by using Equation 1; the RTT for such control messages (by eliminating the processing delay at the Time Server), by using Equation 2; and the one-way delay for such messages (assuming symmetric network delays), by using Equation 3. Independently of the clock technology in use, it is possible to estimate the clock skew between the two involved entities by applying Equation 4. By regularly executing this protocol, it is possible to estimate and compensate for the clock skew between the Time Server and each Media Client in the sync process (Equation 5). It allows time-aligning the virtual clocks of the Media Clients with the one of the Time Server, even if they do not support the same technology for clock sync, and without the need for installing additional plugins or modules. This mechanism is commonly known as *virtual clock sync* [5], and this work has brought it to the web domain by designing an adaptive and lightweight protocol.

A periodic message exchange interval (e.g., 5 s or 10 s) can be set in the *Time Request interval* parameter (see Appendix I). Moreover, it can be dynamically adjusted according to the number of Media Clients to avoid overloading the server, and according to the required clock sync granularity. The regular estimation of the delays and clock offsets will allow for continuously updating such values and dealing with temporal fluctuations.



$$RTT_{C \to TS, end-to-end} = t_4 - t_1 \quad (1)$$

$$RTT_{C \to TS} = (t_4 - t_1) - (t_3 - t_2) \quad (2)$$

$$d_{C \to TS} = \frac{RTT_{C \to TS}}{2} = \frac{(t_4 - t_1) - (t_3 - t_2)}{2} \quad (3)$$

$$Clock\_skew_{C \to TS} = t_2 - t_1 - \frac{(t_4 - t_1) - (t_3 - t_2)}{2} = t_2 - t_1 - d_{C \to TS} \quad (4)$$

$$Client's\ Virtual\ Clock = Client's\ System\ Clock + Clock\_skew_{C \to TS} \quad (5)$$

Figure 5. Control messages exchange to estimate latency and clock skews

5.2.2.2 Distributed media playout synchronization

In order to achieve a synchronized playback among all the distributed Media Clients of each group, a lightweight, adaptive and event-driven IDMS solution has been designed and implemented. A centralized M/S sync control scheme has been adopted (a DCS sync control scheme has not been considered, because its use would imply higher complexity and processing load for the clients [1, 31]). One of the Media Clients is considered as the IDMS Master Client and the rest are considered as the IDMS Follower Clients. The IDMS Master Client in each session coincides with its Master User's Media Client. This scheme has a clear drawback: the IDMS Master Client could suddenly leave the session (voluntarily or not). To overcome this, three dynamic master re-election policies have been implemented: i) the oldest active participant in the session becomes the master (default option); ii) the last master (if any) becomes the new master; and iii) the most active chat participant becomes the new master. The policy to be used can be set in the *Master re-election* parameter (see Appendix I).

The IDMS Master Client periodically sends sync control messages in JSON (JavaScript Object Notation) format, called *IDMS messages*, to all the other IDMS Follower Media Clients in the session, via the secured WebRTC *Datachannel*. The transmission interval (e.g., every 1s) of these messages can be configured (*IDMS Report interval* parameter, see Appendix I) and can also be dynamically scaled if the number of IDMS Follower Clients significantly grows. The messages include the IDMS Master Client's current media playout position and virtual time. Upon receiving an IDMS message, each Follower's Media Client calculates the value of the *asynchrony* between its own media playout timing and the one of the IDMS Master Client. We define *IDMS Asyncrony* as the difference between the playout point of a Follower's Media Client regarding the one of the Master Media Client. Highly accurate sync can be achieved by using a mechanism to compensate for the transit delays experienced by the IDMS Messages. If the computed IDMS Asynchrony exceeds an allowable threshold (*IDMS Asynchrony threshold* parameter, Appendix I), the IDMS Follower Client must adjust its playout timing to synchronize with the one of the Master Client.

The playout adjustments can be performed by following aggressive or smooth strategies (depending on the configuration settings). An aggressive strategy is the simplest one and consists of performing playout skips and pauses, with a magnitude of time equal to the value of the detected IDMS Asynchrony. Additionally, the IDMS Asynchrony can also be eliminated by using AMP, smoothly adjusting (i.e., either slowing down or speeding up) the playback rate during a specific time interval (by using the *playbackRate* property of the HTML *video* object) [32]. AMP is much more convenient because it minimizes the occurrence of long-term playout interruptions, which can be annoying for users [33, 34]. That is the reason why in Wersync, like in other recent IDMS solutions (e.g., [13, 31]), AMP has been adopted. The playout rate is speeded up (if the IDMS Asynchrony value is positive, i.e. the Master is ahead) or slowed down (if the IDMS Asynchrony value is negative) a percentage up to 25%[28] (*percentAMP* parameter, see Appendix I, with values between 0 and 0,25). The adjustment interval duration can be calculated by using Equation 6, and after the adjustment time, the original media playout rate is restored to the normal rate.

$$Adjustment\ Time = \left|\frac{IDMS\ Asynchrony}{percentAMP}\right| \qquad (6)$$

In order to avoid too long adjustment periods, which will be surely very annoying for users, two additional parameters have been added in the configuration settings (see Appendix I): *Maximum Allowed Pause* (MAP) and *Maximum Time of Adjustment* (MTA). Figure 6 presents the flow diagram of the playout adjustment process. *Syncing* is a Boolean variable used to avoid making any adjustment when there is already one taking place.

Wersync allows sharing the execution of timestamped playout control commands between Clients, allowing for a fine grained and immediate synchronization when executing them in interactive sessions. Table II-1, in Appendix II, summarizes the exchanged messages used for playout control, IDMS and virtual clock sync purposes.

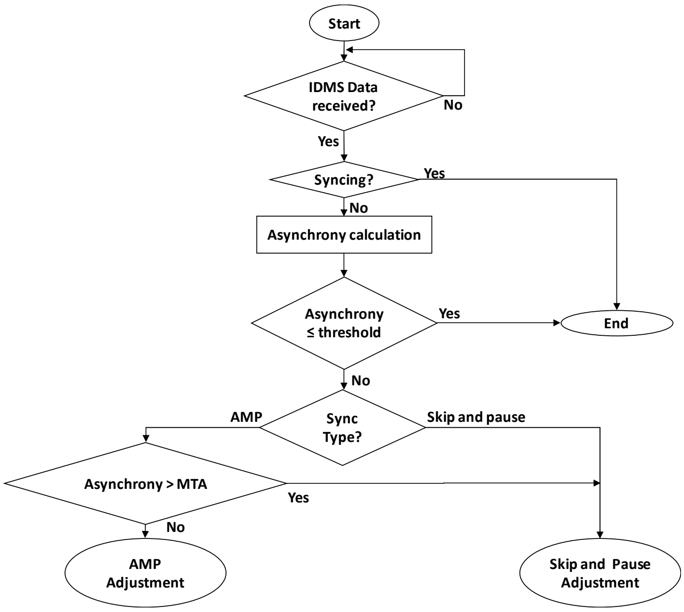


Figure 6. Flow diagram of the playout adjustment process

5.2.2.3 Private chat messages synchronization

In order to synchronize the presentation of private chat messages with the video timeline, another M/S scheme has been adopted for the Inter-Stream Sync solution. In such a process, the video stream is the Master Stream and the text stream is the Follower Stream. Chat messages also includes the position of the video playout (according to the video content timeline) when it was sent by the Media Client. When it is received, the receiver's Media Client checks if the received position of the sender's video playout is more advanced than its own video playout position. If so, the text message is immediately shown to the user. Otherwise, it is stored, and that checking process is

[28] According to [35], if the (increasing or decreasing) percentage is lower than 25%, it is not perceptible by users, and, depending of the content, up to 50% could be tolerated.

repeated periodically (according to the value of the *Inter-Stream Checking Interval* or *ICI* parameter, see Appendix I), e.g., every 50 ms, until it can be shown to the user.

5.2.3 Twitter tool integration

The Twitter-based text messaging tool has been implemented, by making use of the *Twitter API*[29] and without sync with the video stream playout. The use of this API is based on the interaction with an HTTP REST server to which HTTP requests are sent. The *Codebird-js* library[30] in JavaScript has also been used.

5.2.4 Shared board

To implement the shared board tool, the HTML5 *canvas* element has been used. All the events executed by the Master User's Media Client over that element to use the share board controls and related messages are also sent via the secured WebRTC *Datachannel* in JSON format, and are replicated on the *canvas* element of the remote Follower Media Clients once received. Table II-2 in Appendix II summarizes all the messages related to the shared board, their fields and possible values.

5.2.5 Audio and video conference tool

The multi-party audio and video conference channel has been implemented by using WebRTC. Secured WebRTC audio, video and data channels are used. Table II-3 in Appendix II presents the message sent by the Master User's Media Client to the ones of the Follower participants to activate/deactivate them.

5.2.6 Stored Data

A database has been developed in the MySQL-based data server module, with several related tables. Some relevant data can be recorded during the sessions, with research purposes and always with the users' consent, to be processed in order to study the behavior of users when using the platform (out of the scope of this paper).

- *Session_data* table. For each shared session, it stores the session identifier (Id), session name, beginning (creation) instant, duration and session creator user id.
- *Videos_Watched_in_session* table. For each session, it stores the information of the watched videos. For each video watched, it stores the session id, the video id and the duration the video has been watched.
- *Videos* table. It includes, for each video, the video id, the name of the video, the URL of its MPD (manifest) file, and the video characteristics (aspect ratio, width, height, frame rate and duration).
- *Users* table. For each user, it stores the user id, user name, password, age and genre.
- *Masters_in_session* table. For each session, it stores the users that have become Master Media Clients during the session. It stores the session id, the user id and the starting and ending instant being Master User's Media Client (relative to the beginning of the session) for each user holding the Master role during each session.
- *Users_in_session* table. For each session and participating user, it stores the session id, the user id, the joining and leaving instants, and information related with the access device in use, such as Operating System, browser (and version) and language.

---

[29] https://dev.twitter.com (last access: July 2020)
[30] https://github.com/jublo/codebird-js (last access: July 2020)

- *Events_during_playout* table. It stores data about user-generated events affecting the playout continuity, such as play, pause, skip, fast forward, etc. It stores the type of event and the instant in which it took place. Additional information is stored depending on the type of event (e.g. in a skip, initial and final position of the skip are stored).

6. Evaluation

In this paper, it is not the authors' intention to conduct objective or subjective evaluations to compare the performance of the tools/platforms described in Section 2 and Wersync. Wersync includes different innovative features and new technologies regarding the related platforms and therefore, the comparison would not be direct nor fair. Different variables would differ in such comparisons (technologies, features, user interfaces…). Likewise, it is not our goal to compare between variants of technological components for IDMS, but we instead rely on the insights from our previous works for this in this topic (e.g. suitability of sync control schemes [1, 31] and of playout adjustment techniques [32]) and bring this relevant requirement to the web domain, integrating it with interactive features. Having these issues in mind, in this work, two types of assessment have been conducted focused on evaluating Wersync. Firstly, its correct performance, focusing on the achieved sync levels, has been objectively evaluated. Secondly, the sync perception and some essential aspects, such as the GUI of the platform, its usability, the usefulness of its features, its applicability in scenarios of interest, and the awakened interest have been subjectively assessed by 60 users. The assessment scenario, the followed methodology, the involved participants and the obtained results and their discussion are presented in this Section.

6.1 Assessment scenario

The scenario used for the evaluation involves four labs of the authors' University Campus, each with one or several devices as Media Clients (Figure 7). Four 100 Base-T Ethernet LANs interconnected through a Linux-based router are involved. A computer hosting all the server entities of Wersync (PC0) is located in LAN1 together with one computer (PC1) as a Media Client. In both LAN2 and LAN3 there is one computer as other Media Clients (PC2 and PC3). In LAN4, there are several devices: a computer-based Media Client (PC4) and a Wi-Fi Access Point through which a tablet-based Media Client can access the network with a throughput of 76 Mbps. In order to resemble open (non-controlled) scenarios, the Linux-based router, by using NetEm[31], introduces a delay of 100±40 ms, following a normal distribution, and a packet loss probability of 0,1% (both delay and loss rate in only one way, for paths from LAN1 to the other LANs). Additionally, to emulate playout rate deviations between Media Clients, a playout rate increase of 0,01% is forced in the Media Client of PC2, whereas a playout rate decrease of 0,01%[32] is forced in the Media Client of PC4 in order to increase the playout time differences between the Media Clients. No playout rate imperfections have been forced to the other Media Clients, thus having nominal playout rates. Table 3 presents the main features of the hardware equipment involved in the scenario, and Table 8 presents information about the type of content used during the evaluation. Four copies of each video have been generated in different qualities (H.264-AVC; resolutions 1280x720, 854x480, 640x360 and 426x240pixels; 25fps) and then segmented in

---

[31] http://manpages.ubuntu.com/manpages/trusty/man8/tc-netem.8.html (last access: July 2020)

[32] These playout rate variations have been set larger than customary deviations in inexpensive oscillators, which can vary between 10 and 100 ppm [36], in order to force higher asynchronies between the involved Media Clients, and to test if these asynchronies are successfully handled by the IDMS solution included in the platform.

chunks of 3-second duration. Audio is encoded using AAC (48KHz). Ffmpeg[33] and Bento4[34] tools have been used. More details of the process can be found in [37]. The URL of each MPD file is stored in the *Videos table*.

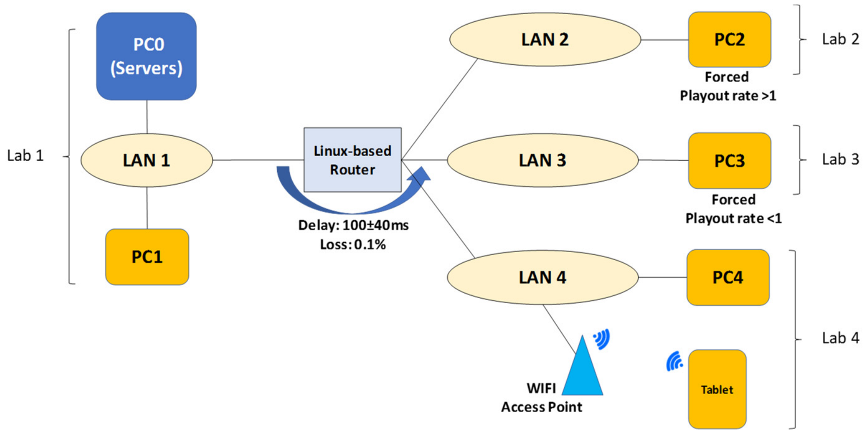


Figure 7. Evaluation scenario

Taking into account the small size of the IDMS-related messages, the viability of using small periods of transmission was considered, in order to achieve accurate IDMS as soon as possible after asynchrony values exceeding the threshold are detected, and still avoiding network congestion. With this in mind, after several tests in the lab, a value of 2 s for the IDMS Report interval was selected. Additionally, an *IDMS Asynchrony Threshold* was selected in order to not exceed the recommended ±400 ms asynchrony threshold for Social TV use cases [25]. Many preliminary tests were carried out with different values and ±100 ms was the lowest threshold value that did not affect the correct performance of the platform, keeping the asynchrony values far below the ±400 ms value.

Name	Specifications
PC0 (Servers/Session Manager)	HP Z620 Workstation, Intel Xeon CPU E5-1650 v2@3.50 GHz, Gigabit Ethernet, Windows 8, 64 bits.
PC1 to PC4 (Media Clients)	Intel Core i5-7500 CPU @3.40 GHz, Gigabit Ethernet, Windows 10, 64 bits with 22" HD LG screen
Tablet (Media Client)	Samsung Galaxy Tab S10
Linux-based router	NUC Intel Celeron N3050 4 GB RAM, Ubuntu 14.04, with embedded Ethernet card and 3 external SMC2209 USB/ETH cards
Access Point	TP-Link TL-WR703N Mini
LAN Switches	CISCO Catalyst 3560

Table 3. Devices specifications

Type of Content	Content
Sports	NBA All-Stars match 2017 (source: Movistar+)
	UEFA Eurocup 2012 final match (source: BBC)
Documentary	World War II documentary (source: National Geographic)
Music	Queen concert at Wembley Stadium
Action film	Avatar movie
Series	Episode s2e22 of Modern Family Series
Reality	Episode of Top Chef reality show (Source: Antena 3)

Table 4. Content description

6.2 Objective evaluation

The components and functionalities of Wersync have been designed and developed with the goal of providing interactivity and sync levels within the range of tens of milliseconds. Different probes and monitoring points have been included in different parts of the platform to test whether the targeted behavior and performance of Wersync has been achieved. In each Media Client, the clock skew is calculated to obtain the virtual common clock reference. Figure 8 presents its evolution in each Media Client in a 1-hour session. It can be observed that the skews measured in each device regarding the clock of the Time Server are different, but all follow the same trend (determined by that clock) and that differences between devices are maintained. There is a difference near 1,5 s between the local

[33] https://www.ffmpeg.org/ (last access: July 2020)
[34] https://www.bento4.com/ (last access: July 2020)

clocks of PC4 and the tablet during the session. If no clock reference and IDMS sync mechanisms are used, these differences could result in playout asynchrony values exceeding the allowed threshold to guarantee an acceptable shared experience.

In order to show the need of an IDMS solution on the platform, Figure 9 shows the evolution of the *IDMS asynchrony* between the playout processes of the Follower Media Clients and the Master User's Media Client (PC1) in a long session without any interaction between users (all of them watching the action film). Every time the asynchrony exceeds the ±100 ms threshold (dashed red lines) a playout adjustment is performed. Due to the fact that playout rates of PC2 and PC3 have been forced to be 0,01% higher and lower than the nominal rate, respectively, their asynchronies increase until they exceed the threshold, and then they are rapidly corrected. Although the tablet playout rate has been configured as the original nominal rate, its asynchrony values increase progressively as time goes by, until the threshold value is exceeded (in second 4800, approximately) and it is also quickly corrected. If no IDMS solution was implemented, the asynchrony values in these devices would have exceeded the ±400 ms value, which would start to be annoying for users, as indicated in Section 2. Figure 10 presents an example of the asynchrony values measured during a 21-minute session in which 20 user actions (Table 5) were forced. As can be observed from that figure (and also from the first row of Table 6), during the session, there are very few dynamic playout adjustments in PCs due to IDMS (e.g., around second 1000 for PC3; second 1025 for PC1; and second 1090 for PC4). This is due to the fact that every time a Pause or Skip action is performed by the Master User's Media Client all the playout processes of the other Media Clients are synchronised to the specific playout position that the Master User's Media Client pauses or skips to (coarse sync [5]). At some points (e.g., near seconds 210, 360 or 510) it can be noted that if a Pause interaction had not been performed, the threshold would have been exceeded. Nevertheless, at the beginning, and every time an action is performed by the Master User's Media Client (except for Pause actions) the playout process of the tablet is considerably lagged in comparison to the one of the Master User's Media Client (near 700 ms in second 300). This is due to the limited resources of the tablet and its WiFi access. Nonetheless, the asynchronies are quickly corrected.

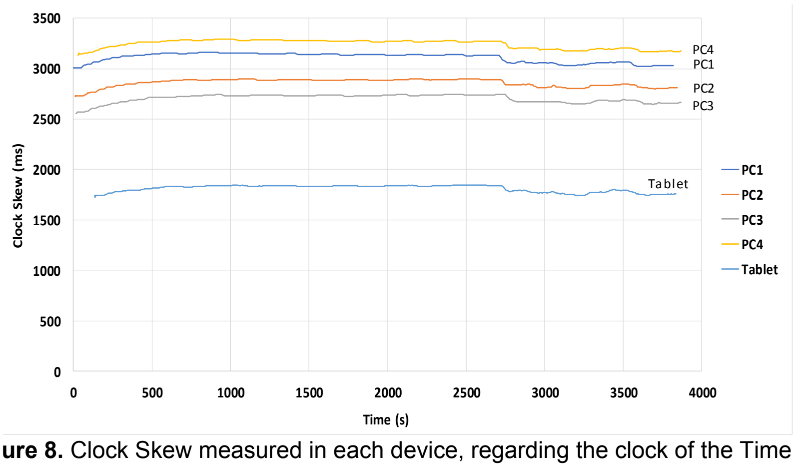


Figure 8. Clock Skew measured in each device, regarding the clock of the Time Server

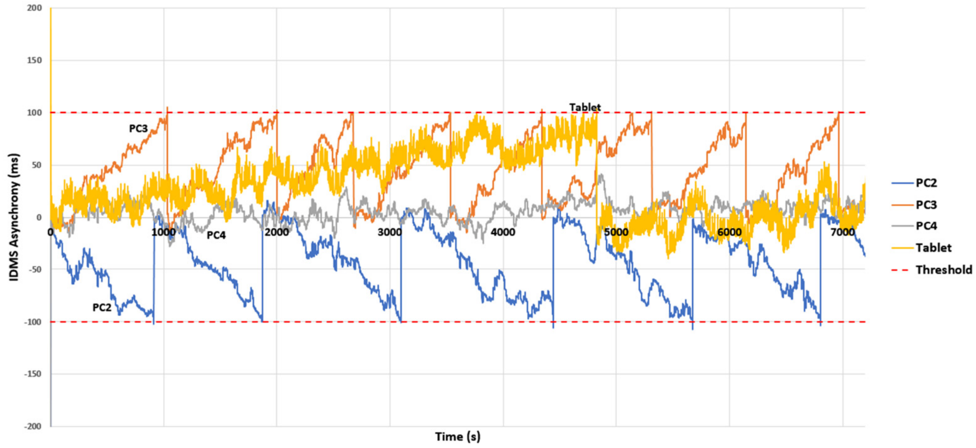


Figure 9. IDMS asynchrony evolution during a non-interactive session.

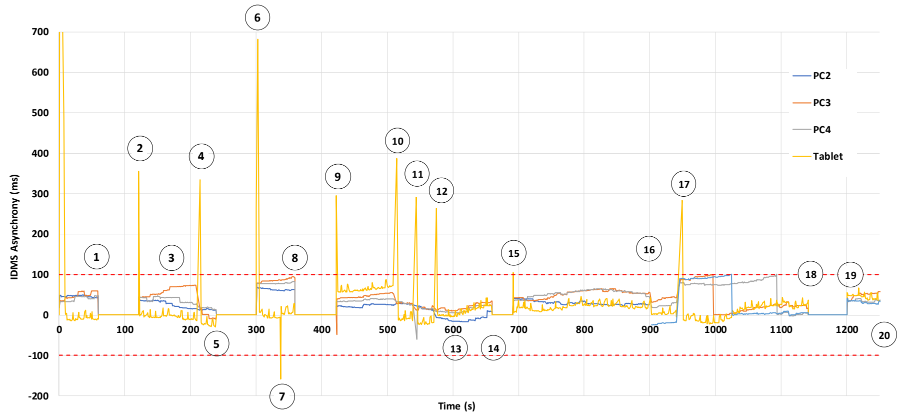


Figure 10. IDMS asynchrony evolution during an interactive session.

In order to be able to repeat the same actions and conduct several tests to collect more conclusive results, the actions in Table 5 have been configured in the Media Clients by using a script and the code has been modified to perform them in an automatic way without users. The measurements were conducted 10 times, and, for each device, the results are summarized in Table 6. In this paper, the distribution of asynchrony values is not considered. Although the configured *IDMS Asynchrony Threshold* value (±100ms) is exceeded several times during the sessions (especially in the tablet), those situations have been quickly corrected (Figure 10) and the measured asynchrony values during the session are much lower than it (last row of Table 6). PC1 and PC2 Media Clients have exchanged the master role during the session, so they have lower number of playout adjustments (when they hold the Master role, they do not measure asynchrony values). Since an *IDMS Asynchrony Threshold* value of ±100 ms is configured, the minimum value of playout adjustments is above this value in all the devices. The maximum value (in all measurements) of playout adjustments for PCs is 501,2 ms for PC4, while for the tablet it exceeds 1 s.

On the other hand, in order to check the performance of the Inter-Stream Sync solution between video and text chat streams, the delay experienced by text messages from the instant they are received until they are shown has also been measured. Most are shown immediately (delays lower than 15 ms) and only very few (2 and 3 in two of the tests) have been shown approximately 50 ms after their arrival, which is satisfactory and unnoticeable.

Action	Time (s)	Action of the Master Media Client	Consequence
1	60	Activation of shared board	Automatic playout pause action in all the Media Clients and all of them view the shared board drawing by the Master
2	120	Play	Shared board deactivation and video playout resumed
3	160	Activation of audio/videoconference tool	Users can use audio/video conference tool during the video playout
4	210	Skip forward to playout point in 43 m 39 s	Video playout skip in all the Media Clients to that point
5	240	Pause and Activation of shared board	Video playout is paused first, and then the shared board tool is activated and used by the master
6	300	Play	Shared board deactivation and video playout resumed
7	330	Chat	Users interact via text chat
8	360	Pause and audio/video chat	Video playout is paused first, and then users interact using the AV tool
9	420	Play	Video playout resumed
10	510	Skip back to playout point in 31 m 11 s	Video playout point is moved forward
11	540	Skip forward to playout point in 1 h 20 m 1 s	Video playout point is moved forward
12	570	Skip back to playout point in 48 m 15 s	Video playout point is moved backward
13	600	Stop audio/videoconference tool	The audio/videoconference tool is deactivated in all the Media Clients
14	660	Pause and activation of shared board	Video playout is paused first, and then the shared board tool is activated and used by the master
15	690	Play	Video playout resumed.
16	900	Master Role switched to PC2	PC2 user becomes the new Master User in the session
17	945	Skip forward to playout point in 2 h 1 m 3 s	Video playout point is moved forward
18	1140	Activation of shared board	Automatic playout pause action in all the Media Clients and all of them view the shared board drawing by the Master
19	1200	Play	Video playout resumed.
20	1260	Stop	End of shared media consumption session.

Table 5. Actions performed by the Master User's Media Client during the session

Parameter	PC1	PC2	PC3	PC4	Tablet
Max number of playout adjustments	2	2	4	3	11
Max. absolute value of adjustment/Asynchrony (ms)	110,7	107,3	474,5	501,2	1022,33
Min. absolute value of adjustment (ms)	100,2	100,4	100,1	100,9	104,7
RMSA ± 95% C.I. (ms)	35,34±12,1	29,18±5,41	44,07±27,18	43,88±5,73	28,54±14,77

Table 6. Obtained results

6.3 Subjective evaluation

The subjective assessment of the platform has been focused on determining key aspects, like the preferences and habits of users, the usability of the platform, its GUI and functionalities, and, finally, on sync perception. Future tests will be planned to assess the impact of other relevant aspects for social viewing, like the content genre, duration of the content, and number of users in a session.

6.3.1 Participants.

60 Spanish participants were recruited for the study, 32 males and 28 females, aged between 17 and 55 years old (mean value: 24,48; standard deviation: 8,08). Four of them were University professors (PhD), six were external employees within different university service companies and the rest were students (42 BSc and 8 MSc). 48,3% of the participants had a technical profile (related to telecommunications engineering), while 51,7% had none (tourism and journalism students and external employees). None of the participants had previous knowledge about Wersync.

6.3.2 Methodology

Several evaluation stages were designed, with a total duration of 45 minutes. At the beginning (first stage, duration: 10 minutes), all participants, after signing a consent form, read a brief description/tutorial of the platform, including its general purpose and a description of all its functionalities. Then, they filled in a questionnaire about general and demographic data, and about their preferences regarding the use of interaction tools on a platform like Wersync. In a second stage, in order to let the participants get familiar with all the provided functionalities, they were divided into pairs (user 1 and user 2) using two close PCs located in the same lab, and they were asked to follow a script with instructions summarized in Table 7, always under the supervision of an observer (duration: 15 minutes), who was the psychologist of the University Campus where the authors are currently working. Each user wore headphones with a microphone and could see both screens in order to see the other user's actions and GUI.

Step	Action
1	User 1 uploads a video located in the Desktop (series video in Table 4)
2	User 1 creates a session, choosing the previously uploaded video as the shared content
3	User 1 enables the Twitter tool, shares the session id through this app. If User 1 or User 2 have no Twitter account, the observer MUST be notified. Then, User 1 can share the session id with user 2 through any other channel (WhatsApp, e-mail…).
4	User 2 joins the session, enables the Twitter tool, if possible, and sends a tweet related to the session.
5	User 1 starts the playout of the content and sends a direct message (e.g., informing that the session has started), if possible.
6	User 1 and User 2 chat through the private chat tool.
7	In min 02:00, User 1 pauses the playout and uses the shared board, User 1 can draw a circle in that scene, pointing to a character. User 1 can also add some text there.
8	User 2 must confirm via the private chat that the drawing is visible, then User 1 removes the drawing.
9	User 1 resumes playout.
10	User 1 shares the file *sample1.jpg* located in the Desktop. User 2 shares the file *sample2.jpg* also located in the Desktop.
11	User 1 enables the video conference tool; User 2 must accept the audio/video connection.
12	User 1 mutes the playout (still playing) and starts a conversation via the video conference tool with User 2.
13	User 2 asks for the Master role, User 1 must grant it.
14	User 2 pauses the playout, unmutes it, skips the content to min 17:00 still paused.
15	User 2 uses the shared board and draws a rectangle. User 2 can also add some text there.
16	User 1 confirms that the drawing is visible through the private chat tool.
17	User 2 removes it and resumes the playout.
18	User 2 disables the video conference.
19	User 2 leaves the session.
20	User 1 must check that he/she has the Master role now.
21	User 1 leaves the session.

Table 7. Steps to familiarize with the platform (in couples)

After this training stage, new groups of 4 participants were formed and they could freely test the platform (also under the supervision of the observer) during 15 minutes, selecting different available stored contents (20-minute excerpts of the video content described in Table 4) and using whatever functionalities they wished. Note that the selection of groups of 4 participants is based on insights from previous studies [1] and on the conducted user-centric activities. The results from the employed questionnaires confirm that this group size is considered very appropriate by near 70% of the participants (Table 9). After that free-testing stage, each user answered several questionnaires (approximately 5 minutes): the System Usability Scale (SUS) questionnaire [38]; a set of qualitative questions regarding the tested functionalities, the aesthetics, usefulness and the overall use of the platform; a set of questions about their perception of the sync performance and experienced delay when using the platform; and a final questionnaire with some few optional open-ended questions for collecting free-form feedback.

The observer did not report any unexpected problem during the sessions and corroborated that the obtained results are coherent with what she observed during the assessment.

6.3.3 Results.

All the participants completed all the tasks and all of them understood the utility and applicability of the platform. None of them had previously used a similar platform. 90% of them were interested in sharing media consumption experiences with remote users. 78,3% declared having previously experienced shared media consumption with remote users (3,3% more than once a week; 13,3% once a week; 23,3% once a month; and 38,3% less than 5 times a year). Regarding the specific types of devices they used while sharing media consumption experiences, the responses were (multiple-choice question): laptops (92,6%), tablets (51,9%), smartphones (48,1%) and smart TV (40,7%). Likewise, the OS of these devices was: Windows (83,3%), iOS (46,3%), and Android/Linux (24,1%).

Participants were also asked about the media content genres they would like to consume in a shared way (multiple-choice question) and the more appropriate size of the group of users sharing the experience. The answers are summarized in Table 8 and Table 9, respectively.

Regarding the interaction channels they consider as appropriate, a 5-level Likert-type scale with the values *Totally Disagree, Partially Disagree, Neutral, Partially Agree* and *Totally Agree* was used. The obtained results are:

- Regarding a text-based channel, 85% agreed (partially, 26,7%, or completely, 58,3%), while only 10% partially disagreed and 5% were neutral. 71,7% preferred a private chat tool, faster and embedded in the platform, instead of using an external tool (e.g., Twitter) with higher delays and less integration. This did not matter to the rest of participants. In the case of using an external tool, 25 participants preferred Twitter and 3 preferred Facebook.
- Regarding an only audio-based channel, 78,36% agreed (partially, 35,06%, or completely, 43,3%), while only 10% disagreed (partially, 8,3%, or completely, 1,7%) and 11,7% were neutral.
- Regarding an audio- and video-based channel, 85% agreed (partially, 28,3%, or completely, 56,7%), while only 8,3% partially disagreed and 6,7% were neutral.
- 98,3% of the participants stated that the interaction channels provided by the platform were appropriate.

Genre	Percentage of participants (%) demanding the content
Series	81,5
Films	75,9
Educational	66,78
Reality shows	50
Sports	42,6
Music	7
News	25,9

Table 8. Preferred Media Content Genres for Shared Media Consumption[35]

Size (people)	Percentage of participants (%)
2	9,3
3-4	69,3
5-10	13,8
>10	7,6

Table 9. Preferred size of the group

All agreed that the platform was easy and comfortable to use. No participants encountered problems nor difficulties using it. 90% declared they would use the platform when it is available on-line. 3,7% declared they would use it more than once a week; 35,2% that they would use it on a weekly basis; 46,3% would use it on a monthly basis; and the rest (14,8%) would use it less than 5 times a year. Regarding the mean duration of the sessions, 3,7% declared it would be more than 2 hours; 40,7% declared it would be between 1 and 2 hours; 40,7% between 30 minutes and 1 hour; and the rest (14,8%) declared it would be between 10 and 30 minutes.

Also, a 5-level Likert-type scale was used to let the participants provide their opinion about the following statement '*the platform has a clear utility and applicability nowadays*'. 88,4% agreed (25% partially and 63,4% completely), while 8,3% were neutral and only 3,3% partially disagreed. 86,7% (26,7% partially and 60% completely) agreed that the platform was useful for providing a feeling of togetherness with other users, friends or relatives that are located in other places, despite the geographical distance; 11,7% were neutral and only 1,7% (1 user) partially disagreed. 85% (26,7% partially and 58,3% completely) agreed that the platform provides an added value to the media contents consumption experience; 11,3% were neutral and 1,7% completely disagreed.

[35] The sum of percentages exceeds the amount of 100% because it was a multiple-choice question (as in many other questions).

Regarding the statement '*this platform can have an impact on the current or future media consumption paradigm'*, 79,6% agreed (44,4% partially and 35,2% completely); and the rest were neutral.

The obtained results regarding the usability of the platform (SUS questionnaire [38]) are a mean value of 90 and a standard deviation value of 6,79, showing an '*excellent*' overall usability of the platform, according to the labelling in [39]. In order to determine whether the technical profile influences in the participant's perception of the usability of Wersync, the responses of users with technical and non-technical profile regarding usability have been compared. No statistically significant differences of means have been found, via independent sample t-test (p=0,22), which confirms that Wersync is easy to use, even for people with no technical profile.

Participants were asked about the relevance of an attractive GUI for this type of multimedia platforms. 3,3% were neutral, while the others stated that it is important (18,4%) or very important (78,3%). Next, they were asked about the attractiveness of the designed GUI. 83,3% agreed it is attractive (55% partially and 28,3% completely), while the rest were neutral. No participants declared they dislike the current GUI. Their suggestions regarding modifications of the GUI were compiled and will be taken into account in future versions of the platform.

The results of the participants' opinion regarding the statement '*The functionality X is useful and interesting in this platform*', by using the same Likert-type scale, are summarised in Figure 11.

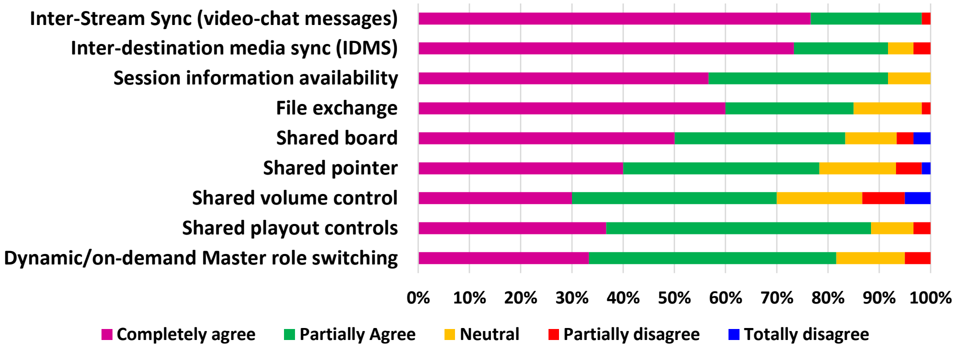


Figure 11. User's impression of Wersync functionalities

On the one hand, Figure 12 summarizes the participant's answers regarding a set of statements to evaluate their perception regarding the sync performance and delays when using Wersync. On the other hand, regarding their perception of the level of sync between users (IDMS) and between streams (Inter-Stream sync), a 5-point Likert scale was used with values ranging from 1 (lowest score) to 5 (highest score). Table 10 summarizes the participant's answers regarding IDMS and Inter-Stream Sync (when using the private chat). When using Twitter, all the participants declared the delays experienced were unacceptable to maintain an interactive conversation. No assessment of sync when using the shared board has been conducted because, when it is shared, the video playout is paused at the same point (i.e., video frame) for all the participants.

Participants were also asked whether they noticed any strange or annoying behaviour regarding the smooth playout of the shared video. Only 3 users perceived some impairments in the video playout during the session. Only two declared that some of them were annoying (the observer notified that both users had problems with the audio/videoconference tool at the beginning but that they were rapidly resolved).

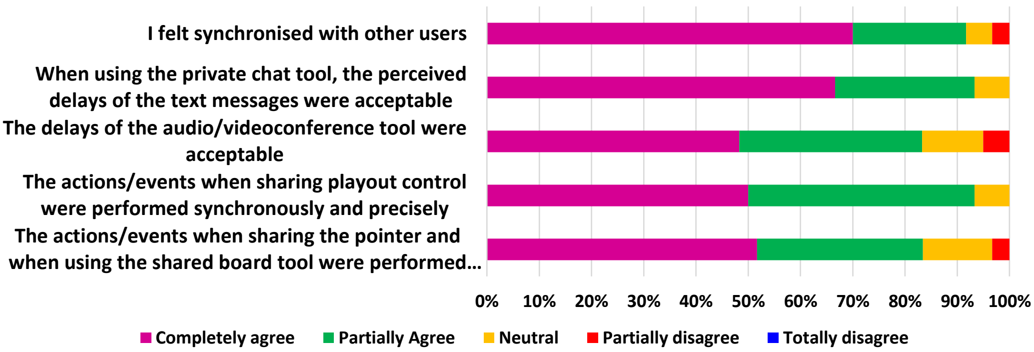


Figure 12. Perception of the sync performance and delays

	Frequencies (number of user that selected each option)					Mean value/ St. deviation
	1	2	3	4	5	
User's perception of the level of sync between users (IDMS)	0	1	3	29	27	4,36/ 0,66
User's perception of the level of sync between video and text streams (Inter-Stream sync)	0	1	0	20	39	4,61/ 0,58

Table 10. User's perception regarding IDMS and Inter-Stream Sync levels

The open-ended comments mostly mention the possible applications of this platform for tele-teaching or tele-working. Some comments were about the GUI. For example, one user did not like the icon used for the shared pointer; others suggested the inclusion of time information in chat messages and the possibility of sending GIF images; and some others recommended slightly increasing the size of some icons or adding an overlay text describing the purpose of some icons/controls when placing the mouse over them. Some users demanded new functionalities. Examples are the possibility of sending private messages to one specific member of the session; the possibility for the Master User to expel another user from the session; or the inclusion of an independent shared board to make annotations (i.e., not over the video player). The inclusion of these suggested functionalities has been left for further work.

6.4 Discussion

The main goal of this paper is to provide an innovative platform for synchronized social viewing, meeting key requirements and to determine whether the platform performs well and is welcome by end users in very relevant scenarios. The chosen evaluation conditions have been based on insights from previous own and related works (e.g. selected IDMS components, used methodology, configured scenarios…), on results from previous surveys [2] (use case, number of users per session…), and on insights and suggestions from the user-centric activities. Overall, the obtained results have been very satisfactory and promising. The features and components of the platform perform adequately, mainly in terms of timely responsiveness, effectiveness and accuracy. On the one hand, regarding the conducted objective assessment, it has been proved that sync solutions included in Wersync quickly correct detected asynchronies and maintain the asynchrony values much lower than the recommended thresholds (e,g, ±400 ms for Social TV use cases [25]). On the other hand, regarding the conducted subjective assessment, the obtained results have also been very satisfactory. Firstly, quite a high percentage of the participants are interested in sharing media consumption experiences with remote users, mainly in groups of 3-4 people and sessions with duration below 2 hours. This reflects the need and potential of platforms like Wersync. Most of them had previously experienced shared media consumption with remote users by using mainly laptops and tablets, but without any involved media sync solution and experiencing excessive delays. Despite expecting sports to be the most demanded content genre for shared consumption, series, films and educational genres were the preferred ones. The heterogeneity of types of consumption devices typically being used by consumers in shared media experiences, and of their platforms, support the decision of having chosen web-based technologies and components for the development of the platform. Secondly, Wersync is easy and comfortable to use, and its applicability and usefulness is clear to the participants. Likewise, most of them have given importance to the availability of an attractive GUI and have liked the design of Wersync. In that sense, it is important to note that the conducted evaluations have also served and will serve to enhance the platform in terms of GUI aspects, by adopting suggestions by participants. However, the results confirm participants are, in general, very satisfied with its current usability and attractiveness. Thirdly, the participants' perception regarding performance aspects, such as interactive and timely responsiveness, sync accuracy and behavior, have been very positive, in general. This supports the decision on the chosen technologies and components for developing the platform and is also a proof of an appropriate design and implementation of its modules and features. In addition, the majority of participants greatly appreciated the features provided by Wersync, finding them very useful and interesting, and also believe they contribute to an increased engagement, enriching the shared content consumption experience. The obtained results for all considered subjective measures (usability, ease of use, attractiveness, usefulness of the features, perceived performance) additionally help to validate the extracted requirements, both in terms of performance and users' reception.

With respect to applicability, participants believe that Wersync can be applicable in other relevant scenarios (e.g., related to leisure, education or tele-work), in which it can provide significant benefits. Despite the fact that the platform has been evaluated in a specific scenario, authors believe in the validity and relevance of the presented results for the research community. Notice that the configured delay and packet loss values correspond to what can be observed nowadays within long-distance fixed line connections or mobile networks (or even higher) and being representative of a broad range of application scenarios. The platform has been tested in a controlled lab environment as a first step to conducting further tests in (non-controlled) real scenarios in future work. New scenarios will also be evaluated to try to answer further research questions, such as performance in different networked scenarios, comparison between different alternatives for technological components, etc. In addition, a relevant sample of 60 participants took part in the subjective study, with balanced gender, age range and professional profiles. Although this work has not evaluated any additional platform, the evaluation of the features of Wersync shed some light on the acceptance of such features also provided by other existing platforms (see Section 2). The results have shown that the technical profile of participants does not influence their perception of the usability of the platform but how their age or gender does has been left for further study. As discussed, other relevant factors can have an impact on the users' QoE and interest in social viewing scenarios, such as variants of technological components, the underlying network scenario, the number of users per session, their profiles and relationships, the communication channel in use, content genre, duration of the session, the targeted use case (entertainment, e-learning…), etc. The results also shed some light on some of these factors, and their impact will also be investigated in depth in future studies.

Finally, apart from its applicability, the presented platform additionally constitutes a valuable and modular research testbed that will serve to further investigate the impact and benefits of technological variants and new components/solutions in the fields of social viewing and remote collaboration. The modularity of Wersync will also enable the accommodation of further requirements that may appear to provide extra features, due to their non-identification in this work, and/or to efficiently use the platform in other scenarios and use cases.

7. Conclusions and Future Work

In this paper, Wersync, a new web-based platform with the main aim of providing social interaction and/or remote collaboration between multiple remote users watching the same content in real-time has been presented. It provides many advantages and innovative features, improving and extending the ones provided by other analyzed platforms. The methodology to gather the requirements and associated features for social viewing, together with the decision of developing a new platform and the design choices in terms of the technological components for its development, have been described. Likewise, the architecture of Wersync, its modules, functionalities and implementation details have been presented.

Since sync is one of the most important key requirements in the kinds of applications Wersync is intended for, the included sync solutions have been described in detail and their performance has been objectively and subjectively evaluated, obtaining very accurate sync levels, far from the limits that start to be perceptible in those applications. An additional subjective evaluation of the platform has been conducted, obtaining very satisfactory results. The high interest of users in the platform, and its development based on standard web components, foresee its potential impact and widespread adoption in the media consumption landscape.

It is worth mentioning that an initial motivation for developing Wersync was also to constitute a flexible and accurate testbed to objectively and subjectively assess the impact of new technological components and strategies on both the sync performance and the perceived user's QoE when sharing media consumption experiences with remote users (e.g., regarding new interaction channels, engagement, etc.). As future work, several extensions to the platform are planned, some of them suggested by the participants in the subjective assessment. Moreover, the support of new content formats will be included, such as 360 video, YouTube videos[36], etc. Contents' Digital Rights Management (DRM) will also be considered in the near future. At this stage, the platform is not targeted for shared media consumption of live streams, but for Video-On-Demand services. It is planned to provide support for live services in future releases of the platform, but more research effort is needed. In addition, integration efforts with other platforms will be explored, like the one for subtitles in [40], and Big Blue Button. After finalizing the planned extensions and integrations, and once more objective/subjective evaluations are conducted, the implementation guidelines and instructions with more details will be documented and the platform will be made public, allowing both interested researchers and the general audience to benefit from the contributions of this work.

[36] Wersync can be easily extended to integrate online video platforms, such as YouTube (as in authors' web-based platform for adaptive subtitles described in [40]), leveraging its resources (e.g., massive storage and availability of videos) and functionalities (statistics, playlists…).

ACKNOWLEDGMENT

This work has been funded, partially, by the Generalitat Valenciana, Investigación Competitiva Proyectos, under Project AICO/2017/059, and also by The Telefónica Chair of "Technologies for social and environmental innovation" in the Polytechnic University of Valencia, under Project “WerSync: Web tool for social interaction and citizen collaboration”. Authors would also like to thank our student Jordi Belda for his contribution in the development of the preliminary version of the platform.

REFERENCES[37]

[1] Montagud, M., Boronat, F., Stokking, H., Brandenburg, R. v., 2012. Inter-destination multimedia synchronization: schemes, use cases and standardization. MMSJ, 18(6), 459-482. https://doi.org/10.1007/s00530-012-0278-9.

[2] Boronat, F., Montagud M., Marfil, D., Luzón, C., 2018. Hybrid Broadcast/Broadband TV. Services and Media Synchronization: Demands, Preferences and Expectations of Spanish Consumers, in IEEE Transactions on Broadcasting, vol. 64, no. 1, pp. 52-69. http://dx.doi.org/10.1109/TBC.2017.2737819.

[3] Vaishnavi, I., Cesar, P., Bulterman, D., Friedrich, O., Gunkel, S., Geerts, D., 2011. From IPTV to synchronous shared experiences challenges in design: Distributed media synchronization, Signal Processing: Image Communication, Vol. 26, Issue 7, pp. 370-377. https://doi.org/10.1016/j.image.2011.01.006.

[4] Montagud, M., Cesar, P., Boronat, F., Jansen J. (Eds), 2018. MediaSync: Handbook on Multimedia Synchronization, Springer, Cham., ISBN:9783319658391.

[5] Boronat, F., Lloret, J., García, M. 2009. Multimedia group and inter-stream synchronization techniques: A comparative study. Information Systems, 34(1), 108-131. https://doi.org/10.1016/j.is.2008.05.001.

[6] Mauve, M., Vogel, J., Hilt, V., Effelsberg, W., 2004. Local-Lag and Timewarp: Providing Consistency for Replicated Continuous Applications, IEEE Transactions on Multimedia, Vol.6, No.1. https://doi.org/10.1109/TMM.2003.819751.

[7] Geerts, D., Vaishnavi, I., Mekuria, R., Deventer, O. v., Cesar, P., 2011. Are we in sync?: synchronization requirements for watching online video together, ACM CHI’11, Vancouver (Canada). https://doi.org/10.1145/1978942.1978986.

[8] Wijnants, M., Dierckx, J., Quax, P., Lamotte, W., 2012. Synchronous MediaSharing: social and communal media consumption for geographically dispersed users, 3rd Multimedia Systems Conference (MMSys 2012), pp. 107-112, Chapel Hill, North Carolina (USA). https://doi.org/10.1145/2155555.2155574.

[9] Wijnants, M., Lamotte, W., De Meulenaere, J., Van, W. den Broeck, 2012. Qualitative assessment of contemporary media sharing practices and their relationship to the sMS platform, 1st International workshop on Socially-aware multimedia (SAM), Nara (Japan). https://doi.org/10.1145/2390876.2390887.

[10] Jansen, J., Cesar, P., Bulterman, D.C.A., 2013. Multimedia Document Synchronization in a Distributed Social Context. ACM Symposium on Document Engineering (DocEng 2013), Florence (Italy), pp. 273–276. https://doi.org/10.1145/2494266.2494314.

[11] Shahab, U.D., 2012. An Architecture and Implementation for Evaluating Synchronization Support for Shared User Experiences, MsC Thesis, Vrije Universiteit (VU) Amsterdam (Netherlands), CWI, Supervisors: D. Bulterman and J. Jacksen. https://pdfs.semanticscholar.org/f1a8/be22fcd6be1ce55870da79171cfaf1458468.pdf. .

[12] Schmitt, M., Gunkel, S., Bulterman, D.C.A., Cesar, P., 2013. A Quality of Experience Testbed for video-mediated group communication, IEEE International Symposium on Multimedia (ISM 2013), Anaheim, California (USA), pp. 514-515. https://doi.org/10.1109/ISM.2013.102.

[13] Rainer, B., Timmerer, C., 2014. Self-Organized Inter-Destination Multimedia Synchronization For Adaptive Media Streaming, In Proceedings of the 22nd ACM international conference on Multimedia (MM '14). ACM, New York, NY, USA, 327-336. https://doi.org/10.1145/2647868.2654938.

[14] Rainer, B., Petscharnig, S., Timmerer, C., 2018. Merge and Forward: A Self-Organized Inter-Destination Media Synchronization Scheme for Adaptive Media Streaming over HTTP, In: M. Montagud, P. Cesar, F. Boronat, J. Jansen J. (Eds), MediaSync: Handbook on Multimedia Synchronization, Springer, Cham. https://doi.org/10.1007/978-3-319-65840-7_21.

[15] Pauwels, D., van der Hooft, J., Petrangeli, S., Wauters, T., De Vleeschauwer, D., De Turck, F., 2017. A Web-based framework for fast synchronization of live video players, 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Lisbon, pp. 524-530. https://doi.org/10.23919/INM.2017.7987322.

[16] Arntzen, I., Bjorch, N., Daoust, F., 2018. Media Synchronization on the Web, In: M. Montagud, P. Cesar, F. Boronat, J. Jansen J. (Eds), MediaSync: Handbook on Multimedia Synchronization, Springer, Cham. https://doi.org/10.1007/978-3-319-65840-7_17.

[17] Liu, Y., Shafton, P., Shamma, D. A., Yang, J., 2007. Zync: the design of synchronized video sharing. In Proceedings of the 2007 conference on Designing for User eXperiences (DUX '07). ACM, New York, NY, USA, Article 12, 8 pages. https://doi.org/10.1145/1389908.1389924.

---

[37] All the links included in the references have been accessed in October 2020

[18] Shamma, D., Bastea-Forte, M., Joubert, N., Liu, Y., 2008. Enhancing online personal connections through the synchronized sharing of video, In CHI '08 Extended Abstracts on Human Factors in Computing Systems (CHI EA '08). ACM, New York, NY, USA, 2931-2936. https://doi.org/10.1145/1358628.1358786.
[19] Oehlberg, L., Duchenaut, N., Thornton, D., 2006. Social TV: Designing for distributed, sociable television viewing, Euro ITV 2006, Athens. https://doi.org/10.1080/10447310701821426.
[20] Geerts, D., 2006. Comparing voice chat and text chat in a communication tool for interactive television, NordiCHI '06, Oslo. https://doi.org/10.1145/1182475.1182537.
[21] Weisz, J. D., Kiesler, S., Zhang, H., Ren, Y., Kraut, R. E., Konstan, J. A., 2007. Watching Together: Integrating Text Chat with Video, ACM CHI '07, San Jose. https://doi.org/10.1145/1240624.1240756.
[22] Huang, E., Harboe, G., Tullio, J., Novak, A., Massey, N., Metcalf, C., Romano, G., 2009. Social Television comes home: a field study of communication choices and practices in TV-based text and voice chat, ACM CHI '09, Boston (USA). https://doi.org/10.1145/1518701.1518792.
[23] Montagud, M., Cesar, P., Boronat, F., Marfil, D., 2015. Social Media Usage Combined with TV/Video Watching: Opportunities and Associated Challenges, IEEE STCSN E-Letter Vol.3 No.2. https://ir.cwi.nl/pub/23726.
[24] Mekuria, R., Cesar, P., Bulterman, D., 2012. Digital TV: the effect of delay when watching football, EuroiTV '12, Berlin (Germany). https://doi.org/10.1145/2325616.2325632.
[25] Rainer, B., Petscharnig, S., Timmerer, C., Hellwagner, H., 2015. Is one second enough? evaluating QoE for inter-destination multimedia synchronization using human computation and crowdsourcing, IEEE QoMEX 2015, Greece. https://doi.org/10.1109/QoMEX.2015.7148107.
[26] Montagud, M., Boronat, F., Belda, J., Marfil, D., 2016. Use of Web Components to Develop Interactive, Customizable and Multi-Device Video Consumption Platforms, Applications and Usability of Interactive TV, Springer-Verlag (2016), Volume 605, pp 26-43. https://doi.org/10.1007/978-3-319-38907-3_4.
[27] Wijnants, M., Quax, P., Lamotte, W., 2013. Cost-effective Web-based Media Synchronization Schemes for Real-time Distributed Groupware, WEBIST 2013, Auchen (Germany). http://hdl.handle.net/1942/15452.
[28] Belda, J., Montagud, M., Boronat, F., Martínez, M., Pastor, J., 2015. Wersync: a Web-based Platform for Distributed Media Synchronization and Social Interaction, Adjunct proceedings ACM International Conference on Interactive Experiences for Television and Online Video (ACM TVX 2015), Brussels (Belgium). https://ir.cwi.nl/pub/23729.
[29] Belda, J., Montagud, M., Boronat, F., Pastor, J., 2015. Plataforma Web 2.0 para la Sincronización Distribuida de Contenidos Multimedia e Interacción Social, (in Spanish), VI Congreso de TV Digital Interactiva – IV Jornadas Iberoamericanas de Aplicaciones y Usabilidad de la TVDi, Mallorca (Spain). https://ir.cwi.nl/pub/23728.
[30] Melvin, H., Shannon, J., Stanton, K., 2018. Time, Frequency and Phase Synchronization for Multimedia – Basics, Issues, Developments and Opportunities, In: M. Montagud, P. Cesar, F. Boronat, J. Jansen J. (Eds), MediaSync: Handbook on Multimedia Synchronization, Springer, Cham. https://doi.org/10.1007/978-3-319-65840-7_4.
[31] Montagud, M., Boronat, F., Stokking, H., Cesar, P., 2014. Design, Development and Assessment of Control Schemes for IDMS in a Standardized RTCP-based Solution, COMNET, 70(9), 240-259. https://doi.org/10.1016/j.comnet.2014.06.004.
[32] Montagud, M., Boronat, F., Roig, B., Sapena, A., 2017. How to perform AMP? Cubic adjustments for improving the QoE, Computer Communications, Volume 103, Pages 61-73, ISSN 0140-3664 http://doi.org/10.1016/j.comcom.2017.01.017.
[33] Su, Y., Yang, Y., Lu, M., Chen, H. 2009. Smooth Control of Adaptive Media Playout for Video Streaming, IEEE TMM, 1(7), 1331-1339. https://doi.org/10.1109/TMM.2009.2030543
[34] Hossfeld, T., Seufert, M., Hirth, M., Zinner, T., Tran-Gia, P., Schatz, R., 2011. Quantification of YouTube QoE via Crowdsourcing, IEEE ISM 2011. https://doi.org/10.1109/ISM.2011.87.
[35] Park, S., Kim, J., 2008. An adaptive media playout for intra-media synchronization of networked-video applications, J. Vis. Commun. Image Represent., vol. 19, no. 2, pp. 106–120. https://doi.org/10.1016/j.jvcir.2007.09.002
[36] Ferrari, F., Meier, A., Thiele, L., 2010. Accurate clock models for simulating wireless sensor networks, SIMUTools 2010, Torremolinos, Spain, March 2010. https://doi.org/10.4108/ICST.SIMUTOOLS2010.8693
[37] Gómez, D., Boronat, F., Montagud, M., Luzón, C., 2016. End-to-end DASH platform including a network-based and client-based adaptive quality switching module, in Proc. Int. Conf. Multimedia Syst. (MMSys), Klagenfurt, Austria, Art. no. 38.
[38] Brooke, J. 1996. Sus-a quick and dirty usability scale. In book "Usability evaluation in industry", issue 194, pages 189-194, Taylor and Francis Ed. https://hell.meiert.org/core/pdf/sus.pdf.
[39] Bangor, A., Kortum, P., Miller, J., 2009. Determining what individual sus scores mean: Adding an adjective rating scale. J. Usability Studies 4(3), 114-123. https://dl.acm.org/citation.cfm?id=2835589.
[40] Montagud, M., Boronat, F., Pastor et al., 2020. Web-based platform for a customizable and synchronized presentation of subtitles in single- and multi-screen scenarios. Multimed Tools Appl. https://doi.org/10.1007/s11042-020-08955-x

APPENDIX I. Configuration Settings in Wersync platform

The following table presents all the parameters that can be configured in the platform, their description, and examples of their possible values.

Parameter	Description	Example
IDMS interval	Interval between IDMS messages sent by the Master Media Client	2000 ms
Master re-election	Master re-election policy	Oldest
Time Request interval	Interval between Requests sent by Media Clients to the Time Server	10000 ms
IDMS Asynchrony threshold	Allowable threshold for the asynchrony value between Master and Follower Media Clients' video playout processes	160 ms
Adjustment Type	Type of playout adjustment to be performed by Follower Media Clients	'S&P' or 'AMP'
PercentAMP	Percentage the playout rate is slowed down or fastened up in AMP	0,1 (10%)
Maximum Time of Adjustment (MTA)	Maximum allowed time for performing an adjustment	5000 ms
Maximum Allowed Pause (MAP)	Maximum allowed time for performing a pause adjustment. If exceeded, a playout skip will take place	5000 ms
ICI_interval	Inter-Stream Sync Checking Interval	50 ms
Socket.IO port	Port for communications with the Communications Server	83

Table I-1. Configuration parameters

APENDIX II. Control messages

The following tables summarizes the messages related to synchronization processes (Table II-1), shared board (Table II-2) and videoconference (Table II-3) tools.

Message Purpose	Direction(*)	Type	Fields	Description
IDMS	M->F	Playout Timing	*RefPosition*	Video playout position (according to the video content timeline).
			RefTime	Wall-clock timestamp (according to the common virtual timeline).
Playout Control	M->F	Play	*Position*	Position (according to video content timeline) from which the video playout must start (or be resumed) or has been started (resumed).
			Time	Wall-clock timestamp (according to the common virtual timeline).
		Pause	*Position*	Position (according to the video content timeline) in which the video playout has been, or needs to be, paused.
			Time	Wall-clock timestamp (according to the common virtual timeline).
		Seek	*Position*	Position (according to the video content timeline) to which the video playout must seek to or has been sought.
			Time	Wall-clock timestamp (according to the common virtual timeline).
		Fast Forward	*Rate*	New playout rate
Virtual Clock Sync	C->TS	Time Request	*t1*	Client's System Clock timestamp at which this message is sent.
	TS->C	Time Reply	*t1*	Client's System Clock timestamp at which the corresponding Request message was sent by the Client.
			t2	TS Clock timestamp at which the corresponding Request message was received by the TS.
			t3	TS Clock timestamp at which this message is sent by the TS.

M: IDMS Master Client; **F**: IDMS Follower client; **TS**: Time Server; **C**: Media Client (Master or Follower)

Table II-1. Control messages to achieve sync

Type	Field	Value
Activate/Deactivate Board	ActivateBoard	True/false
Mousedown (click)	x	X coordinate
	y	Y coordinate
Mousemove	x	X coordinate
	y	Y coordinate
Mouseup	--	--
Activate Drawing	Drawing	'Line', 'Circle', 'Rectangle' or 'Text'
Erase	--	--
Text Change	Text	Current text in text box
Color Change	Color	New Color (format: 'rgb(RR,GG,BB)')
Thickness Change	Thickness	New Thickness value

Table II-2. Shared board related messages/events

Type	Field	Value
Activate/Deactivate audio and videoconference	Conference	True/false

Table II-3. Videoconference related message/event