Design and Simulation of a Distributed Control Scheme for Inter-Destination Media Synchronization

Mario Montagud, Fernando Boronat

Universitat Politècnica de València, IGIC Institute
C/ Paraninf 1, 46730, Grao de Gandia, SPAIN
mamontor@posgrado.upv.es, fboronat@dcom.upv.es

Hans Stokking

TNO: Netherlands Organisation for Applied Scientific
Research TNO, Brassersplein 2, Delft, the Netherlands
hans.stokking@tno.nl

Abstract—Inter-Destination Multimedia Synchronization (IDMS) is a key requirement to enable satisfying group shared media experiences. This paper presents the design and simulation of a Distributed Control Scheme (DCS) for IDMS. In this scheme, geographically dispersed receivers exchange (multicast) RTCP reports on their arrival and playout timing for RTP packets, and each one of them uses the collected reports from the other receivers belonging to the same logical group (or cluster) to adaptively adjust its local playout timing such that an overall synchronization status (within allowable limits) is maintained. Also, the strengths (interactivity, flexibility, scalability, and robustness) and weaknesses (traffic overhead, security) of a DCS for IDMS are discussed, in comparison with centralized approaches. Moreover, a simple technique to enhance the performance in terms of coherence when using the DCS for IDMS is proposed. Simulation results prove the effectiveness of the proposed control scheme and techniques to guarantee concurrently synchronized playout points for independent clusters of receivers.

Keywords- IDMS; DCS; RTP/RTCP; Simulation

I. INTRODUCTION

Traditionally, media consumption has been a passive and isolated activity. However, the advent of new media streaming technologies, interactive social applications, and synchronous communications, as well as the convergence between them, point to an evolution towards dynamic group shared media experiences. In this new paradigm, distributed consumers, independently of their location, the access network and the devices they are using, can be immersed in a virtual networked environment in which they can share services, interact and collaborate among themselves within the context of simultaneous media content consumption.

However, some challenges (e.g., synchronization, universal session handling, presence awareness, QoE, etc.) must be faced to enable the usability and acceptance of those advanced interactive services [1]. This paper is focused on one of the major challenges ahead, which is to ensure concurrently synchronized playout for each one of the involved users, such that they can perceive a consistent and coherent group shared media experience. This is known as Inter-Destination Multimedia Synchronization (IDMS) [2]-[3] and is becoming essential in several newer media sharing applications [3], such as: synchronous e-learning, networked multi-player games, or 3D tele-immersion (3DTI). For

instance, one of the most prominent use IDMS cases is Social TV ([1], [3], [5], etc.), in which significant events (e.g. a goal in a football match) should be perceived by all the users almost simultaneously in order to not break their interaction patterns.

The IDMS problem is not handled efficiently in current media content delivery technologies mainly due to the existence of a set of variable factors [3], some of which can be either related to the distribution network or to the user equipment's features, which can seriously disturb the original media timing at the receiver side, and result in different (and time-variant) delays when multicasting media content to a set of distributed users. Without intervention, such different delays lead to differences in playout timings. According to the measurements in [5], this delay variability can accumulate up to 8 seconds, and may further increase as the session goes on, thus becoming a serious barrier when interaction is needed between the user and the media content, or between different users in the context of specific content. Therefore, additional adaptive IDMS techniques must be provided to enable satisfying group shared media experiences in a synchronous and coherent way.

Up to date, several (centralized and distributed) approaches (defining proprietary protocols) to carry out IDMS have been devised [2]. In this paper, a preliminary version of a centralized RTP/RTCP-based IDMS solution ([6], [7]) has been enhanced to solve some compatibility constraints and extended to be able to also adopt a Distributed Control Scheme (DCS). As well, the suitability of a DCS for specific IDMS use cases is explored. Simulation tests prove the feasibility of the proposed DCS for IDMS and its satisfactory responsiveness in terms of interactivity, flexibility, robustness and consistency.

The structure of the paper is as follows: Section II presents some IDMS related works; Section III reveals the suitability of a DCS for specific IDMS use cases; Section IV describes our evolved IDMS solution and its novel functionalities; Section V gives some performance results; and finally, Section VI outlines our conclusions and future work.

II. RELATED WORK

Despite the increasing relevance of IDMS in emerging distributed applications [3], solutions to accomplish this kind of synchronization are scarce [2]. Regarding IDMS solutions, three structural schemes can be distinguished,

depending on the role played by each participant and on the communication processes between them to exchange relevant information for IDMS (Fig. 1): two centralized schemes, the Master/Slave (M/S) Scheme and the Synchronization Maestro Scheme (SMS); and a distributed one, the DCS. An exhaustive taxonomy of IDMS solutions adopting those control schemes can be found in our previous work in [2]. In the M/S Scheme (e.g. [8]) only a master receiver multicasts feedback reports about playout timing to the rest of (slave) receivers (Fig. 1a). Accordingly, slave receivers have to adapt their playout timing to match the one reported by the master. The SMS (e.g. [6], [7]) is based on the existence of a sync manager (that can be the media server, one receiver, or an independent entity), which gathers the playout timing information of all the active receivers and, if needed, sends back to them new adapted control messages to make them enforce IDMS adjustments (Fig. 1b). In the DCS (e.g. [1], and [9]-[15]), all the receivers multicast feedback information about their playout timing (Fig. 1c). Accordingly, each one of them can locally decide the IDMS reference from among its own playout timing and the ones collected from the other receivers.

In [9], an IDMS approach using a DCS is introduced for the first time, which adaptively keeps the temporal and causal relationships according to the network load. In [10], a bucket mechanism (in which users' events are delayed enough time to prevent inconsistencies before being executed) is used as a DCS-based solution to be applied in interactive Multiplayer Online Games (MOG). The work in [11] proposes a DCS-based algorithm for meeting the ordering, responsiveness and interactivity properties in MOG, while preserving a global consistent state (IDMS). Similarly, [12] presents a DCS-based mechanism designed to achieve both interactivity and consistency in MOG. In [13], a popular DCS-based IDMS solution is presented. It consists of two algorithms: i) *local lag*, to compensate for short term inconsistencies; and ii) *time warp*, to undo inconsistencies that may still occur due to various uncontrollable factors. This solution was adopted in [14], providing open, intelligent, and interoperable support services for social applications. More recently, in [1], such algorithms have been adapted to achieve coherent execution of user's actions at all the clients, so that a consistent version of a shared video watching experience is perceived by all of them. In [15], another DCS-based approach is presented, by taking into consideration different conversation roles in a MOG.

Figure 1. IDMS Control Schemes.

III. IDMS CONTROL SCHEMES COMPARISON

Each one of the IDMS control schemes has its own advantages and disadvantages. An exhaustive qualitative comparison between centralized (SMS and M/S Schemes) and distributed approaches (DCS) for IDMS was presented in [3]. The goal of such a comparison was to reveal their suitability and effectiveness for specific network environments and applications requirements. The above IDMS schemes were also compared and evaluated in a Multicast MANET in [16].

An overview of the comparison in [3] is shown in Table I, for each one of the analyzed factors. Each one of the IDMS schemes has its own strengths and weaknesses, so the choice between them is largely application-dependent. This table shows that the DCS outperforms the other schemes in terms of robustness, flexibility and fairness. It can also provide better performance than the SMS in terms of scalability and interactivity, and it is superior to the M/S Scheme regarding other factors such as consistency and coherence.

The above reasons made us extend our previous SMS-based IDMS solution ([6], [7]) to also adopt a distributed approach, since it can be a more suited option in those use cases in which high performance in terms of robustness, fairness, flexibility, scalability and interactivity is desired, despite of a slight cost in terms of traffic overhead, consistency or security. In those scenarios in which bandwidth availability is not a limitation, security aspects can be ensured, and multicast feedback mechanisms between the receivers are supported, the DCS becomes a good candidate for IDMS.

IV. DCS-BASED IDMS SOLUTION

A. Suitability of RTP/RTCP for Media Synchronization

Most of the existing IDMS solutions ([2]) define new proprietary protocols, with their own control messages, that may increase the network load and may lead to incompatibility between implementations.

Currently, many interactive multimedia applications make use of RTP/RTCP standard protocols, specified in RFC 3550. On the one hand, the timestamps, sequence numbers, and payload type identification provided by RTP packets are very useful for reconstructing the original media timing, reordering and detecting packet losses at the client side, thus allowing intra-stream synchronization.

TABLE I
IDMS CONTROL SCHEMES COMPARISON

Is DCS better than ... (Scheme) in terms of ... (Factor)?		Factors									
		Robustness	Scalability	Traffic Overhead	Interactivity	Consistency	Causality	Coherence	Flexibility	Fairness	Security
Schemes	SMS	Y	Y	N	Y	N	N	N	Y	Y	N
	M/S	Y	N	N	N	Y	N	Y	Y	Y	N

On the other hand, the reporting features provided by RTCP are useful to obtain quality feedback about data delivery as well as for acquiring inter-stream synchronization (e.g. lip-sync).

Further extensions to RTP/RTCP are accepted in RFC 3550 to include profile-specific information required by particular applications, and the guidelines are specified in RFC 5968. Likewise, RFC 3611 allows the definition of new RTCP eXtended Report (XR) blocks useful for exchanging additional QoS metrics required by targeted applications. As IDMS involves the distribution of feedback reports about receipt and playout timing, and this information can be considered as a QoS metric (it can reflect the effect of jitter, network load, packet losses, clock skews/drifts, CPU overload, etc.), RTCP becomes a promising candidate for carrying out IDMS.

Another advantage of using RTP/RTCP for IDMS is that the optimum transmission rate for the RTCP reports does not need to be designed, as required in most of the proprietary solutions in [2]. This is because the RTCP report interval mechanism is specified in RFC 3550, in which an upper bound of 5 % of the session bandwidth is recommended for the control traffic, and it is dynamically adjusted according to the number of active participants, thus allowing scalability to large groups.

B. Preliminary Version of the IDMS Solution: Limitations

A preliminary version of our RTP/RTCP-based IDMS solution ([6], [7]) extended the RTCP Receiver Reports (RRs) in order to include the playout point of each receiver (sequence number of the Media Unit - MU - being played out and its playout time). It also defined new RTCP Application-Defined (APP) packets to estimate network delays and to exchange playout settings instructions. This solution only made use of an SMS (the media server being the sync manager), and it was satisfactorily tested both in controlled real WAN scenarios [6], and simulated ones [7]. However, its deployment in large scale environments presented some compatibility limitations because the proposed RTCP extensions were adequate for proprietary implementations, but not for open standard solutions (they would break the operation and cause inconsistencies in most RTP end-systems and middle boxes).

Therefore, an evolved version of our IDMS proposal, which is currently under standardization process [17], is presented in this work. First, the guidelines in RFC 5968 have been followed to choose the most adequate RTCP extension point. Second, this IDMS solution has been extended to be able to also adopt a DCS since, as stated in Section III, this scheme is preferable to centralized ones in several IDMS scenarios. Likewise, several control techniques have been adapted and designed to enhance the responsiveness of the proposed DCS-based IDMS.

C. Adoption of New Control Messages

Two new control messages have been defined in the evolved version of our IDMS solution to solve the above inter-operability constraints [17]. First, a new RTCP XR block type, called IDMS report, is specified to enable each i -

th receiver to inform about its IDMS parameters: i) the original RTP timestamp of the MU being played (t'_i)¹ at that moment; ii) its reception instant (r_i); iii) its playout time (p_i); and iv) the group or cluster identifier to which the receiver belongs. Second, a new RTCP packet type, called RTCP IDMS settings, has been defined to enable a sync manager to send IDMS setting instructions to a specific group (cluster) of receivers. This packet includes: i) an RTP timestamp to which it refers; ii) the target reception/playout instants; and iii) the cluster identifier to which it is sent. This RTCP IDMS settings packet is useful for SMS-based solutions, as well as for Coarse Synchronization ([6], [7]), i.e. for providing the receivers guidance on when to begin the playout process. The format of both messages can be found in [3] and [17].

D. DCS Operation

During the media session, receivers regularly exchange RTCP RR to inform about QoS (RFC 3550). Additionally, IDMS reports must be included in each RTCP compound packet to enable an overall synchronization control.

Each receiver must only register the information of the incoming IDMS reports from all the other receivers belonging to the same cluster, despite the fact that it may receive IDMS reports from all the participants in the multicast session (Fig. 2). This way, once each i -th receiver has collected the overall playout information in its own cluster, it can compute the playout time discrepancy (i.e. asynchrony) in that cluster by comparing the local playout delays of all the receivers belonging to it. The maximum asynchrony (Δ_{max}) will be given by the time difference between the most lagged and the most advanced playout points of all the active receivers in a specific cluster (N_c):

$$d_{\max/\min} = \max/\min\{d_i = (p_i - t'_i), \forall i \in N_c\} \quad (1)$$

$$\Delta_{\max} = (d_{\max} - d_{\min}) \quad (2)$$

Figure 2. Control Messages Exchange for IDMS (Cluster-based DCS).

¹ The generation RTP Timestamp can be converted to its associated NTP-based wall-clock timestamp ($t'_i \rightarrow t_i$) by using the mapping time info provided by RTCP SRs.

E. Master Reference Selection Policies

If the detected asynchrony exceeds an allowed threshold ($\Delta_{max} > \tau_{max}$), reactive playout adjustments must be triggered to restore the synchronicity. In contrast to SMS-based approaches (e.g. [6], [7]), in this newly proposed DCS-based IDMS operation the necessary adjustments will be directly calculated by the receivers. Each one of them will select an IDMS target playout point (d_{IDMS}) among all the collected IDMS reports, by employing one of the following dynamic master selection policies [7]: i) sync to the slowest receiver ($d_{IDMS} = d_{master} = d_{max}$); ii) sync to the fastest receiver ($d_{IDMS} = d_{min}$); or iii) sync to the mean playout point ($d_{IDMS} = d_{mean}$). After that, each i -th receiver can compute the asynchrony between the selected d_{IDMS} and its local playout delay for the n -th MU being played out in that moment ($d_{n,i}$):

$$\Delta_{n,i} = (d_{IDMS} - d_{n,i}) \quad (3)$$

Consequently, each receiver must enforce playout adjustments to get synchronized (i.e. to minimize the detected asynchrony) with such target point. It can be done by following two possible techniques [7]. The first one is based on aggressive playout adjustments (see section IV.G), while the second one makes use of smooth adjustments (see section IV.H).

F. Fault Tolerance

During the DCS operation, if a specific IDMS report (just one) sent by “receiver i -th” is not received by “receiver j -th”, the IDMS control algorithm will not be drastically affected. This is because the RTCP reports are sent regularly (RFC 3550), and “receiver j -th” can wait for the reception of a next report from “receiver i -th”. If more than one successive IDMS reports are lost, receivers could have to wait for an excessive period of time in order to collect the playout timing from all the other receivers in a specific cluster. So, a control timer has been included to manage the triggering of the necessary playout adjustments in this evolved IDMS solution. This way, playout adjustments can be triggered either as a result of an asynchrony (over a threshold) detection or as a timeout event of this monitoring timer. Every time playout adjustments are performed, that timer is being reset.

The flow chart of the designed IDMS algorithm using the DCS (implemented in each distributed receiver) and SMS (implemented in the sync manager) is sketched in Fig. 3.

G. Aggressive Playout Adjustments

If $\Delta_{n,i} > 0$ (see equation (3)), the playout process of the i -th receiver is advanced, i.e. in front of the reference. So, using aggressive adjustments, that receiver must ‘pause’ (stop playing) its playout process during $\Delta_{n,i}$ seconds to synchronize, causing a probable *freezing effect* (Fig. 4). Otherwise, if $\Delta_{n,i} < 0$, the playout process of the i -th receiver is lagged. In that case, that receiver must ‘skip’ (jump or move forward) a certain number of MUs to minimize the detected asynchrony (Fig. 4).

H. Smooth Playout Adjustments

The above reactive playout adjustments could originate a noticeable degradation of the user perception (QoE) because, on the one hand, some important information may not be presented to the users (due to the *skipped* MUs) and, on the other hand, a sensation of loss of continuity may also be noticed (due to the *paused* MUs).

Thus, an Adaptive Media Playout (AMP) technique has been adopted to minimize the occurrence of the above long-term playout discontinuities. Previous works on AMP have been mostly focused to improve the intra-stream sync quality (e.g. [18]) and, occasionally, for inter-stream sync purposes (e.g. [19]), in both audio and video streaming applications. However, in [7], authors proposed to extend the use of AMP for IDMS purposes, since this technique can enable independent *clusters* of receivers to adjust their playout timing, within perceptually tolerable ranges, in order to smoothly acquire an overall synchronized status. In this paper, such AMP technique for IDMS, which was designed for an SMS-based solution, has been adapted to be applied in this new proposed DCS-based IDMS solution.

Figure 3. Flow Chart of the designed DCS and SMS for IDMS.

Figure 4. Playout Rate Imperfections & Adjustments to Acquire IDMS.

The operation of the AMP technique is as follows. Initially, the playout controller of each i -th receiver must play out the buffered MUs at a non-adaptive playout rate (in MU/s) given by $\mu_{n,i}=1/(t'_{n+1}-t'_n)$, as they were generated by the media server. Once each n -th MU finishes its presentation period, the next $(n+1)$ -th MU must be played out, and the buffer occupancy must be updated. Every time an asynchrony situation is detected, the target playout point for IDMS (d_{IDMS}) is computed and the AMP process is triggered. At this point, the AMP technique will attempt to either speed up or slow down the video playout in order to minimize $\Delta_{n,i}$ by means of smoothly adjusting its local playout delay to the d_{IDMS} (Fig. 4).

However, in order to perform AMP, the allowed ratio within which the video playout speed can be varied without degrading the media quality must be considered. Previous related subjective tests ([18]) have shown that video playout speed variations of up to 25% are often *unnoticeable* and, depending on the content and the frequency of the adjustments, variations up to 50% are sometimes *acceptable*. Thus, we assume in our simulation tests that video playout adjustments up to 25% lead to unnoticeable quality impairments, and we define a playout factor for each n -th MU in each i -th receiver ($\phi_{n,i}$) to specify this variation ratio. On the one hand, high values of the playout factor (near the upper limit, i.e. $|\phi_{max}| \approx 0.25$) will result in a more rapid synchronized status. On the other hand, low values of the playout factor would be more unnoticeable to users, but the overall synchronized status will be reached later. In this work, for simplicity, a linear adjustment policy has been adopted. The number of MUs involved in the AMP process (N^{AMP}) must be enough to allow an extremely deviated (advanced or lagged) receiver, with an asynchrony regarding the selected d_{IDMS} near the allowed threshold (i.e. $\Delta_{max} \approx \tau_{max}$), to adjust its playout timing without exceeding the maximum playout factor (i.e. $|\phi_{max}| \leq 0.25$). That number is given by the expressions in (4):

$$N^{AMP}_{advanced} \geq \left\lceil \frac{\tau_{max}}{\frac{1}{\mu_i(1-\phi_{max})} - \frac{1}{\mu_i}} \right\rceil; \quad N^{AMP}_{lagged} \geq \left\lceil \frac{\tau_{max}}{\frac{1}{\mu_i} - \frac{1}{\mu_i(1+\phi_{max})}} \right\rceil \quad (4)$$

This way, the designed AMP technique is able to achieve the IDMS goal while avoiding noticeable and annoying discontinuities in the receivers' playout processes.

I. Coherence Adjustment Technique

Let us assume that “receiver i -th” detects an asynchrony situation and starts its AMP process. During this period, the RTCP timer for that receiver expires and it multicasts an IDMS report including its (currently or recently adjusted) local playout point. It could be possible that “receiver j -th” is still waiting for that report from “receiver i -th” to complete all its group registries and compute the asynchrony in the cluster they belong to. In such a case, the asynchrony situation will not be detected by “receiver j -th” because it has been (partially) corrected by “receiver i -th”. This is not a serious constraint because, anyway, the overall asynchrony

in that cluster will be kept below the allowed threshold. But, if “receiver j -th” has not been selected as the IDMS reference, it will not be synchronized to the target playout point (d_{IDMS}) selected by the other receivers in that cluster. This means that all the receivers in that cluster may not be synchronized simultaneously, thus there will remain a residual asynchrony among them, lowering the overall accuracy of synchronization.

Consequently, a simple algorithm to solve this situation is proposed in this paper, thus providing better coherence/consistency in our DCS-based IDMS solution. It simply consists of using a bit of one of the fields reserved “for future use” in the IDMS report to indicate that an out of sync situation has been recently detected by the sender of this report (by setting it to ‘1’). This way, once that IDMS report is received by the other receivers in that cluster, they will be aware of such out of sync situation, and they will also adjust their playout process to acquire a more fine-grained synchronization, using the IDMS timing information of the last cycle (i.e. the one computed the last time all the IDMS reports from that cluster were gathered).

V. EVALUATION

Modeling and simulations were conducted using NS-2. We have tested our DCS-based IDMS solution in a multicast scenario with seven distributed receivers (Fig. 5), with variable delays to the media server, belonging to two different clusters (Table II). All the links were bidirectional, their propagation delay were set to 10 ms, and their capacity was configured as shown in the figure. The media server transmitted with a specific rate of $\theta=25$ MU/s. Additionally, heavy and fluctuating background traffic was configured over the network topology in order to cause jitter variability. We set playout rate deviations (Fig. 4) than larger customary in inexpensive oscillators, which can vary between 10-100 ppm [7], in order to force higher asynchronies between the receivers, and to test if they were successfully handled by our IDMS solution². Those values are shown in Table II. The duration of the simulations was set to 10 minutes and the value of τ_{max} was set to 80 ms in order to trigger playout corrections slightly before reaching an asynchrony of 100 ms that can be already perceivable and annoying in some IDMS use cases, according to the conclusions in [3].

Figure 5. Simulated Scenario.

² The effect of the delay variability and the playout rate imperfections over the local and global media synchronization (especially IDMS) can be found in [7].

TABLE II
RECEIVERS' PARAMETERS

Receiver	Cluster	RTT	Rate Skew	Rate Drift
R1 (LAN 1)	C1	~10 ms	$\gamma_1 = 0.05\%$	$\varepsilon_1 = 0.02\%$
R2 (LAN 2)	C1	~125 ms	$\gamma_2 = -0.02\% \rightarrow -0.03\%$	$\varepsilon_2 = 0.02\%$
R3 (LAN 3)	C1	~288 ms	$\gamma_3 = -0.05\% \rightarrow -0.02\%$	$\varepsilon_3 = 0.02\%$
R4 (LAN 4)	C1	~44 ms	$\gamma_4 = -0.015\%$	$\varepsilon_4 = 0.02\%$
R5 (LAN 5)	C2	~288 ms	$\gamma_5 = 0\%$	$\varepsilon_5 = 0.02\%$
R6 (LAN 5)	C2	~288 ms	$\gamma_6 = -0.02\%$	$\varepsilon_6 = 0.02\%$
R7 (LAN 3)	C2	~288 ms	$\gamma_7 = 0.015\%$	$\varepsilon_7 = 0.02\%$

A. Simulation Results

In all the simulation tests, smooth adjustments (AMP) were employed to acquire IDMS since, as it was shown in [7], this reactive technique clearly outperforms the aggressive adjustments (skips & pauses) policy, because using AMP the receivers are more fine-grained synchronized and long-term (annoying) playout disruptions are avoided.

1) *Cluster-based IDMS to the Slowest Receiver.* Fig. 6 illustrates the playout delay evolution of the receivers belonging to both clusters to acquire IDMS, using the proposed DCS, when the “sync to the slowest receiver” policy was employed. It can be observed that the asynchrony between the playout states of the receivers in each cluster progressively increased mainly due to the configured deviations in their local playout rates (Table II). Every time an asynchrony larger than τ_{max} was detected in each cluster, faster (slave) receivers smoothly adjusted (i.e. slowed down) their playout timing to match the one reported by the most lagged (master) receiver in the same cluster they belong (R3 in C1 and R6 in C2), as seen in the zoom view. However, we can observe that the R2 in C1 (in which the receivers were sparser than in C2) was not always simultaneously synchronized (e.g. at 200-th, 300-th and 480-th seconds, approximately) to the IDMS target playout point. This is because when the R1 detected an out of sync situation (after gathering the IDMS reports from all the other receivers in that cluster), R2 was still waiting for the IDMS report from R1. This way, when R1 sent its IDMS report, including its local playout point, R2 did not detect that out of sync situation because R1 had already begun (or even finished) its adjustment process.

Despite this policy guaranteeing fairness in those applications where receivers can compete among themselves (e.g. in battle MOG), this strategy may not be appropriate if the playout process of the master receiver is extremely lagged, because the playout buffers may progressively become flooded with MUs (overflow situation), with the probable degradation of the real-time perception. So, large playout buffers and/or the adoption of novel adaptive buffering control techniques would be required to avoid such a situation (left for further work). As an example, in a 90-minute on-line football match session, *Coarse Sync* techniques ([6], [7]) could be triggered at the half time or when the game is stopped by any action (corner, fault ...).

Also, we can see in that figure that all the receivers were perfectly synchronized at the beginning of the session, despite the different Round Trip Times (RTT) from the media server to each one of them (Table II).

Figure 6. Cluster-based IDMS to the Slowest Receiver using the DCS.

2) *IDMS to the Mean Playout Point.* Fig. 7a illustrates the evolution of the playout processes of all the receivers belonging to C1 when they were synchronized to the mean playout point, using the DCS. It can be observed that though the asynchrony in C1 was kept below the allowed threshold at any moment, the receivers were not always simultaneously synchronized to the IDMS target (mean) playout point. The same situation is shown in Fig. 7b, but when the coherence technique was enabled. In such a case, the receivers did adjust their playout timing every time τ_{max} was exceeded in that cluster, thus achieving higher performance in terms of coherence. Moreover, in such a case, a more uniform evolution of the playout delay in all the receivers can be observed. Fig. 7c shows that the playout rate was varied within tolerable limits (AMP) during the session lifetime, which may result in good results in terms of QoE, due to the unnoticeable adjustments, based on the conclusions in [18].

This master selection policy provides more fine-grained synchronization and low frequent playout adjustments, but it cannot guarantee buffer overflow or underflow situations because the existence of inaccurate (extremely advanced/lagged) receivers cannot be predicted and would have a quantitative impact on the calculation of IDMS target (mean) playout point. So, as in the other master selection policies, novel dynamic and adaptive sync techniques should also be developed in future studies.

The same process for the receivers in C2 was assessed, when using both the DCS and the SMS. As the evolution of the receivers' playout processes in both cases followed the same tendency and the graphs for both IDMS schemes were very similar (almost indistinguishable), in Fig. 8a only the graph for DCS is shown. A zoom view of the playout adjustments for the receivers belonging to C2 using both the DCS and the SMS is shown in Fig. 8b (left and right graphs, respectively). It can be observed that the asynchrony situation was corrected earlier using the DCS because each receiver started the playout adjustments once it collected the IDMS reports from all the other receivers in its own cluster (C2), and they were quite close to each other (see Fig. 5). Using the SMS, receivers did not start the adjustments until the reception of the RTCP IDMS settings packet (including the target playout point) from the sync manager (the media server in our case). This occurred later because, as it can be

observed in Fig. 5, there is a significant network delay between the receivers and the media server (see Table II). Moreover, the transmission interval of RTCP packets must follow the timing rules specified in RFC 3550, which specifies that the fraction of the total amount of control traffic added by RTCP must be limited to 5 % of the RTP session bandwidth. This percentage is further divided into two parts, where the 25 % must be reserved to the senders, and the remaining can be consumed by the receivers. Accordingly, the RTCP period interval is given by the equations in (3), where $n_{senders/receivers}$ is the number of senders/receivers in the session, $avg(RTCP_{size})$ is the average size of all received and sent (compound) RTCP packets (including IDMS reports, as well as transport and network layer headers, e.g. UDP + IP), and $BW_{session}$ is the allocated bandwidth for the RTP Session:

$$T_{sender}^{RTCP} = \frac{n_{senders} \cdot avg(RTCP_{size})}{0.25 \cdot 0.05 \cdot BW_{session}}, \quad T_{receiver}^{RTCP} = \frac{n_{receivers} \cdot avg(RTCP_{size})}{0.75 \cdot 0.05 \cdot BW_{session}} \quad (5)$$

The above report intervals are further randomized within the range $\langle 0.5; 1.5 \rangle$ to prevent bursty control traffic (RFC 3550). If the sync manager is implemented within the media server resources, such as in our simulated case, in which $n_{senders}=1$, $BW_{session}=200$ Kbps, and supposing a value of $avg(RTCP_{size}) \approx 1000$ bits, a delay of up to 0.6 seconds (maximum RTCP report interval period) could be accumulated between an asynchrony situation being detected and the transmission of the RTCP IDMS packet. Such an effect would be more significant if the sync manager functionality is implemented as a part of an RTP receiver in a large scale media session ($n_{receivers} \uparrow \uparrow$).

Even though the maxim playout asynchrony in each cluster may slightly increase during this additional delay, the issue here is that the out of sync situation will not be repaired during this time interval when using the SMS.

Finally, it can be appreciated in Fig. 8b that using the SMS all the receivers finished their adjustment processes almost simultaneously, because all of them were synchronized at the target playout point included in the RTCP IDMS settings packet sent by the Sync Manager. Using the DCS, however, each receiver started to adjust its playout process once it collected the overall playout status in its own cluster. Thus, the IDMS adjustment period did not begin/finish at the same time in all of them (lower performance in terms of coherence), as can be also appreciated in the zoom view of Fig. 7c.

3) *Playout Asynchrony Distribution.* Fig. 9 illustrates the distribution of the playout asynchrony in C1 when the “sync to the mean playout point” policy was employed, using the SMS and the DCS (with and without the coherence adjustment technique). It can be appreciated that the percentage of MUs played out with low asynchrony ranges was significantly higher in the SMS because all the receivers where synchronized almost simultaneously once they received the RTCP IDMS settings packets (coherence), as shown in Fig. 8b. That percentage was significantly enhanced in the DCS when the coherence adjustment

technique was enabled, as previously discussed. Conversely, the bar chart also reflects that the percentage of MUs played out with an asynchrony exceeding the allowed threshold (τ_{max}) was slightly superior in the SMS than in the DCS (interactivity).

5) *Traffic Overhead.* The RTCP report interval is dynamically computed according to the number of active participants in the session and their role (senders or receivers), so that the total amount of control traffic is fixed to the 5 % of the allocated $BW_{session}$. Therefore, the traffic overhead added by our IDMS solution will be always significantly lower than this percentage when using both control schemes (DCS and SMS). Using the DCS, the number of IDMS reports sent (multicast) by each receiver during the 10-minute session in each one of the simulations was around 2% of the total RTP data packets sent by the media server (~15000). Using this scheme, only the transmission of one RTCP IDMS packet is needed at the beginning of each stage in which the session is divided since each receiver locally calculates and performs the required adjustments according to the collected IDMS reports. Contrarily, using the SMS, the sync manager must multicast a new RTCP IDMS packet every time an asynchrony situation is detected. As an example, when the sync to the mean playout point was employed using the SMS, 5 packets were sent to C1 and only 3 packets to C2 (because the playout deviations in that cluster were minor).

The newly defined RTCP reports in this evolved IDMS solution are larger than the ones defined in the preliminary version. The IDMS report has a length of 288 bits vs 96 bits of extension of the RTCP RR EXT in the previous version of our IDMS solution [6], whilst the RTCP IDMS settings packet has a length of 224 bits vs 192 bits of each RTCP APP ACT in the previous version of our IDMS solution [6]. However, this only minimally affects the frequency of the RTCP report interval (see equations in (5)) since the total amount of RTCP traffic is bounded to 5 %. Anyway, the traffic overhead added by our IDMS solution still remains very low, because we have not defined a new proprietary protocol either, but we have taken advantage of the RTP/RTCP feedback and extension capabilities for designing our own adaptive IDMS solution.

VI. CONCLUSION

In this paper, a preliminary version of a centralized RTCP-based IDMS solution has been enhanced to solve some inter-operability constraints and extended to also adopt a Distributed Control Scheme (DCS). Likewise, the suitability of a DCS for IDMS has been explored, especially in those use cases in which flexibility, scalability, robustness and interactivity features must be provided. Simulation tests have proved the effectiveness of the proposed DCS-based IDMS solution for keeping an overall synchronized status (consistency/coherence) in a cluster-based shared media experience.

For future work, we will assess the impact of the application of the proposed IDMS control schemes and techniques over the QoE by implementing them in real video sharing applications.

Figure 7. IDMS to the Mean Playout Point using the DCS in C1.

Figure 8. IDMS to the Mean Playout Point using DCS and SMS in C2.

Figure 9. Playout Asynchrony Distribution: SMS vs DCS (C1).

ACKNOWLEDGMENT

This work has been financed, partially, by Universitat Politècnica de València (UPV) under its R&D Support Program in PAID-01-10 & PAID-00-12 Projects, and by FP7/2007-2013 under grant agreement no. ICT-2011-7-287848 (HBB-NEXT project).

REFERENCES

- [1] I. Vaishnavi, et al., "From IPTV to synchronous shared experiences challenges in design: Distributed media synchronization", *Signal Processing: Image Communication*, 26 (7), 370-377, August 2011.
- [2] F. Boronat, J. Lloret, and M. García, "Multimedia group and inter-stream synchronization techniques: A comparative study", *Inf. Syst.* 34, 1, 108-131, March 2009.
- [3] M. Montagud, F. Boronat, H. Stokking, R. van Brandenburg, "Inter-destination multimedia synchronization: schemes, use cases and standardization", *Multimedia Systems Journal*, Article in Press, 2012.
- [4] D.A. Shamma, M. Bastea-Forte, N. Joubert, Y. Liu, "Enhancing online personal connections through synchronized sharing of online video", *ACM CHI'08 Extended Abstracts*, Florence, April 2008.
- [5] R. Mekuria, "Inter-destination synchronization for TV-Broadcasts", Master Thesis, Delft University of Technology, 2011.
- [6] F. Boronat, J.C. Guerri, J. Lloret, "An RTP/RTCP based approach for multimedia group and inter-stream synchronization", *Multimedia Tools and Applications Journal*, Vol. 40 (2), pp. 285-319, June 2008.
- [7] M. Montagud, F. Boronat, "Enhanced adaptive RTCP-based Inter-Destination Multimedia Synchronization approach for distributed applications", *Computer Networks*, 56 (12), 2912-2933, August 2012.
- [8] Y. Ishibashi, A. Tsuji, and S. Tasaka, "A Group Synchronization Mechanism for Stored Media in Multicast Communications", *Proc. of the INFOCOM '97*, Washington, April 1997.
- [9] Y. Ishibashi, and S. Tasaka, "A distributed control scheme for group synchronization in multicast communications", *Proc. of International Symposium Communications*, Taiwan, 317-323, November 1999.
- [10] C. Diot, L. Gautier, "A Distributed Architecture for Multiplayer Interactive Applications on the Internet", *IEEE Network*, vol. 13, n° 4, pp. 6-15, August 1999.
- [11] C.E. Palazzi, et al., "On maintaining interactivity in event delivery synchronization for mirrored game architectures", *IEEE Global Telecom. Conference Workshops*, Dallas (USA), 157-165, 2004.
- [12] E. Cronin, B. Filstrup, S. Jamin, and A.R. Kurc, "An efficient synchronization mechanism for mirrored game architectures", *Multimedia Tools and Applications*, vol.23, n° 1, pp.7-30, May 2004.
- [13] M. Mauve, J. Vogel, V. Hilt, and W. Effelsberg, "Local-Lag and Timewarp: Providing Consistency for Replicated Continuous Applications", *IEEE Trans. on Multimedia*, Vol.6 (1), February 2004.
- [14] K. Hosoya, Y. Ishibashi, S. Sugawara, K.E. Psannis, "Group synchronization control considering difference of conversation roles", *ISCE '09*, 948-952, May 2009.
- [15] C. Hesselman, et al., "Sharing enriched multimedia experiences across heterogeneous network infrastructures", *IEEE Communications Magazine*, Vol.48 (6), 54-65, June 2010.
- [16] T. Nunome, S. Tasaka, "Inter-destination synchronization quality in a multicast mobile ad hoc network", *IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications*, Berlin (Germany), 1366-1370, September 2005.
- [17] R. van Brandenburg, H. Stokking, M.O. Van Deventer, F. Boronat, M. Montagud, K. Gross, "RTCP for inter-destination media synchronization", draft-ietf-avtcore-idms-06, IETF Audio/Video Transport Core Working Group, Internet Draft, July 2012.
- [18] Y. Su, Y. Yang, M. Lu, H. Chen, "Smooth Control of Adaptive Media Playout for Video Streaming", *IEEE Transactions on Multimedia*, Vol.1 (7), November 2009.
- [19] Y. Ishibashi, S. Tasaka, H. Ogawa, "Media Synchronization Quality of Reactive Control Schemes", *IEICE Trans. on Communications*, Vol.E86-B, 10, 3103-3113, October 2003.