An Architecture and Implementation for Evaluating Synchronization Support for Shared User Experiences

By

Shahab Ud Din

Dissertation for

Masters in Computer Sciences

Vrije Universiteit, Amsterdam

Supervised by:

Dick Bulterman, CWI, Amsterdam
Jack Jansen , CWI, Amsterdam
Date: 22nd October, 2012

Abstract

Synchronous shared experience has been studied in recent past in perspective of stream-based media. Inter-destination synchronization mechanism is fundamental requirement of synchronous shared experience. Inter-destination synchronization in stream-based media does not work well with the group of participants which varied available bandwidth and computing resources. Document- based media provides functionality of alternative content selection on bases of available resource. This makes document-based media a potential solution for inter-destination synchronization when participants of synchronous shared experience have varied bandwidth and computing resource.

This thesis investigated techniques for inter-destination synchronization for stream-based media and adopted them to work with document based media. Firstly we designed and scalable and efficient signaling architecture for exchange of necessary information between participants. We did a prototype implementation of designed architecture. Such architecture was required for implementation and evaluation of inter-destination synchronization techniques for document-based media. Secondly we studied existing inter-destination synchronization techniques for stream-based media in literature. We identified classes of techniques and adopted two of them to work with document-based media. We implemented and evaluated two classes of techniques in this thesis. In order to make synchronous shared experience interactive, action of the participants need to be synchronized across participants. These actions are user interactions which can change state of presentation. Finally we extended inter-destination synchronization techniques to synchronize user interaction actions across distributed participants.

Dedicated

To

Prophet Muhammad (SAW)

To follow Him as role model is the ultimate objective of my life.

Acknowledgements

All praise is due for Allah who bestowed us power and wisdom to nourish our lives in the best way possible. It's my pleasure to pay my gratitude to my day to day supervisor Jack Jansen for his thorough guidance and my supervisor Dick Bulterman for his upper hand and supervision throughout the project span. Without their cooperation and timely direction, it might have been a worst project of my life. I want to admit that I tried my best to be a good researcher under their umbrella of guidance, knowledge and experience. I acknowledge the technical support and discussions by SEN5 team specially Pablo and Rafael.

I am immensely thankful to my wife Shaza Hanif, without her cooperation it was almost impossible to successfully complete not only this project but my whole Masters. I have no doubt in admitting that I was able to perform well only because I was not only fully relaxed and comfortable but encouraged from home. Special thanks to him for bearing me and for his encouragement and patting at the moments when I was discouraged and dishearten.

I pay my gratitude to my father Muhammad Ibrahin (late), my mother Irshad Begum, my father in law Muhammad Hanif (who passed away during my thesis), and my mother in law Shamim Akhtar (choti ammi). They all suffered being without our company and always gave us warmth of their lap through unending prayers. Many a times, I specially requested my both Ami's for prayers and always felt strength afterword. I am also very thankful to my sisters and brother who continually knocked (via telephone) and inquired "what's the progress Shahab?" It was due to their encouragements and pushes that enabled me to proceed during ups and downs in different phases of the project.

It would be a complete injustice if I didn't thank my friends; Ahmad, Archana, Umair, Tanveer, Saleem, Masood, Imran, and Omair for bearing my tight schedule during studies. Special thanks to Haji Mukhtar who always showed concern to my studies. A special thanks to Sarah Weeks and Chris Weeks for proof reading.

Special thanks to Higher education Commission (HEC), Pakistan for financial support during my initial phase of master.

Shahab Ud Din

TABLE OF CONTENT

Table of Contents

Chapter 1:

Introduction

In the last two decades, the world has become a global village due to the ongoing developments in the field of communication technology. As compared to the past, people are moving to different geographical locations, to different cities or countries, for study, employment and other purposes. This leads to disintegration of families and friends. Different members of families are living in different geographical location. It is very common that a group of high school friends will live in different cities after graduation for higher studies and jobs.

In spite of geographical segregation of family or friends, we like to remain connected with families and friends. Our most enjoyable activities are group or collaborative activities, like watching the TV-play together with family members or friends. Despite being at distinct geographical locations, we like to participate in these group activities to feel a part of family or close to a dear friend. This leads to the emergence of the concept of the *shared experience*.

The objective of shared experience is to facilitate people at distributed locations to participate in the group activities, in a similar way as they used to participate in the past in their living room. There are many examples of these activities, like watching the TV-play together with family sitting on a couch, playing board games in a living room with friends and family and celebrating birthday parties.

Being at distributed locations, participants of shared experience may lie in different internet domains and can have different internet speed and available bandwidth. The computers at participating distributed location can have different computing resources as well. So these distributed participants can have varied bandwidth and processing resources, which will make the shared experience challenging. Our research work investigates the methods to deal with these challenges.

Setting the scene: Ali's scenario

Consider the scenario of figure 1. Ali is a student studying in Western Europe. His parents are resident in Pakistan and his sister is living in North America. Ali has recently graduated from his university and has had his graduation ceremony. Due to the space and time difference, none of his family members were able to attend his graduation ceremony. However, Ali recorded the ceremony on video and later uploaded it on the web so that all the family members can watch it. The family decided to watch the video at the same time, so that they can feel like they are watching it together in one room. Alongside watching the video, they also have the audio chat facility (like Skype conference call), so that Ali can comment on the different scenes of the video.

For such kinds of shared experience, to relate Ali's commentary with the presented video, it is required that similar scenes have to be displayed in all the three locations at a particular time. In order to achieve this objective, the media presentation across multiple destinations should be synchronized. In literature, it is named as a synchronous shared experience [42].

The geographical separation of all three locations of participants makes synchronized shared experience challenging. The computers at different locations may have varied resources: like processing power, memory and network bandwidth. Consider sister's computer have high-speed internet link and high resource computer. Ali's computer have low-speed link with low network bandwidth but parent's computer has a very slower internet connection and low computing power. Due to these varied bandwidth connections and processing resources, media play-out at locations will not be synchronized. This implies that, some mechanism is required to minimize the impact of varied bandwidth, to have synchronous shared experience. In literature this mechanism is termed as inter-destination synchronization or group synchronization [44].

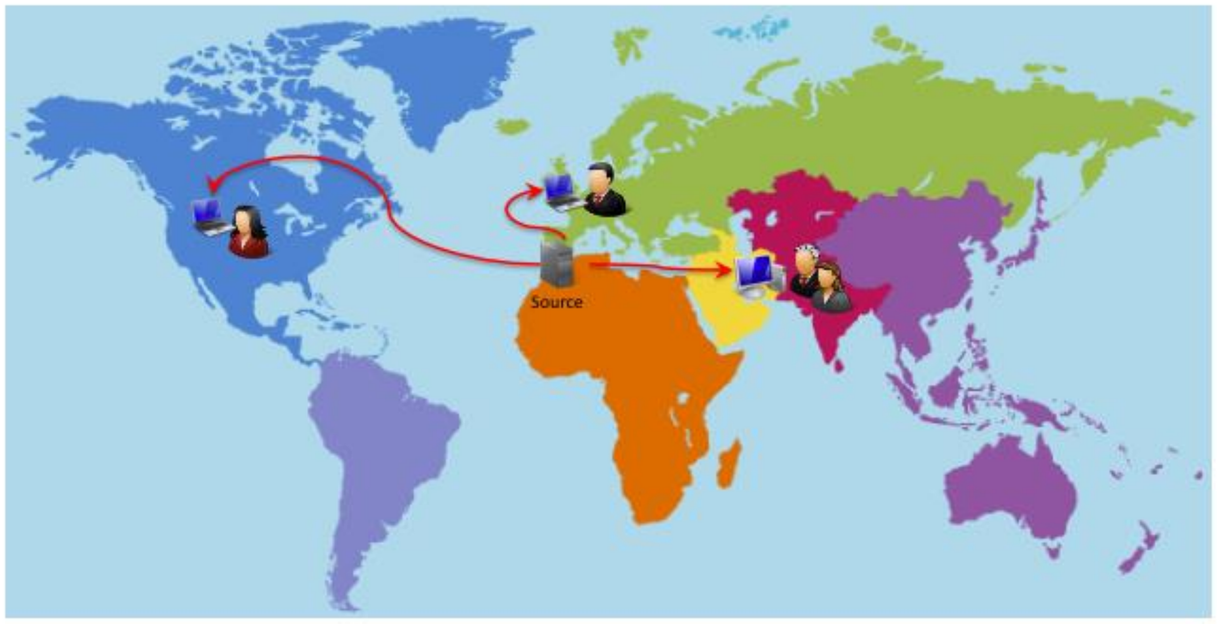


Figure 1.1: Ali's scenario.

Synchronous shared experience will not mimic the situation of a watching together if the user interactions with the video are not synchronized. While watching a movie together we can change the state of the play-out of movie by interaction like 'pause', 'play' and 'jump to a specific segment'. In share user experience these user interaction events should be executed across all participants in a synchronized manner. It will be required to make this type of experience as interactive synchronous shared experience. If one of the participants (Ali) of synchronous shared experience pause the movie we want it paused at all participating destinations (sister and parents).

In order to achieve synchronous share experience, we require inter-destination synchronization mechanism across participants. Existing inter-destination synchronization mechanism works well with stream-based media. In stream-based media all the participants will play-out same contents either from same or different source. In stream-based media, participants with varied bandwidth will not be able to synchronize their play-out when bandwidth differences are high. In situations where bandwidth differences are high we need mechanism where authors can specify the alternative contents for play out. We need mechanism to make it possible for participants to select among the alternative contents based on available bandwidth. These alternative contents will be different and suited to specific range of bandwidth, but are semantically same. Stream-based media do not have the ability to have alternative representations (high-quality video, lower-quality video, and slideshows) that have the same timing and same semantic meaning but different bandwidth/processing requirements. So in varied bandwidth and processing stream-based media will not synchronize the play-out at different locations to maintain the shared experience.

In document-based media authors can specify the alternative contents for play-out. It is possible for participants to select among the alternative contents based on available bandwidth. These alternative contents are different and suited to specific range of bandwidth, but are semantically same. This alternative content play-out characteristic of document-based media makes it a potential solution for inter-destination synchronization in varied bandwidth. This work therefore adopts the stream-based synchronization techniques to work with document-based media.

1.1. Synchronization in stream-based media

Consider that in Ali's scenario, geographically separated participants are linked via network and are using stream-based media. In stream-based media computers at all three locations (receivers) have to receive and play-out the same stream-based media file. The stream-based media file uploaded by Ali at source location, has temporal relations among the Media Units (MUs) of multiple streams. These streams can have video and associated audio streams. Computers at all three locations have to ensure these relationships during the play-out of streams. As the available resources at all three locations are different it will be difficult for destinations to ensure temporal relations across destinations. As a result the synchronization across destination is not possible.

In stream-based media all the locations will be receiving the same stream-based media file. If the variation of the bandwidth across the receivers at participating locations is not very high, stream-based media synchronization can be achieved. But in case of high variations of bandwidth among destinations it will be hard to achieve the stream-based synchronization. The reason is that all the three receivers will have to play-out the same stream-based media file irrespective of the resources they have. In order to achieve synchronization in this scenario with stream based media participants will lose too much data which will degrade the quality of the play-out. A slow participant with low bandwidth availability will have to drop many frames to catch up faster participants which have high bandwidth available. Similarly a faster participant will have to store frames for an extended duration to synchronize itself with slow participants. In this case the faster participant will have to drop too many frames to avoid buffer overflow.

1.2. Synchronization in document-based media

Structured document based multimedia is different from usual stream based multimedia. The structured document based multimedia specifies the time-based interactions between different multimedia objects. It does not encode the multimedia objects themselves, but references web addresses where the media can be found. In creating a multimedia presentation consisting of video, still images, audio, and captions, the document would specify when each of these various media elements is activated, as well as where they are rendered. Synchronization in structured document based multimedia can be defined in the document by specifying the duration of the objects and the temporal relations between the different objects in the presentations. In document based media, presentation of different media object is synchronized with the one document clock. In order to achieve synchronization across destinations we only have to synchronize the document clock across destinations. Once the document clock is synchronized across destination the alternative content selection functionality will make it possible to synchronize the presentation even destination have different bandwidth and computing resources.

In document based media it is possible to define the alternative media objects based on available band width. Based on the available bandwidth, the receiver will select among these alternative media objects, for play-out. These alternative media objects are different but they are semantically the same. For example an audio object is associated with three alternatives: a high resolution video object, a low resolution video object and a sequence of images. All three alternative objects have the semantically same contents. Depending upon the available bandwidth at the receiver, it can select one of the alternatives associated with the audio object. An example of a Synchronized Multimedia Integration Language (SMIL) is given in Figure 1.2 to elaborate the functionality of alternative content selection.

```
<switch>
      < video src="video-high-resolution" systemBitrate="1000000" />
      < video src="video-low-resolution" systemBitrate="100000" />
      <par>
              < audio src="audio-very-low" />
              < seq>
                  < img src="image1.jpg" end="7s"/>
                  < img src="image2.jpg" end="3s"/>
                 < img src="image3.jpg" end="9s"/>
                   < img src="image4.jpg" end="12s"/>
              < /seq>
       < /par>
 </ switch>
```

Figure 1.2 Example code for selection of alternative contents based on available bandwidth

In Ali's scenario, the sister has the higher bandwidth connection so the system will select the high resolution video for play-out and Ali's system with lower available bandwidth will select the low resolution video, while the parent's computer with very low available bandwidth will selects the sequence of images to play-out with the associated audio object.

As document-based media provide the functionality of alternative content selection on bases of available bandwidth recourses, it a potential solution for synchronization of play-out across destinations. In case of varied bandwidth participants can select the contents according to available bandwidth. These alternative contents will have the same timing information and are semantically same. A faster participant with higher available bandwidth can select high quality video and a slow participant with lower available bandwidth can select lower quality video.

1.3. Research Domain

The geographical separation of receivers makes synchronized shared experience hard to accomplish, specifically when the best effort IP network is used for communication. The current packet switching networks do not provide any guarantee on delay bounds of packet delivery. Rather, they only promise best effort delivery of data to the intended recipient. The delay variation is the inherent characteristic of best effort network. Due to delay variation at receivers, they will have high differences in available bandwidth.

As described in the previous section, situations where available bandwidth differences are high, stream-based media synchronization is hard to achieve. In these situations document-based media synchronization is a potential solution due to its functionality of selecting different but semantically same contents based on available bandwidth. Document-based media provide functionality to synchronize the media contents of different objects at same location, but they do not provide the functionality of the synchronizing the play-out of the contents across the destinations. In order to achieve synchronous shared experience in varied bandwidth difference across multiple locations, it is one of the objectives of this thesis to provide a platform to facilitate the inter-destination synchronization mechanism in document-based media.

Main Research Question: *How can we achieve inter-destination synchronization among the receivers with varied bandwidth using document-based media?*

Many techniques of inter-destination synchronization mechanism are presented in literature, but they are based on stream-based media. Studying the synchronous shared experience in document-based media is incomplete without achieving the inter-destination synchronization in document-based media.

1.4. Research Questions

In this section the main research question is split in to three sub questions. We identified three elements of the inter-destination synchronization in document-based media which impose a number of requirements on distributed multimedia applications based on document-based media across the Internet: Firstly, to achieve inter-destination synchronization in document-based media, we need to have a platform which facilitates the implementation and evaluation of existing stream-based media synchronization techniques for document-based media presentations. Secondly, inter-designation synchronization in document-based media demands that media presentation should be synchronized at all destinations. Thirdly, all the interactions which alter the state of the presentation by the user at a particular destination should be executed at all destinations. In the process of executing these user interactions across destinations, the state of the presentation should keep synchronized at all destinations. This leads us to the following research questions:

Research Question 1.1: *Can we create a platform which facilitates the implementation and evaluation of existing stream-based media synchronization techniques for document-based media presentations? Which techniques are relevant for synchronization of document-based media presentation?*

Document-based media provide functionality to synchronize the media contents of different objects at the same location. They do not provide the functionality of synchronizing the play-out of the contents across the destinations. This functionality has not been previously implemented in document-based media. It was necessary to design and implement a platform which can allow us to implement the inter-destination synchronization algorithm for document-based media.

Research Question 1.2: *How much synchronization can existing stream-based media synchronization techniques achieve in cases of document-based media presentation?*

As inter-destination synchronization algorithms for document based media have not been studied before, so it is required to study algorithms related to stream-based media present in literature. An adaptation of these algorithms for document-based media is required before implementation. Then the performance measurement and evaluation of these algorithms is required.

Research Question 1.3: *How much synchronization can we achieve in case of user interactions in document based media presentation? How much is the presentation at different destinations de-synchronized in case of user interaction and how long will it take to re-synchronize them?*

The user interaction like, ’play’, ‘pause’, ‘stop’, ‘resume’ and ‘navigation control’ should be executed across all participants in order to achieve the inter-destination synchronization. It should be evaluated that for the navigation control user action how much the media content at distinct location is de-synchronized and after how much of an interval the contents are re-synchronized again.

1.5. Our Research Approach

We intended to use Synchronized Multimedia Integration Language (SMIL) for writing structured document-based media content. The main reason for selection of SMIL was that it provides well tested method for alternative content selection mechanism based on available bandwidth. An introduction of SMIL is presented in chapter 3.

The objective of this thesis is to achieve inter-destination synchronization among the receivers with varied bandwidth using document-based media, so we did not develop our own play back mechanism. We only focused on inter-destination synchronization mechanism. We extended the functionality of ambulant [40] SMIL player to provide the inter-destination synchronization mechanism in SMIL. This provided us with the platform to implement the inter-destination techniques for document-based media. The ambulant Open SMIL Player is an open-source media player with support for SMIL 3.0, which is the latest version of SMIL.

We surveyed the existing inter-destination synchronization technique in literature. These techniques were for stream-based media. We adopted some of them for inter-destination synchronization in document-based media by using our platform which facilitates to implementation of them. The detail of platform and implemented technique is presented in chapter 4. We implemented and measure enough algorithms to validate our platform.

SMIL has well tested and validated functionality of the alternative content selection based on available bandwidth. For performance measurement and validation of the implemented techniques, we focused only on the inter-destination measurements of videos. The details of the experimental set-up and the evaluation mechanism are presented in chapter 5.

1.6. Implication of research work

The research work in this thesis is in context of synchronous share user experience but the synchronization platform and inter-destination synchronization techniques designed for document-based media can also be implacable in other domains. These techniques can include distributed e-learning environment for varied resource participants. The functionality of alternative contents selection in document-based media can be exploited to design a multi lingual e-learning platform and multi lingual shared experience platform. The research findings of these theses can act as basis of research in above said domains.

1.7. Organization

The rest of the thesis is organized in the following order. Chapter 2 describes the detailed background and state of the art techniques which are related to our research domain. Most of the findings of this chapter have been previously published and they have been included verbatim. Our research methodology is presented in detail in chapter 3. In chapter 4 we discuss the design and implementation of the platform to facilitate the inter-destination mechanism for document-based media. The detail of implemented techniques is also discussed in this chapter. Chapter 5 includes the experimental set-up, evaluation techniques and results. The thesis is concluded in Chapter 6 along with the future research directions.

Chapter 2:

Background and Related Work

For a better comprehension of components required to achieve interactive synchronous shared experience, it will be more appropriate to first illustrate the related concepts and their relevance with our work. Starting from, reasons which cause the challenges to achieve the objective of the interactive synchronous shared experience. Then we will discuss in detail types of distributed media synchronization. At the end we will discuss state of the art techniques and solutions in each type of synchronization techniques.

2.1. Causes of Asynchrony

MUs of the media stream suffer different type of delays from the generation at source to presentation at receiver. These delays can be different for different Media Units (MUs) depending upon the load at sender, network and receiver. These delay variations for different MUs cause asynchrony in the media presentation at the receiver. We can divide delays into three types: the delay caused by sender, network and by the receiver. Figure1 gives a pictorial representation of all three components of end-to-end delay.

Delay at sender: Capturing, coding, packetizing, protocol layer processing and transmission-buffering delays depend on the sender load and clock speed. At the different time instances, the sender may have different loads variations, which can cause the variation in these delays for different MUs. Moreover, if the related sub-streams are captured or/and sent by different sources, then, these delays experienced by different sub-stream can be more variable.

Network delay: Network delay is the delay experienced by the MUs in the network to reach its receiver, which varies according to network load. This delay can include the propagation delay and queuing delay at the intermediate routers. Network jitters is delay variations of inter-arrival of MUs at the receiver due to varying network load. This is due to the fact that the queues of the intermediate routers between sender and receiver may have different loads at the different time instances. This delay can cause intra-media asynchrony. Network skew is the time difference in arrival of temporally related MUs of different but related streams, i.e. differential delay among the streams, which can cause inter-media asynchrony. Clock drift is the rate of change of clock skew because of temperature differences or imperfections in crystal clocks. Clock skew is the clock time difference between the sender and the receiver. This is possible if the sender and the receiver are using local clock information instead of global clock information. The sender and receiver are considering time synchronized with respect to clock only if they are using the Network Time Protocol (NTP) or Global Positioning System devices.

Delay at receiver: The presentation, decoding, de-packetizing, protocol layer processing, and buffering delay at the receivers can be different for different MUs. These delay variations are present at the receiver due to the fact that different receivers may have different processing capabilities and different loads at the different time instance.

Depending on the nature of the application some or all of these problems may be relevant to different applications. Different synchronization mechanisms are needed to cope with these problems, to ensure the temporal ordering of streams and to maintain the presentation quality.

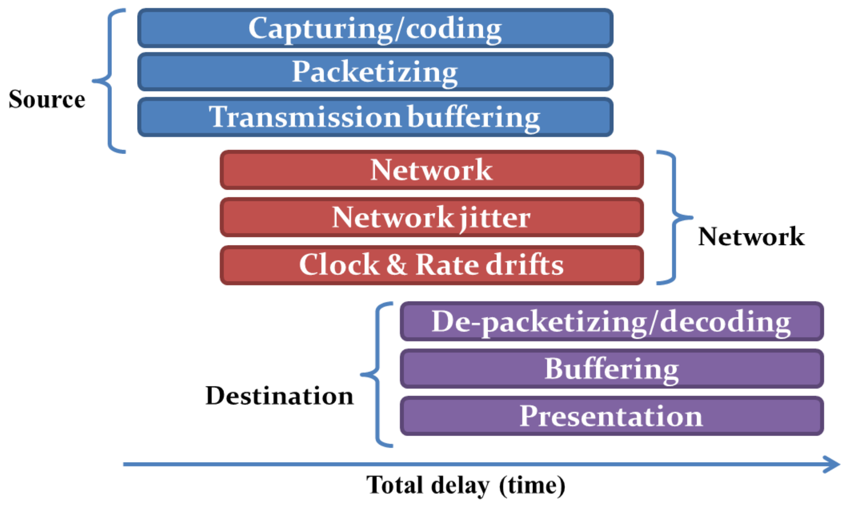


Figure 2.1 cause of asynchrony

2.2. Distributed Media Synchronization Types

Typical synchronization solutions can be classified in to two basic types: (1) Intra-media/intra-stream synchronization deals with the reconstruction of the temporal relations between the MUs of the same media stream, at the presentation time. For example, maintaining the frame sequence and frame rate of the video stream to ensure a smooth presentation. (2) During presentation, reconstruction of the temporal relations between the MUs of the different but related media streams is referred as Inter-media synchronization. A typical example of the inter-media synchronization is lip synchronization [1] between the corresponding audio and video stream.

Developments in computer and communication technology led to the popularity of distributed multimedia applications. In these applications, a geographically separated sender and receiver are linked via communication network. The sender is capturing the media stream with temporal relations and sending to receiver(s), which have to ensure these relationships during the presentation. The unreliability and unpredictability of best effort packet switching network make it hard for receiver to keep intact relations between the one or multiple streams. An accurate and explicit process of restructuring of the MU's at the receiver is required which is called *distributed multimedia synchronization*. In distributed multimedia environment, (3) apart from the two basic synchronization problems described earlier, another type of synchronization is required in case of multicast communication and is called inter-destination or group synchronization. This is required when geographically scattered group of receivers have to present the same stream(s) approximately at the same. (4) With the emergence of Interactive Distributed Multimedia Applications (IDMA) a new type of interactive synchronization emerges and examples are [2-6]. In these types of applications, users can modify the presentation state of stream and this modification has to be communicated to all receivers to maintain the synchronized view of the presentation among them.

2.2.1 Intra media synchronization

The reconstruction of temporal relations between media units of the same continuous media stream is referred to as intra-stream synchronization. For audio streams the basic media unit is voice sample. The spacing between the samples is determined by the sampling process and the objective of the inter-media synchronization is to ensure the same spacing at the play out time. For video streams the basic media unit is the video frame and the temporal relation is the frequency of the video frames. The frame production rate determine the spacing between the frames and at play out time similar frequency of frames have to be ensure by reconstructing the temporal relationship.

Many schemes have been proposed in the literature to ensure the temporal relationship at play out time. All the schemes use a receiver buffer for the temporary storage of incoming MUs. The audio/video samples/frames are then played out at appropriate time from the buffer. The objective of the whole process is to provide a presentation that resembles as much as possible to the temporal relation that was created during encoding process.

The ideal intra-stream synchronization quality is achieved by completely eliminating any kind of distortion in the temporal relationships of MUs and to completely restore the stream to its initial form. If the delay variability is unbounded, meaning that an infinitely long inter arrival period may appear, then no technique with a finite buffer can eliminate the distortion from the MUs. But for some assured services (QoS) which promise the bounded network delay, these techniques can achieve assured/ideal synchronization.

Intra-media synchronization schemes use buffer to handle the unpredictable delay in the network. But the use of MU buffer introduces end to end delay in the application which is directly proportional to the size of the buffer. All the Distributed Multimedia Applications (DMAs) have their own end to end delay tolerance requirement [33] depending upon the nature of the application. Interactive bidirectional applications like online quizzes have very strict end to end delay requirements and the applications like video conferencing have slightly less strict latency requirements. On other hand applications like video on demand (VOD) can allow much larger latencies. All the proposed schemes provide for some compromise between the intra-stream synchronization quality and the increase of end-to-end delay due to the buffering of MUs. For all the schemes there is a tradeoff between the synchronization quality and end to end delay, on one extreme there can be a buffer less scheme with minimum delay by presenting the frame as soon as they arrive and the assured synchronization that completely eliminate the effect of jitter on the cost of high end to end delay.

2.2.2 Inter media synchronization

The inter-stream/inter-stream synchronization is concerned with maintaining the temporal and/or logical dependencies among several streams in order to present the data in the same view as they were generated. Due to jitter in the network, at receiver MUs will not arrive in synchronized manner although they have been sent in a correct timely manner. The temporal relationship with in sub-streams is destroyed and the time gaps between arriving MUs vary according to the occurred jitter. Thus a synchronized presentation cannot be achieved at the receiver if arriving MUs of sub-streams would be played out immediately. Hence intra-stream and inter-stream synchronization is disturbed. To mitigate the effect of the jitter, MUs have to be delayed at the receiver such that a continuous synchronized presentation can be guaranteed. Consequently MUs have to be stored in buffer and the size of the buffer will correspond to the amount of jitter in the network.

For example in video conferencing applications speech and video Media Units (MUs) must have the temporal relationships at the time the streams were captured at source. These speech and video MDUs captured at the same time have to be played out together at receiver. Like any two different streams, the audio and video stream can be affected by the network delay differently. If these streams would be played without any synchronization mechanism at the receiver, the audio and the corresponding lip movement in the video will not be synched. This temporal relation between the audio and the video stream is called inter-stream synchronization or Lip synchronization. A pictorial representation of lip synchronization is presented in Figure 2.

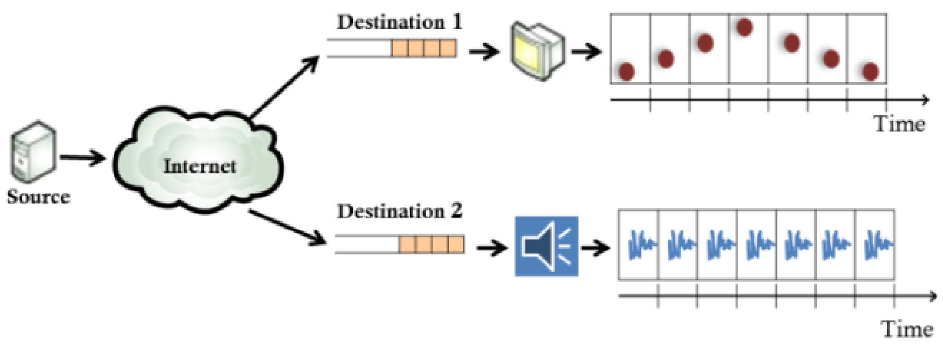


Figure 2.2 Inter media synchronization

The ideal inter-stream synchronization quality is achieved by completely eliminating any kind of distortion in the temporal relationships of MUs among multiple streams and to completely restore the stream to its initial form. This objective must be achieved on the fly as MUs arrive at the receiver, having crossed a network that alters the spacing between MUs, by imposing a variable network transfer delay.

There are many algorithms in literature which applied in different applications to achieve the inter media synchronization. Due to the different nature of the application it is challenging to compare the performance of the algorithms. These algorithms used many synchronization techniques both on sender and receiver side. There is no benchmark found in the literature to compare these techniques with. Most of the algorithms evaluate their performance with the satisfaction of the users of the target application. So instead of algorithm we decided to survey these techniques which are the building blocks of algorithms.

The study of inter media synchronization technique is summarized in comprehensive manner in [24, 28, 29].

2.2.3 Inter destination synchronization

In multicast media communication, apart from intra media and inter media synchronization we can find another type of synchronization called group or inter destination media synchronization (IDMS). It is the synchronization of the media presentation at different receiver and involves the play out process of different streams at different receiver. To add to the complexity of the problem these different receivers may be located at different geographical locations and may have different processing capabilities. These receivers may not only be of different type like smart phone and laptop computer but also may have the network connection of the different speed. Network quizzes can be a good example of this scenario, where the objective will be to achieve the fairness among all the participants of the quiz. Solution will be required to display all the questions of the quiz to the entire participant at the same time.

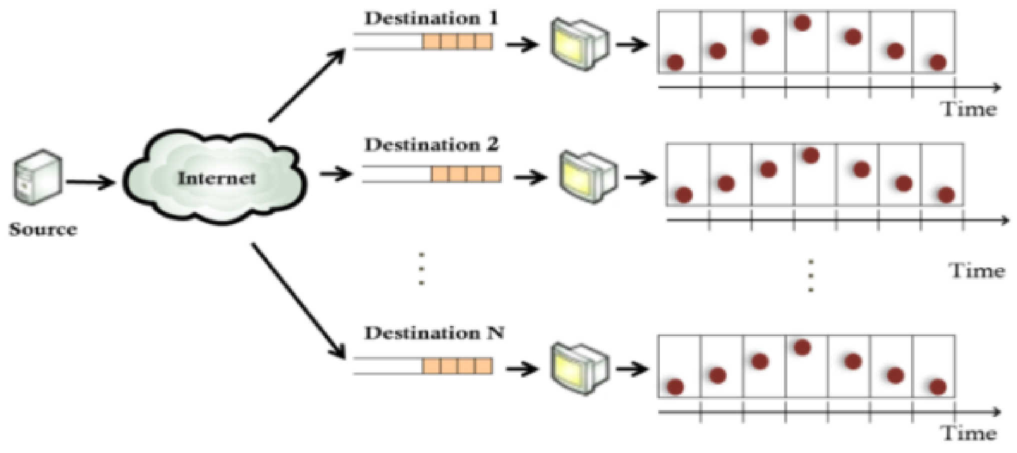


Figure.2.3 Inter destination synchronization

The other example can be of the real time distance learning (tele-teaching), where the teacher distributes (multicast) a multimedia lesson to number of students (receivers) who are located a different geographical areas. In this scenario teachers can also make comments about the live streaming of the lesson. Another similar example is of the interactive internet TV (Internet Social TV) where different groups of friends are watching a live online football match at different geographic locations. Consider the case when these groups can chat (audio/video) to each other to comment on the game to experience of watching the match together from distinct location. It will be very important to synchronize the streams so that they can watch the different events of the match at the same time to have the real experience of watching together. Figure 3 illustrate the scenario of inter destination synchronization pictorially.

The level of required synchrony among the receiver depends on the application on hand. Considering the above three cited examples: to ensure the fairness among the participants of the online quiz a hard synchronization will be required. In case of the other two examples required level of synchrony is a bit soft as compare to the online quiz case.

2.2.4 Event synchronization

The problem with implementing event synchronization is that in a distributed system events will always take some time to travel from one location to another, and therefore cannot be perfectly simultaneous. During this elapsed time various actions may occur at the other end, which may conflict with the original events. For distributed games this problem has been addressed by a technique called local lag and time warp.

2.2.5 Time synchronization Techniques

The de facto standard of time synchronization over the Internet is the Network Timing Protocol (NTP) [35]. Other protocols, such as, Precision Time protocol [34], also exist but are not widely used over the Internet. Even though a number of other time synchronization algorithms exist, for the application to run over the Internet NTP remains the predominant one over the Internet. But not all the system has the luxury of NTP synchronization mechanism.

Although global time synchronization between the destinations can achieve the most accurate results but the global time synchronization is not always available. Some other techniques are present in literature which does not provide the global time synchronization. They are not as accurate as global time synchronization techniques are but they synchronize the destination in estimated manner. In some literature they are called virtual clock synchronizations.

2.3. Existing state of the art techniques/solutions

Most synchronization mechanisms in the literature are either very abstract, independent of the application at hand or very application specific. It is challenging to summarize the techniques. Although in many cases the intra and inter media synchronizations are applied together and it seems difficult to analyze them partly, but we decided to summaries the state of the art literature by the type of synchronization techniques. In the next four sub sections of this chapter we will discuss the state of the art techniques for intra media synchronization, inter media synchronization, inter destination synchronization, and interactive distributed media synchronization respectively.

2.3.1 Intra media synchronization techniques

We divided the intra-stream synchronization in to two basic categories: Time oriented techniques and Buffer oriented techniques. In time oriented techniques the sender put a time stamp on the MUs and the sender and the receiver use clock in order to measure the delay and jitter. Receiver on the basis of these measurements devises a technique to ensure synchronous play-out of the streams. Buffer oriented techniques don’t use the clock rather they implicitly measure the network delay and jitter by the occupancy of the receiver buffer. The detailed description of the state of the art intra media synchronization techniques is presented in [43].

A. Time Oriented Techniques

We divide the Time oriented techniques in to three sub categories depending upon the timing information: techniques using global clock synchronization, techniques using approximated clock synchronization, and techniques using local clock synchronization.

The techniques in which sender and receiver use some mechanism for the synchronization of their clock are said to **use global clock information**. The existence of having the globally synchronized clock allows the receiver to measure the exact network delay of MUs. This network delay when added to the buffering delay at the receiver, it makes up end to end delay of MUs. If the receiver has these exact measurements, it can guarantee that the MU will be delivered and played out before or at the required time.

When global clock is not available, receivers estimate the one way network delay and variability of the delay. These techniques suite the applications which don't require a constant end to end delay. These techniques can be categorized as techniques **without global clock information**.

Apart from the two above mentioned time oriented techniques there is another category of techniques with **approximated clock synchronization**. These techniques do not require a global clock, so cannot guarantee constant end to end delay like the techniques based on global clock information. But they are better than the techniques without global clock information which only promise fluctuating end to end delay due to the variable network delay. In these techniques the receiver establishes a total delivery delay by measuring round trip time (RTT) between the sender and receiver. The receiver ensures that no MU will be presented after maximum delay value calculated by some expression of the RTT between sender and receiver. As a result of this assurance these techniques promise a soft delivery guarantee that is between the above mentioned techniques.

B. Buffer Oriented Techniques

The class of buffer-oriented techniques deals with the fundamental synchronization/latency tradeoff but don’t require the timestamps of MUs or the use of clocks. Buffer oriented techniques implicitly assess jitter by observing the occupancy of the receiver buffer instead of using timestamps. As these techniques don't rely on any timing information, they cannot provide the absolute/constant end to end delay guarantee. The total end to end delay comprises of the fluctuating network. The better stream synchronization quality can be gained by the increase in the buffering delay which will result in the increase in the end to end delay. So by using these techniques, delay performance that can approach the requirements of interactive applications is feasible, but cannot be guaranteed in absolute values. Due to this lack of guarantee regarding the end to end delay, buffer-oriented techniques are usually employed in video applications where the interactivity requirements are more relaxed than in audio applications.

The fundamental idea is to adjust the receiver's consumption rate of the frame according to the buffer occupancy. As a result of more frames in buffer the receiver increase it consumption rate to avoid buffer overflow which will make the presentation. While in case of less frames in the buffer the receiver will decrease its consumption rate to avoid buffer underflow which will cause the increase in the presentation time of a frame and ultimately decrease the smoothness of the stream. Buffer oriented techniques can be divided in to two broad categories: Pause/drop MUs techniques and Dynamic regulation of MU's duration.

2.3.2 Inter media synchronization techniques

There several ways of classifying the technique are possible, we chose to categorize them by their location, purpose, type of contents (live/stored) and synchronization information used. Before describing the categories of the technique it's important to note that any algorithm can use multiple of these techniques to achieve the synchronization mechanism even from different categories. More over these classifications are neither exhaustive nor orthogonal to each other as one specific technique can be categorize according to the location, purpose, content and information used. The detailed description of the state of the art techniques of inter media synchronization is presented as appendix B.

Location of synchronization technique: The synchronization control can be performed by the source or receiver. The synchronization control on receiver is used more as compare to the source. In case control is performed by the source most of the time it will require some feedback information from the receiver. The receiver will tell the source about the degree of asynchrony at the current instance.

Purpose of synchronization technique: we divided the techniques in to four sub categories with respect to its purpose: The basic Control schemes are required in almost all the algorithms. These must be present in all algorithms to provide synchronization. Examples are adding synchronization information in MUs at source and Buffering of MUs at receiver. The preventive control techniques are used to prevent the asynchrony in the streams. These are applied synchronized streams to keep them in the same state. The reactive control techniques are used to recover from the asynchrony, once it occurred. The common control techniques are techniques which can be applied as the preventive as well as reactive control techniques.

Type of media: Some of the techniques are used only for the store media and some for the live media, while some can be used for both types of media. Both types of media may have different implications for a particular technique. Some techniques can suit more to the stored media and others to live media.

Information used for synchronization technique: The information included in the MU for the synchronization purpose can be different like timestamp, sequence number. Some techniques used one among the sequence number and timestamp, while the other may use both.

2.3.3 Inter destination synchronization techniques

The IDMS techniques cited in the literature falls under one of the three categories: **master/ slave receiver scheme** (MSRC), **synchronization maestro scheme** (SMS) and **distributed control scheme** (DCS). The techniques presented in literature may have some variation but the basic concept of the technique lies in one of the above said categories. Here we present the basic control scheme of each category. For better understanding of these three schemes consider that M sources and N destinations are connected through a network. MUs of M different stream have been stored with timestamps in M source, and they are distributed to all the destinations by multicasting. The timestamp contained in an MU indicates its generation time. The streams fall into a master stream and slave streams. At each destination for inter stream synchronization, the slave streams are synchronized with the master stream by using timestamps.

A. Master/Slave Receiver Scheme (MSRS)

In MSRS the destinations are divided into a master destination and slave destinations. The master destination will be in control and will calculate the play out timing of the MUs independently according to its own state of the received stream data. The slave destinations should output MUs at the same timing as the master destination. In practice multiple streams will be received at each destinations and one of these stream will act as master stream for the purpose of inter stream synchronization at each destination. MSRS achieves group synchronization by adjusting the output timing of the MUs of master stream at the slave destinations to that at the master destination. It adjusts the timing by modifying the target output times at the slave destinations. This is because it determine the output time of an MU by referring to the target output time of the MU.

In order to synchronize the slave destinations with the master destination, the master destination sends control packets to the slave destinations. At start the master destination multicasts a control packet including the output time of its first MU (of master stream) to all slave destinations. This is called initial play out adjustment. For the continuous synchronization among receivers the master periodically multicast control-packets when the target output time of the master destination is modified. The master notifies all the slaves about the modification by multicasting a control packet which contains the amount of time which is modified and the sequence number of the MU for which the target output time have been changed. Figure 4 present the different type of message exchanges in the basic MSRS.

This technique was initially presented in [21] and then presented in [22] by extending the RTP/RTCP messages for containing the synchronization information. The benefit of this technique is its simplicity and less amount of information exchange as control packet to support IDMS. Only the master destination will multicast the control packets occasionally when its target play out time is modified or it will periodically multicast the control packets to accommodate the newly joined slave destination. Another factor which can influence the performance of the scheme is the selection of the master destination. If the slowest destination is selected as master it can cause buffer overflow on fast slave destination which will result as high packet drops at faster slave destination. Similarly if the faster destination is selected as the master destination it can cause the buffer underflow in the slower slave destination which can result as the poor presentation quality at slow destinations. In [32] all the possible options with effects are discussed for the master selection in this scheme. One issue with this technique is the associated degree of unfairness with the slave destinations. The other problem is that the master can act as bottleneck in the system.

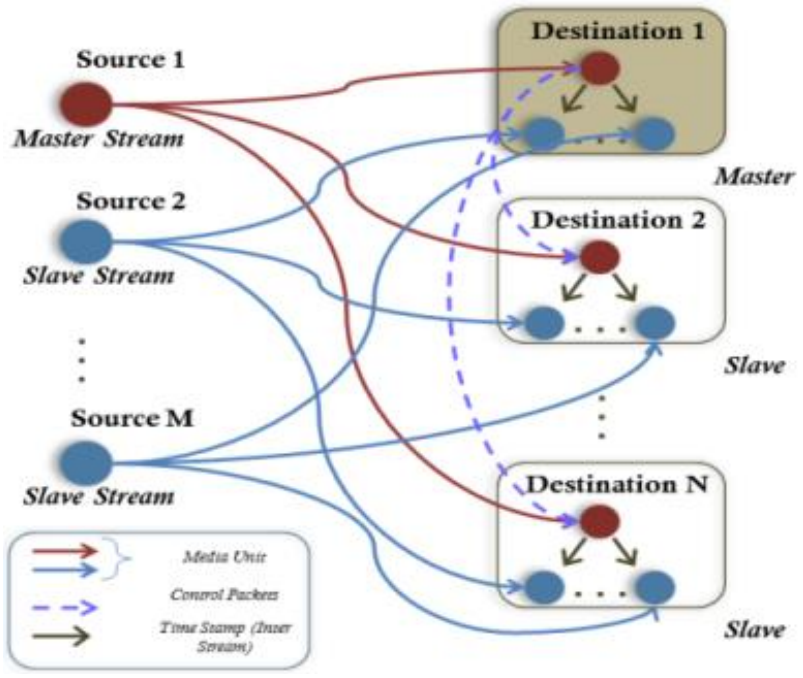


Figure 2.4 Master Slave Synchronization Scheme

B. Synchronization Manager Scheme (SMS)

SMS does not classify destinations into a master and slaves; therefore all the destinations can be handled fairly. It involves a synchronization manager in order to synchronize the master stream among all destinations. The role of synchronization manager can be performed by one of the source or receiver. Each destination estimates the network delay and uses the estimates to determine the local play out timing of the MU. Each destination then sends this estimated play out timing of MU to the synchronization manager. The manager gathers the estimates from the destinations, and it adjusts the output timing among the destinations by multicasting control packets to destinations. SMS assumes that clock speed at the sources and destinations is the same, and that the current local times are also the same (i.e., globally synchronized clocks). The basic scheme is illustrated pictorially in figure 5.

The SMS was initially presented in [23]. RTCP based schemes which follow the same basic principle was presented in [24]. The advantage of this scheme over MSRC is its fairness about the destinations, as the feedback information of all the destinations is accounted for in determining the play out time of the MU. But this fairness will cost more communication overhead among the destination and the synchronization manager. Like the MSRC this scheme is also a centralized solution so can face the bottleneck problem.

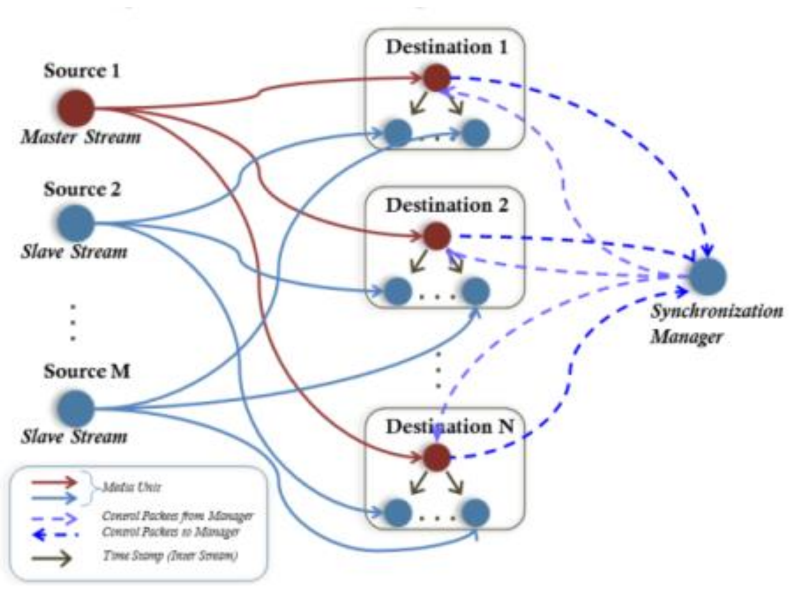


Figure 2.5 Synchronization manager scheme

C. Distributed Control Scheme (DCS)

Unlike MSRC and SMS technique, DCS neither classify the destination into master and slave nor has a specific synchronization manager. In this technique every destination estimates the network delay and then determines the play out timing of the MU. It then sends (multicast) this play out time to the entire destination. Every destination will then have the entire view of the estimated time of MU at all destinations. Each destination has the flexibility to decide the reference play out time among the timing of all the destinations. The pictorial representation of the scheme is illustrated by figure 6. This scheme was presented in [25-27].

This scheme gives higher flexibility to each destination to decide the play out time of MU. For example if by selecting the play out time of other destination can achieve higher IDMS quality but it may cause the inter stream or intra stream synchronization degradation. In this case the destination has the flexibility to choose between the types of synchronization depending upon the nature of application on hand. If the application on hand demands the higher inter stream or intra stream synchronization and can sacrifice on the IDMS synchronization to certain limit and destination can selects its own determined play out time and vice versa. DCS is distributed scheme by nature and will not suffer by bottleneck problem. If one or more destinations leave the system it will not heart the overall scheme. This greater flexibility and the distributed nature of DCS make it complex in terms of processing as before deciding play out time of MU the destination have to do more calculations and comparisons. Also it has higher message complexity as every destination will multicast the estimated play out time.

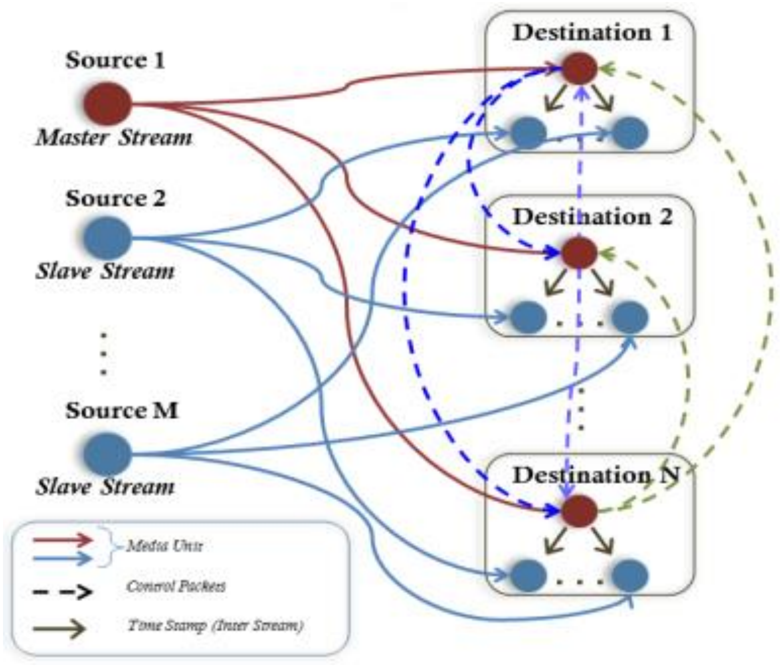


Figure 2.6 Distributed control scheme

2.3.4 Event synchronization Techniques

Event synchronization can be considered as a subclass of group synchronization. There are a number of relevant scenarios such as distributed gaming, instant messaging solutions and VCR-like distributed control execution while watching streaming video. The basic problem arises from the differences in event delivery time (jitter) seen by participating nodes. This leads to inconsistent copies across the nodes playing the same media. In this area [36, 37] proposed an algorithm called local lag with time-warp to handle this inconsistency in distributed games. The general approach is to estimate the worst delay in the network for a participating node and to enforce this delay on all participating nodes, including itself. If an inconsistency is found at a participating node then the events are rolled back to the last known consistent state and played fast forward to reach the current time. A similar concept based on the local lag mechanism is used in synchronizing distributed games, called bucket synchronization.

C. Diot et al.in [3] presented that with a multicast communication architecture and with a simple synchronization mechanism, a fully distributed interactive application can provide an acceptable level of consistency to distributed interactive applications on the Internet. C.M. Huang et al. in [4] modeled interactive synchronization control has been one of the most complicated and crucial system interactive, multi stream, multimedia presentation systems.

2.4. Alternative Contents Selection in document based media

The major advantage of the document based media over the stream media is that in document based media all the media components are referenced in the document. In the structured document it is also possible to define the alternative contents depending on the available properties of the destination device.

Among the alternative contents, suitable option can be selected based on the device properties. These properties can be the size of the screen of the device, available bandwidth and the host language of the destination computer.

Cesar et al. [47] performed the alternative content selection property of structured document based media using SMIL language in context of adaptation in ubiquitous environments. They discussed experiences of building the system and on the benefits of using structured documents in the form of automatic transformations of multimedia services.

2.5. Summary

We added all the synchronization techniques for sake of completeness of document. But not all the sections of this chapter are directly related to our work. We did not implement any of the intra-media synchronization technique in this work. We did implement the inter-media synchronization technique for synchronization of video and audio but it was not the main focus of this thesis. The work in thesis is focused about the inter-destination synchronization and event synchronization in order to achieve the goal of interactive synchronous shared experience. Sections 2.2.3, 2.2.4, 2.3.3 and 2.3.4 are most relevant with our work.

The preliminary steps towards synchronous shared experiences can already be seen in services, such as Yahoo Zync [39]. In Zync two users situated at different locations can watch a YouTube video together while communicating via text. The play out of the videos at both ends is synchronized. In this way Zync integrates instant messaging with video playback, enabling users to share their experiences about the video. Our work is different from yahoo Zync in two aspects: first in yahoo Zync synchronize the videos between two destinations and we are doing it among multiple destinations, secondly it does not handled the user events and we are investigating the results of user event exchange to make it interactive shared experience and thirdly yahoo Zync synchronizes the stream based videos like in YouTube and we are synchronizing the structured document based presentations.

Chapter 3:

Research methodology, Tools and Techniques

This chapter will define the adapted methodology of this research work. We will then discuss briefly ambulant player [40] and associated SMIL [41] language.

3.1. Research Methodology

The research objective of thesis described in chapter 1 is:

> **Main Research Question:** *How can we achieve inter-destination synchronization among the receivers with varied bandwidth using document-based media?*

The research methodology adopted is described in this section to answer the main research question. We divided our research methodology in multiple steps taken in order to achieve the objective of the thesis. In the following section we elaborate these steps one by one.

Selection of document-based media: The research goals of the thesis demand that we require a language for structured document-based synchronization and a media play back engine which support that language. The first step was to select the language and we decide to use SMIL for that purpose. The reason for selection of SMIL is that it provides well-tested functionality of "selection of alternative contents". In SMIL author can select between alternative contents on basis of available bandwidth, system screen size and host computer language. Alternative contents in all cases will be different but semantically are same.

Selection of playback engine for SMIL: The next step was to select the playback engine for structured document based media files. We selected ambulant for the said purpose. The ambulant Open SMIL Player is an open-source, full featured SMIL player. It is intended to be used within the researcher community in projects that need source code access to a production-quality SMIL player environment. It may also be used as a stand-alone SMIL player for applications that do not need proprietary media formats. Ambulant have a plug-in support for its extension. We implemented plug-in for ambulant player to support inter-destination synchronization mechanism for document-based media.

Extension of ambulant SMIL player: SMIL has built-in capability of inter-media synchronization mechanism between different media objects referred in a document. Ambulant is a SMIL based playback engine for document based media. It can only provide inter-media synchronization mechanism. This provides us the mechanism to synchronize different media objects like video, audio, and text on one location.

SMIL does not provide any mechanism for synchronization of play-out across different location, so inter-destination synchronization capability was not present in current implementation of ambulant player. We decided to extend the current implementation of ambulant player to provide this functionality. We built a plug-in for ambulant which provides the support for inter-destination synchronization mechanism between ambulant players running at different locations. Details of the implementation of this plug-in are presented in chapter 4 of this thesis.

Another important step was to make the communication possible between ambulant players running at different destinations. We decided to build a communication platform over internet transport layer protocol. By using this communication mechanism the ambulant players at distinct locations can exchange the control information necessary for synchronization. Later on we design the different protocol and algorithms for exchange to control information. The detail of these protocol and algorithms is presented in chapter 4.

Experiment Set-up: SMIL provide well-tested mechanism for alternative content selection based on available bandwidth. Rather focusing on alternative content selection, we decided to focus on the synchronization of the videos between two ambulant players. Our decision to measure the play-out difference only in videos was derived due to following reasons.

- In document-based media all type of media (video, audio, text etc.) use same document clock for synchronization purpose. In our synchronization techniques we are synchronizing this document clock across distributed participants. So measuring play-out differences in only one type of media is enough.

- It is easier to measure the play-out difference in case of videos. To measure play-out differences we need clearly distinguishable brakes in continuous media. Video with frequent scene changes as source media ensures we have these clearly distinguishable brakes. Tools are available to measure the frame differences in two videos. In case of other media like audio measuring the play-out difference is difficult due to two reasons. At first place it is difficult to produce audio with clearly distinguishable brakes and second unavailability of specialized tools to measure play-out difference in audio data. It is possible to measure the play-out differences in these media using subjective evaluation by involving end users, but it was out of the scope of this thesis.

Once the SMIL presentations are synchronized at two destinations, the challenge was to measure the synchrony between two presentations. For this purpose we decided to record the video of SMIL presentations at two locations by placing the monitors of two computers side by side. Now the SMIL presentations of two locations are recorded in one video and we can measure the time difference between two destinations. For the evaluation purpose we made a SMIL presentation with as many as 50 scene changes so that we can measure the play out difference at every scene change at two SMIL presentations. The experiment set up is discussed in detail in section 5.1.

Measurements: To measure the play out difference between two SMIL presentations we evaluated the recorded video of the presentations by two methods: The manual and automated method. In manual method we used the *virtualdub* software to go through the video frame by frame and record the frame difference of every scene change in the SMIL presentation. Once the frame difference was calculated we can calculate the time difference between play out of presentation. For automated evaluation we used software which detects the scene changes and their frame numbers at two SMIL presentations in recorded video. We used this automated evaluation for validation of the results gathered by manual evaluation.

We measure the play out difference of the SMIL presentations at two destinations for every algorithm and also measure the play out difference in absence of our synchronization mechanism. Then we compared the results of our algorithms with the results gathered in absence of our synchronization algorithms. We also compared the results of algorithms among themselves.

In addition to the measurements of play-out differences we also measured the difference between event execution time at two ambulant players on two distant locations. These events include play, pause, resume, and navigation event. We particularly focused on navigation event. In case of navigation event, play-out at two locations will be de-synchronized. By virtue of our synchronization mechanism play-out will be synchronized again. We measured the difference between the event execution, period of de-synchronization and time it take to re-synchronize. As per our knowledge, inter-destination synchronization of events has not been studied in document-based media. Details of these measurements are presented in section 5.2.

3.2. A brief intro of the SMIL

In this section we will briefly discuss the SMIL language. For simplicity we are discussing the most relevant part of SMIL in this thesis. We are ignoring some key features of SMIL like layout and metadata. Detailed description of these sections can be seen in [41].

Synchronized Multimedia Integration Language (SMIL, pronounced "smile") is the first member in the family of open, XML-based standards developed and supported by the World Wide Web Consortium (W3C). It can be used not only to develop time-based multimedia presentations, but also to implement media-rich interfaces for PC or embedded applications and devices.

SMIL allows integrating a set of independent multimedia objects into a synchronized multimedia presentation. Using SMIL, an author can

- describe the temporal behavior of the presentation
- describe the layout of the presentation on a screen
- associate hyperlinks with media objects

3.2.1 SMIL Interactions: For the interaction designer, SMIL is useful for any presentation or interface requiring time-based interactions with media. SMIL enforces the separation of structure and media in a presentation or interface. By design, the media objects themselves are kept separate from the SMIL document describing the presentation. This greatly simplifies maintenance and eases the task of, for example, using a single SMIL document to describe a presentation format while updating the media to produce a new presentation. It also allows the media objects to change over time, even though the basic presentation logic remains the same. SMIL is used to determine the interaction between media elements. It does not define the media itself. Media elements can be audio, video, text, decorated text such as HTML or time-based text.

3.2.2 Discrete and continuous Media: SMIL includes support for declaring media, using element syntax. The media that is described by these elements is described as either discrete or continuous:

Discrete: The media does not have intrinsic timing, or intrinsic duration. These media are sometimes described as "rendered" or "synthetic" media. This includes images, text and some vector media.

Continuous: The media is naturally time-based, and generally supports intrinsic timing and an intrinsic notion of duration (although the duration may be indefinite). These media are sometimes described as "time-based" or "played" media. This includes most audio, movies, and time-based animations.

3.3.3 SMIL Basic Structure: SMIL's declarative syntax to describe the interaction between time-based media relies on three primary constructs, or containers: *par, seq*, and *excl*. *Par* is for parallel, *seq* for sequential, and *excl* for exclusive. Two media objects placed in a par container play in parallel, meaning they both play at the same time. Two media objects placed in a *seq* play sequentially. That is, they play one after another. *Excl* means that only one media element in the group of elements can play at a time. Usually, some sort of event logic is used to determine which media object is playing. This event can be end of another event or user created event.

Each of SMIL's time containers define a local timeline, in which a group of related media objects can be managed. The nature of the time container provides a basic set of activation constraints that eases the designer's task of creating a presentation. A *seq* container imposes general slideshow-like temporal constraints among the objects: Adding new slides with default timing is easy, since only a new media reference needs to be added to the SMIL file. None of the timings on individual objects needs to be changed. In a *par* container, a common multi-track timeline is defined that provides a common reference time base for the activation of multiple objects. The *par* and *seq* containers can be nested, allowing an audio track to accompany a slideshow, or to provide several logical collections of media objects within the same presentation context.

3.3.4 SMIL Timing Model: SMIL Timing defines elements and attributes to coordinate and synchronize the presentation of media over time. The term media covers a broad range, including discrete media types such as still images, text, and vector graphics, as well as continuous media types that are intrinsically time-based, such as video, audio and animation. The following concepts are the basic terms used to describe the timing model. The detailed specification can be seen in [41].

Time containers: Time containers group elements together in time. They define common, simple synchronization relationships among the grouped child elements. In addition, time containers constrain the time that children may be active. Several containers are defined, each with specific semantics and constraints on its children.

Description of Timing: The time model description uses a set of adjectives to describe particular concepts of timing, which are as under:

- *Implicit:* This describes a time that is defined intrinsically by the element media (e.g. based upon the length of a movie), or by the time model semantics (e.g., duration of par time container).
- *Explicit:* This describes a time that has been specified by the author, using the SMIL syntax.
- *Desired:* This is a time that the author intended - it is generally the explicit time if there is one, or the implicit time if there is no explicit time.
- *Effective:* This is a time that is actually observed at document playback. It reflects both the constraints of the timing model as well as real-world issues such as media delivery.
- *Definite:* A time is definite if it is resolved to a finite, non-indefinite value.

Local time and global time: Global time is defined relative to the common reference for all elements, the document root. This is sometimes also referred to as *document time*. Within a document, when a given element is active or "plays", the contents of that element progress from the beginning of the active duration to the end of the active duration. There will also be a progression from the beginning to the end of each simple duration. It is often convenient to talk about times in terms of a given element's simple duration or its active duration. Generically, this is referred to as local time, meaning that times are relative to an element-local reference. The following terms are used to more precisely qualify local times:

- *Active time*: Time as measured relative to the element's active duration. A time is measured as an offset from the active begin of the element.
- *Simple time*: Time as measured relative to the element's simple duration. A time is measured as an offset from the beginning of a particular instance of the simple duration.
- *Media time*: Time as measured relative to the element's media duration. A time is measured as an offset from the beginning of the media, as modified by any clipBegin or clipEnd attributes.

To be meaningful, these terms are described relative to some element. For example, when describing timing semantics, element active time refers to active time for the element under discussion, and parent simple time refers to simple time for that element's parent. When measuring or calculating time, a reference element and the local time are specified. The measured time or duration is defined in terms of the element time progress. E.g. if the reference element pauses, this may impact the semantics of times or durations measured relative to the element.

3.3.5 SMIL Synchronization: SMIL Timing defines elements and attributes to coordinate and synchronize the presentation of media over time. The main objective is to synchronize all media with *document time*. The term media covers a broad range, including discrete media types such as still images, text, and vector graphics, as well as continuous media types that are intrinsically time-based, such as video, audio and animation. Three synchronization elements support common timing use-cases. These elements are referred to as *time containers*. They group their contained children together into coordinated timelines.

- The *seq* element plays the child elements one after another in a sequence.
- The *excl* element plays at most one child at a time, but does not impose any order.
- The *par* element plays child elements as a group (allowing "parallel" playback).

The *par/seq/excl* time-container structure of SMIL provides a default set of timing relations among objects. In many cases, all timing within a presentation can be determined by simply placing objects in an appropriate time container. The default timing can be further refined by using a collection of timing attributes: attributes that add a specific begin offset, an end time, or duration to a media object.

SMIL Timing also provides attributes that can be used to specify an element's timing behavior. Elements have a *begin*, and a simple duration. The *begin* can be specified in various ways for example, an element can begin at a given time, or based upon when another element begins, or when some event (such as a mouse click) happens. The simple duration defines the basic presentation duration of an element. Elements can be defined to repeat the simple duration, a number of times or for an amount of time. The simple duration and any effects of repeat are combined to define the active duration. When an element's *active duration* has ended, the element can either be removed from the presentation or frozen (held in its final state), e.g. to fill any gaps in the presentation. An element becomes *active* when it begins its *active duration*, and becomes inactive when it ends its active duration. Within the active duration, the element is active, and outside the active duration, the element is inactive.

The attributes that control these aspects of timing can be applied not only to media elements, but to the time containers as well. This allows, for example, an entire sequence to be repeated, and to be coordinated as a unit with other media and time containers. While authors can specify a particular simple duration for a time container, it is often easier to leave the duration unspecified, in which case the simple duration is defined by the contained child elements. When an element does not specify a simple duration, the time model defines an implicit simple duration for the element. For example, the implicit simple duration of a sequence is based upon the sum of the active durations of all the children.

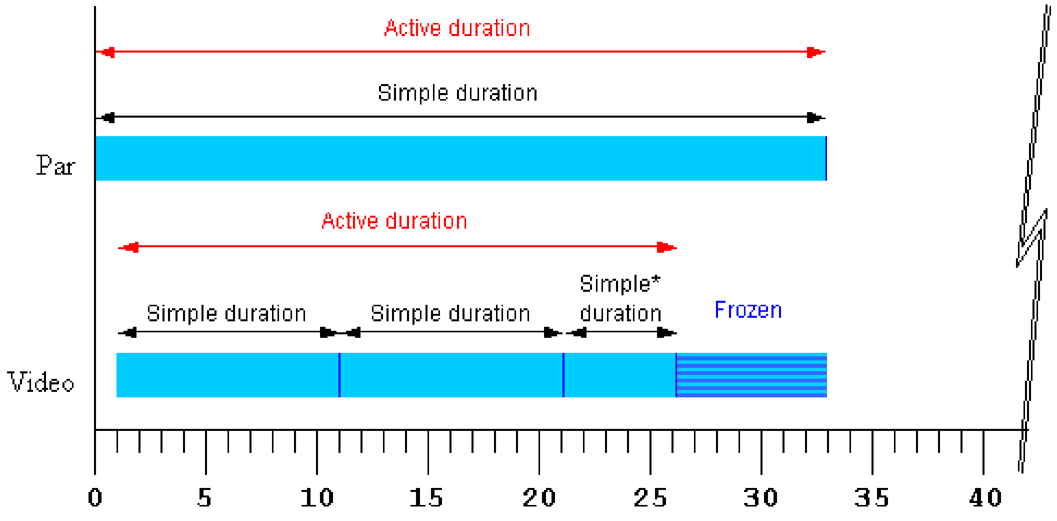


Figure 3.1: illustrates the basic support of a repeating element within a simple "par" time container. Figure taken from [41]

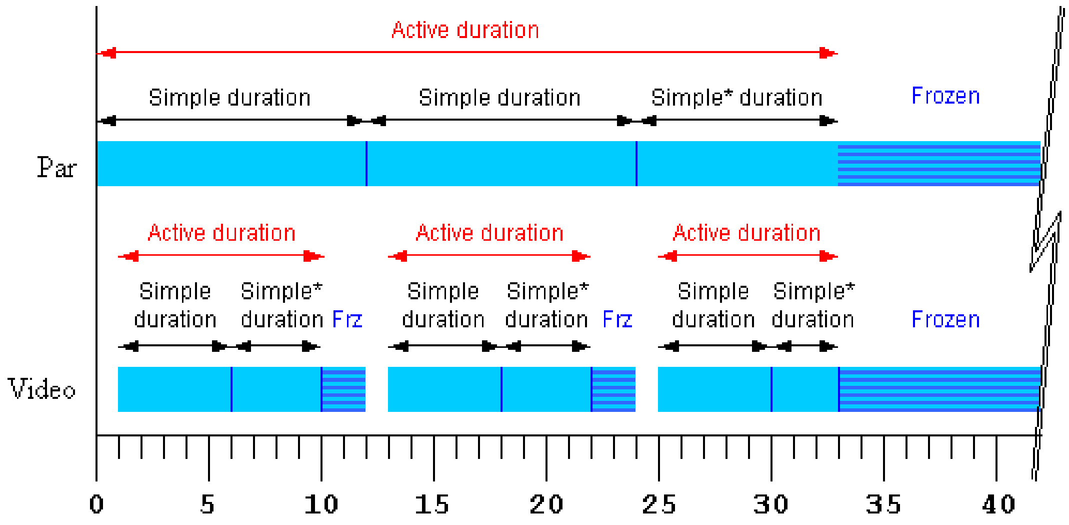


Figure 3.2 illustrates the effects of a repeating par time container as it constrains a video child element. . Figure taken from [41]

Each time container also imposes certain defaults and constraints upon the contained children. For example in a *seq*, elements begin by default right after the previous element ends, and in all time containers, the active duration of child elements is constrained not to extend past the end of the time container's simple duration.

The timing attributes in SMIL override any timing within a particular media object. For example, a clip can be trimmed or extended using SMIL timing attributes. The attributes clipBegin/clipEnd also allows a fragment of a larger video to be played. Unlike other formats such as MPEG4, which define a timeline based on the encoding of a particular video object, SMIL provides a flexible timeline that abstracts timing away from the media and into the overall presentation.

SMIL also provides time navigation within a presentation via temporal hyperlink architecture. Jumping from one object to another in a presentation has the effect of adjusting the presentation timeline to the context of the link's destination anchor. This allows all of the related content that would otherwise be active when the destination had been reached normally to also be active once the link is followed. It is the SMIL scheduler that determines the temporal relationship among elements, meaning that each of the individual media objects does not need to be aware of the presence of other media in the presentation, which is a major benefit of SMIL.

3.3.5 SMIL events and interactive timing: Begin and active end times in SMIL can be specified to be relative to events that are raised in the document playback environment. This supports declarative, interactive timing. *Interactive* in this sense includes user events such as mouse clicks, events raised by media players like a media complete event, and events raised by the presentation engine itself such as a pause and resume events.

Interaction in SMIL is provided by a declarative event-based architecture that distinguishes between internal player events (such as a media object's beginning or terminating) and external user events (such as an object within the presentation like a next stop, pause, resume, play button, selected interactively by a user). Nearly all SMIL timing-related actions define a *begin* or *end* event. This allows companion media objects to be scheduled interactively, based on the duration or (conditional) activation of related content. No scripts are required to control this interactivity; instead, a *begin* or *end* condition is set on the companion object.

For user-centered interaction, SMIL provides a mechanism for associating user events such as mouse clicks with the stat or end of either individual media objects or with SMIL timing containers. This allows basic interaction within a presentation with the need to invoke a scripting architecture. SMIL also provides basic support for interaction using a DOM interface.

3.3.6 SMIL Time graph: A time graph is used to represent the temporal relations of elements in a document with SMIL timing. Nodes of the time graph represent elements in the document. Parent nodes can "contain" children, and children have a single parent. Siblings are elements that have a common parent. The links or "arcs" of the time graph represent synchronization relationships between the nodes of the graph.

The SMIL Timing Model defines how the time container elements and timing attributes are interpreted to construct a time graph. The time graph is a model of the presentation schedule and synchronization relationships. The time graph is a dynamic structure, changing to reflect the effect of user events, media delivery, and DOM control of the presentation. At any given instant, the time graph models the document at that instant, and the semantics described in this module. However, as user events or other factors cause changes to elements, the semantic rules are re-evaluated to yield an updated time graph.

When a *begin* or end value refers to an event, or to the *begin* or *active* end of another element, it may not be possible to calculate the time value. For example, if an element is defined to begin on some event, the begin time will not be known until the event happens. Begin and end values like this are described as unresolved. When such a time becomes known (i.e. when it can be calculated as a presentation time), the time is said to be resolved. A resolved time is said to be definite if it is not the value "indefinite

In an ideal environment, the presentation would perform precisely as specified. However, various real-world limitations (such as network delays) can influence the actual playback of media. How the presentation application adapts and manages the presentation in response to media playback problems is termed runtime synchronization behavior. SMIL includes attributes that allow the author to control the runtime synchronization behavior for a presentation.

3.3. ambulant SMIL player

In this section we will briefly discuss the Ambulant Player. Design of the ambulant core is discussed in section 4.1. More details about the design of ambulant player can be seen in [40, 45]. Most of the material in this section is taken from [40, 45] and adapted to the context of this thesis.

The ambulant Open SMIL Player is an open-source, full featured SMIL 3.0 player. It is intended to be used within the researcher community (in and outside our institute) in projects that need source code access to a production-quality SMIL player environment. It may also be used as a stand-alone SMIL player for applications that do not need proprietary media formats. The player is available in distributions for Linux, Macintosh, and Windows systems ranging from desktop devices to PDA and handheld computers.

The ambulant target community is not viewers of media content, but developers of multimedia infrastructures, protocols and networks. The ambulant player is complete implementation the existing partial SMIL implementations produced by many groups. The user interface of the ambulant player is shown in Fig 3.3.

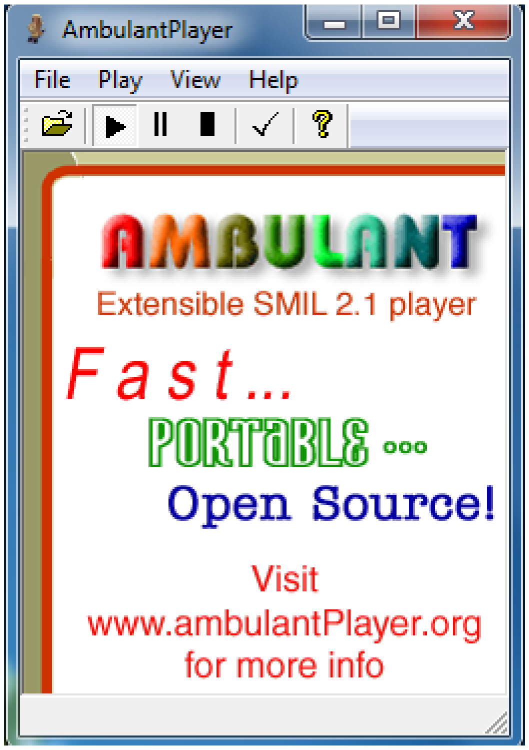


Figure 3.3 ambulant player

The ambulant Player Core Architecture

Figure 3.4 shows a slightly abstracted view of the ambulant core architecture. The view is essentially that of a single instance of the ambulant player. Although only one class object is shown for each service, multiple interchangeable implementations have been developed for all objects during the player's development. As an example, multiple schedulers have been developed to match the functional capabilities of various SMIL profiles.

Arrows in the figure denote that one abstract class depends on the services offered by the other abstract class. Stacked boxes denote that a single instance of the player will contain instances of multiple concrete classes implementing that abstract class: one for audio, one for images, etc. All of the stacked-box abstract classes come with a factory function to create the instances of the required concrete class.

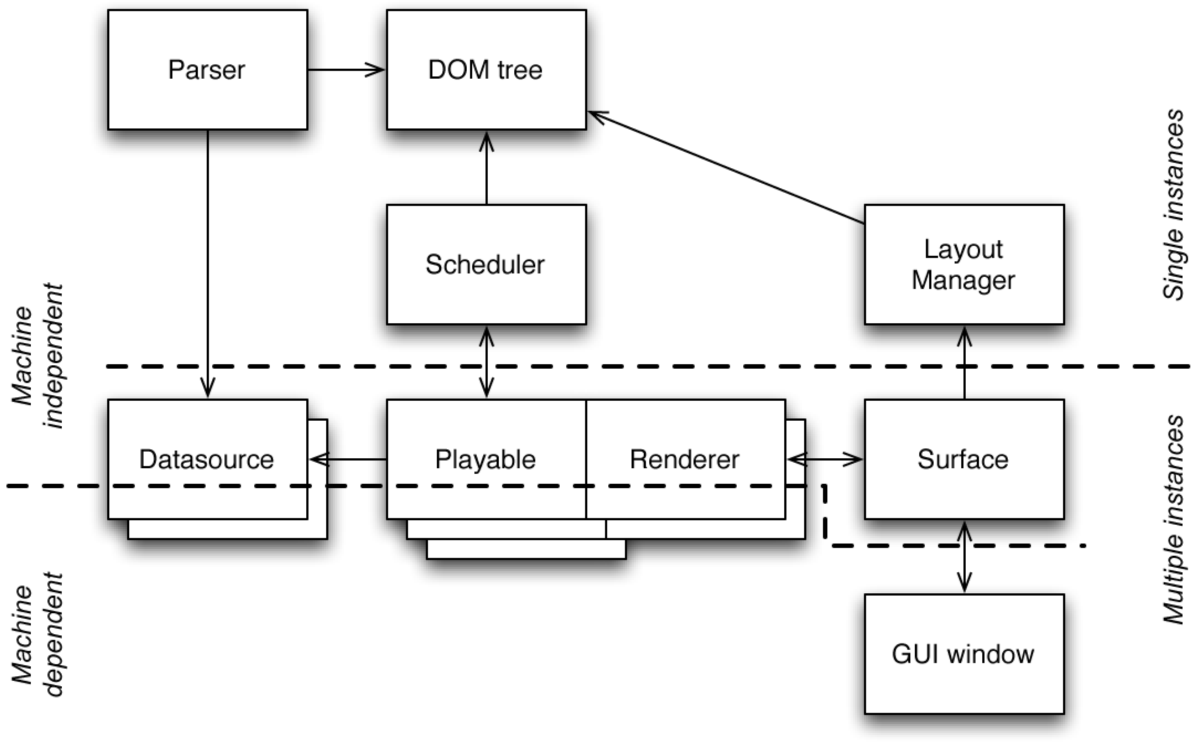


Figure 3.4 ambulant player core architecture Figure taken from [45]

.

The bulk of the player implementation is architected to be platform independent. This platform independent component has already been reused for five separate player implementations. The platform dependent portions of the player include support for actual rendering, UI interaction and data source processing and control.

When the player is active, there is a single instance of the scheduler and layout manager, both of which depend on the DOM tree object. Multiple instances of data source and playable objects are created. These interact with multiple abstract rendering surfaces. The playable abstract class is the scheduler interface (play, stop) for a media node, while the renderer abstract class is the drawing interface (redraw). Note that not all payables are renderers (audio, SMIL animation).

The architecture has been designed to have all components be replaceable, both in terms of an alternative implementation of a given set of functionality and in terms of a complete re- purposing of the player components. In this way, the ambulant core can be migrated to being a special purpose SMIL engine or a non-SMIL engine (such as support for MPEG-4 or other standards).

The abstract interfaces provided by the player do not require a “SMIL on Top” model of document processing. The abstract interface can be used with other high-level control models (such as in an XHTML+SMIL implementation), or to control non-SMIL lower-level rendering (such as timed text). Note that in order to improve readability of the illustration, all auxiliary classes (threading, geometry and color handling, etc.) and several classes that were not important for general understanding (player driver engine, transitions, etc.) have been left out of the diagram.

Chapter 4

System Architecture and Implementation

For evaluation of inter-destination techniques in document based media, it is important to have a platform which facilitates implementation and evaluation of inter-destination techniques designed for doc-based media. This chapter addresses the issues related to design and implementation of such a platform. The design of this platform should be open enough to implement different techniques designed for inter-destination. Detailed design of the platform is discussed in section 4.1. In particular this chapter answers the research question 1.1.

Research Question 1.1: Can we create a platform which facilitates the implementation and evaluation of existing stream-based media synchronization techniques for document-based media presentations? Which techniques are relevant for synchronization of document-based media presentation?

Inter-destination techniques discussed in chapter 2 are related to stream-based media. We adopted these techniques to work with document-based media. The algorithm design of these techniques is also discussed in this chapter in section 4.2. As discussed in chapter 2, there are three classes of techniques for inter-destination synchronization. Due to time limitations we designed and implemented only two classes of techniques to work with document-based media. In rest of the chapter, section 4.1 discusses the design and implementation of the platform which facilitates implementation and evaluation of inter-destination techniques for document-based media. Section 4.2 discusses implementation of inter-destination techniques using the platform and adaptation issues to work with document-based media.

4.1 System Architecture

This section presents the architecture of the platform which facilitates the implementation of inter-destination synchronization in document-based media. Inter-destination synchronization in document-based media for this research work is defined as “document- based media play-out at geographical distributed users, communicating with each other in order to synchronize play-out among them”. Application level and network level views of such inter-destination synchronization mechanism are presented in Fig 4.1.

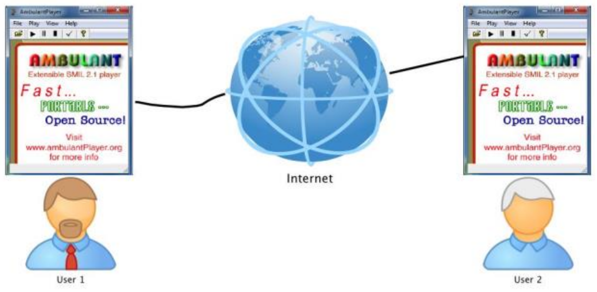

AMBULANT
Extensible SMIL 2.1 player
Fast...
Portable...
Open Source!
Visit www.ambulantPlayer.org for more info
Internet
User 1
User 2

Figure 4.1 Network/Application level view of system

This section defines an architecture that is used throughout this chapter to achieve the requirements of inter-destination synchronization in document-based media. These requirements are:

1. *An efficient mechanism to play-out document-based media at geographically distributed locations (participants).*
2. *An efficient mechanism exchange of control information between distributed participants.*
3. *An efficient mechanism to control the play-out at distributed user, according to received control information from other participating participants.*

We discuss these requirements one by one in the remaining part of this section.

4.1.1 Ambulant as document-based media player

We used ambulant Open SMIL Player for play-out of the document based media at geographically distributed users. It may also be used as a stand-alone SMIL player for applications that do not need proprietary media formats. The architecture of the ambulant player core with its modules which are relevant to our work is presented in Fig 4.2.

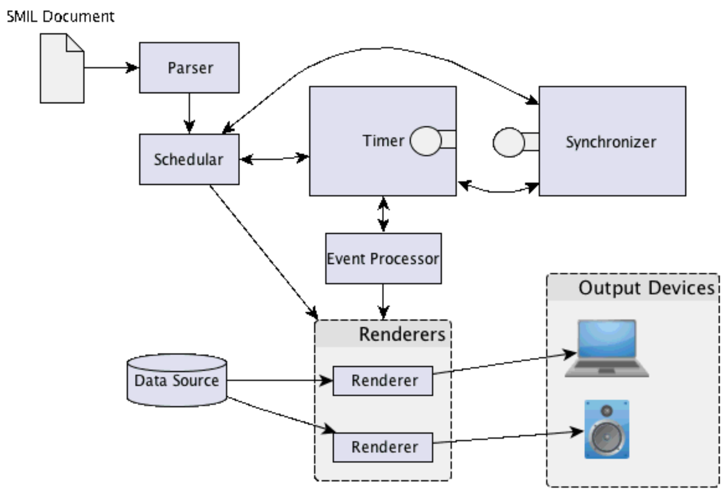


Figure 4.2 Architecture of ambulant-player

For the simplicity, only the most relevant parts of the ambulant core architecture are included in this figure 4.2 when compared to Fig 3.4. SMIL document contain timing and synchronization information of all streams associated with the presentation. The role of the parser is to parse the SMIL document and create the DOM graph. The nodes of the DOM graph are the SMILE elements with timing information. The relation between the nodes represents the synchronization of the SMIL multimedia presentation.

The Scheduler module knows document time by having access to DOM time graph. It is responsible for all the timing related issues which include, increasing or reducing the speed of the presentation, reading the current time value and many others. Scheduler both depend on the DOM graph objects. The scheduler is responsible for playing/pausing the media node while the renderer is responsible for rendering the media object on associated device. In data source multiple instance of the play-able objects are created as per specification of DOM graphs objects. The data source objects are responsible for retrieving data and delivering it to the renderers.

The Timer module is responsible for low level timing information which includes the current time value of the document clock and the system clock.

The Event processor module is responsible for processing the event generated by the scheduler, renderer, and events generated during presentation by the interaction of user. Event processor will schedule the events with help of a timer to control the presentation state according to generated events. Events includes the action to be executed in scheduler or renderer module and the event generated by user interaction.

- Scheduler generated events: Example of scheduler generated event is start-play-back (start of media). This event will be executed by the renderer.
- Renderer generated event: Example of renderer generated event is stop-play-back (end of media). It will be executed by scheduler. Another example of render generated event can be display-next-frame from the available sources. In case of this event renderer emits to itself after it has received new data from its data source. This event will be executed by the renderer itself.
- User generated events: These events are the user interaction with the media presentation. These events will be executed by scheduler.

Inter-media synchronization is already supported by ambulant player. In this implemented support the main idea is to synchronize different type media with the global document clock. So, none of the mechanism for the inter-destination support was required to build. Our job is to reuse the existing mechanism.

For inter-media synchronization in ambulant one of the media (most of the time audio) is selected as primary media. The audio renderer will adjust the time according to the audio clock. As a result of this adjustment the other associated media will adjust their-play out to synchronize with audio media. More specifically if the audio renderer will adjust the timer, which will delay or speed up (or even skip) the "display next frame" events that the video renderer uses to drive video display.

For inter-destination synchronization we reused built-in mechanism for inter-media support. It includes delaying the document clock (global clock) and forward the document clock. If the document clock is delayed all the scheduled events will be delayed. If the document clock is forwarded it depends on the current state of the renderer. If the renderer gets an event with time too far in the past, it will ignore this event if there is another similar event to be executed

4.1.2 Exchange of Control Information

Ambulant player provides us with the functionality of document-based media but it does not provide inter-destination synchronization functionality between the distributed users. In order to get an inter-destination synchronization between participating distributed users it is necessary that they can exchange the control information between them. This control information is used to synchronize the play-out at distributed locations. This control information may include timing information, user generated event and clock information at the distributed locations. Different inter-destination synchronization techniques may use some or all types of control information depending upon the algorithm used.

We decided to extend the ambulant player to exchange the information among participants. We designed a plug-in for ambulant player for this purpose. The plugin will be responsible for the exchange of information between two participants. Figure 4.3 shows the exchange of information between ambulant players at distributed locations.

At one location the ambulant player will act as server while at the other locations it will act as client. In the server-ambulant plug-in we have two modules one is the server and other is server control. Our algorithms are implemented in your client and server modules, and the client control and server control modules only control Ambulant. The server module is responsible for all sorts of communication with client module of client-ambulant. This communication includes connection establishment, transfer of time stamps, events and clock values across ambulant-server and ambulant-client. The server-control and client-control are responsible for communication with rest of the modules of local ambulant player. Similarly the client module at client-ambulant is responsible for communication with server-ambulant and the client-control is responsible for communication with the rest of client-ambulant module.

Server-control module at server-ambulant will get the required information from the ambulant modules and will send it to the client-ambulant through the client server communication. While client-control at client-ambulant will receive the information sent by the server-ambulant and will pass it to rest of Ambulant-core modules to synchronize the media presentation at client-ambulant. The nature, frequency and type of control information exchange between the server-ambulant and client-ambulant will be discussed in section 5.2 where we discussed the algorithms for exchange of control information. In order to get the control information from the ambulant player, we implemented API's in ambulant player. A short description of these API's is presented in table 4.1

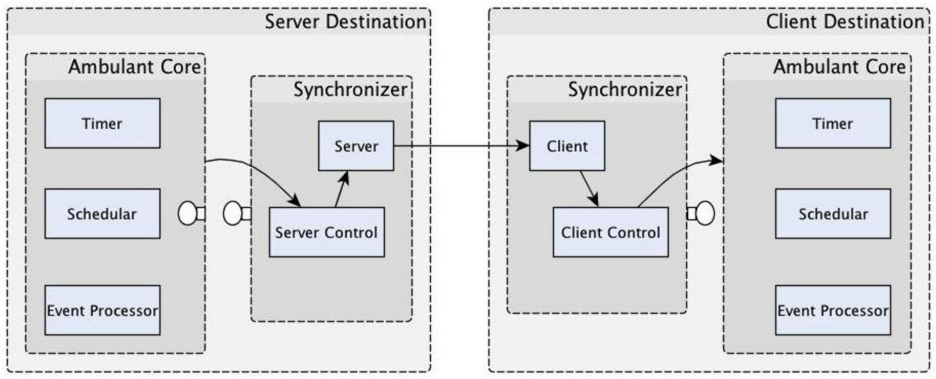


Figure 4.3 Communication flow between two ambulant players

`void Interdestinationserver::accept_client()`	Accept new participant
`void Interdestinationserver::broadcast_timestamps(time_type timestamp);`	Broadcast time stamps to all participants
`void Interdestinationserver::broadcast_events(Event event);`	Broadcast user generated event to all participants
`void Interdestinationserver::broadcast_event_skips(char* path);`	Broadcast navigation event to all participants
`int Interdestinationclient::receive_timestamps();`	Receive Time stamp from other participant
`int Interdestinationclient:: receive_events();`	Receive user generated event from other participant
`int Interdestinationclient::receive_events_skip();`	Receive navigation event from other participant

Table 4.1 API's for communication between synchronizer modules at distributed users

4.1.3 Play-out presentation mechanism

Once the control information is available at the synchronizer module, it is sent to the ambulant player which will control the play-out of the SMIL presentation. We implemented API’s for the purpose of controlling the ply-out at ambulant player. We use these API’s at synchronizer and the short description of these APIs is presented in table 4.2.

`Timer::time_type elapsed()`	Get the current time elapsed. It is the value of the document clock
`void gui_player::play()`	Start document playback
`void gui_player::stop()`	Stop document playback
`void gui_player::pause()`	Pause document playback

`signed_time_type Timer::set_drift(signed_time_type drift)`	Signals that synchronizer module has detected a clock drift. Positive values means the clock has to speed up, negative numbers that the clock has to slow down.
`gui_player:: clicked_external(lib::node *n, lib::timer::time_type t)`	Simulate a navigation action on a node, from a remote participant

Table 4.2 API's to control SMIL presentation from synchronizer module

4.1.4 Inter-destination Synchronization in Ali's scenario

One of the research objectives of the thesis is to design a platform which facilitates the implementation and evaluation of existing stream-based media synchronization techniques for document-based media presentations. We presented our synchronization algorithm test bed in fig 4.4. The different modules of test bed are presented with high level details. Our test bed provides the following primary functionalities which are necessary to implement different type of inter-destination synchronization algorithms.

- One of the participants is can start as server and other can start as client to join the user shared experience.
- Participants are able to access the local timing information of the media which is being played out by ambulant core.
- Participants are able to exchange timing information among them.
- Participants are able to control the play-out of the media at their local ambulant core.
- Participants are able to exchange events among them.

Above stated five functionalities are required to implement all type of algorithms. Apart from these basic five functionalities, our test bed provides other functionalities required to implement specific algorithms. We defer our discussion about these functionalities at moment and will discuss them while discussing specific algorithms.

We introduce the inter-destination synchronization mechanism with the example of Ali's scenario. In this section we will see how our design synchronization algorithm test bed will implement Ali's scenario. To solve the Ali's scenario discussed in chapter1, we require that ambulant player with synchronizer is available at every participating location. At Ali's place ambulant player starts in server mode. This will start communication server waiting for the clients to connect. Ali's sister and parents will ambulant player on their computer in client mode. These client-ambulant player established connection with Ali's server-ambulant player. This established the communication channel to exchange the necessary information for synchronization purposes between players at Ali's and rest of the destinations.

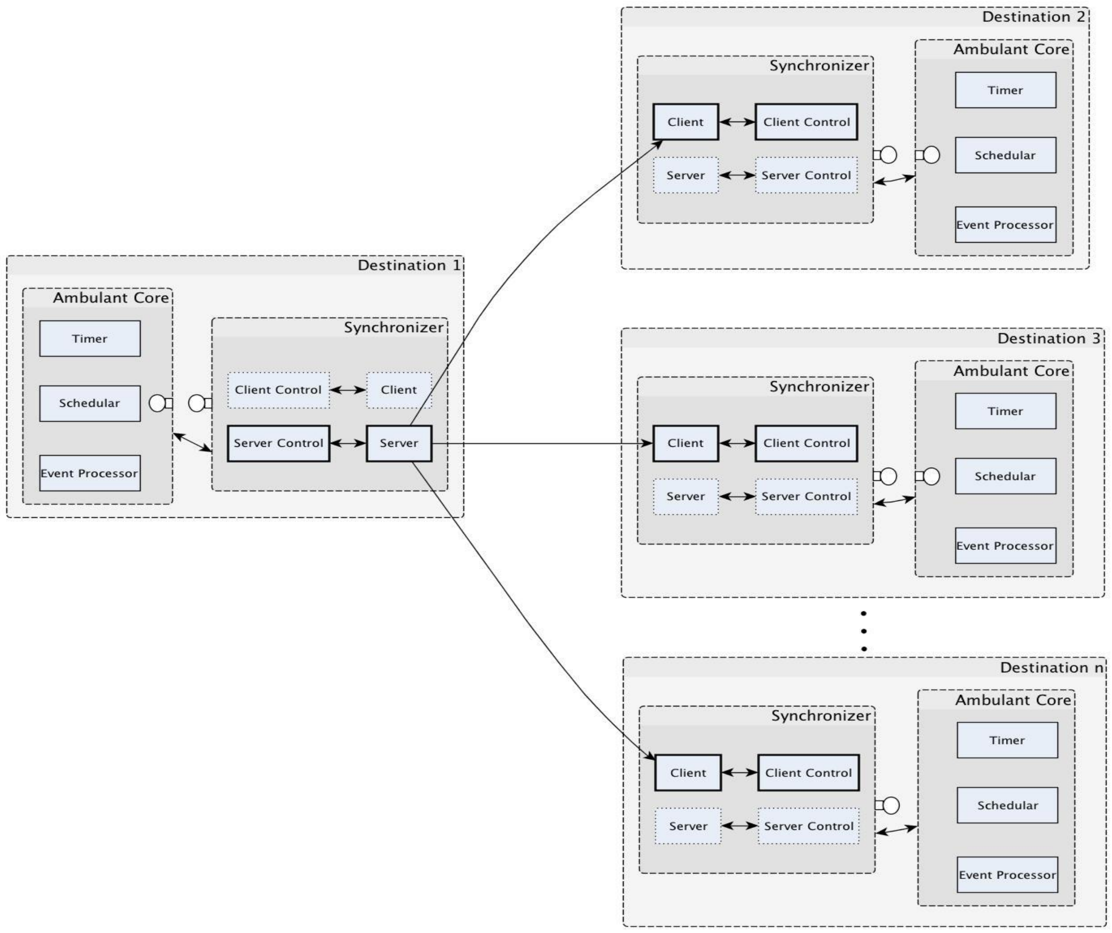


Figure 4.4 Inter-destination synchronization with ambulant player

When Ali pressed the play button at his player, this user event is communicated to the rest of participating players at remote destinations. This event is executed at all the destinations which started the SMIL presentation in ambulant players at all destinations, in synchronous manner. The next step is to keep these presentations synchronized across all destinations. For this purpose the periodic timestamp messages are communicated from server-player to clients-players depending upon the algorithm used. By virtue of these timestamps the client-player will keep them synchronized with the server-player. Any future user events will be sent to the client-player in the same fashion.

After receiving the periodic timestamp from server-ambulant, client-ambulant compared it with the local timestamp value and calculated the drift between two player’s media presentations. The client-ambulant sent this drift to the local ambulant player by using API’s. The ambulant-player will act according to received drift value, which synchronized the presentation state with the server-ambulant.

4.2 Inter-destination Synchronization Techniques

We have implemented two basic algorithms for inter destination synchronization techniques, namely Master-Slave Technique and Synchronization-Manager Technique. We have implemented some variations of these techniques which we describe in the following sections one by one.

4.2.1 Master Slave Technique

In Master-Slave, destinations are divided into a master-destination and slave-destinations. The master-destination will be in control and will periodically send its presentation time stamps (which are document clock values) independently according to its own presentation state. The slave-destinations should present at the semantically same presentation as the master-destination. In practice one SMIL presentation contain multiple type media like audio, video and text. One of these media will be selected as primary media for inter-media synchronization at each destination. This technique achieves inter-destination synchronization by adjusting the document clock value of the SMILE document at the slave-destinations to that of the master-destination.

The pictorial description of the Master-Slave technique is presented in fig 4.5. The number associated with the exchanged information shows the sequence of information exchange. The server-control at the master destination will receive the timestamp from the ambulant core. The server component will send this information to all the connected destinations. The client component at the slave destination receives this time stamp information. The client-control component will receive the local timestamp at slave destination. It calculates the drift between the presentation state of master destination and its own presentation. It sends this drift to the scheduler component which controls the presentation state.

1. *Synchronizer at Master reads time stamp from Ambulant Core.*
2. *Synchronizer at Master broadcasts al time stamp to all destinations*
3. *Synchronizer at all slave destinations read time stamp from their ambulant cores*
4. *After calculation, all the slaves' destinations will set the drift at respective ambulant core.*

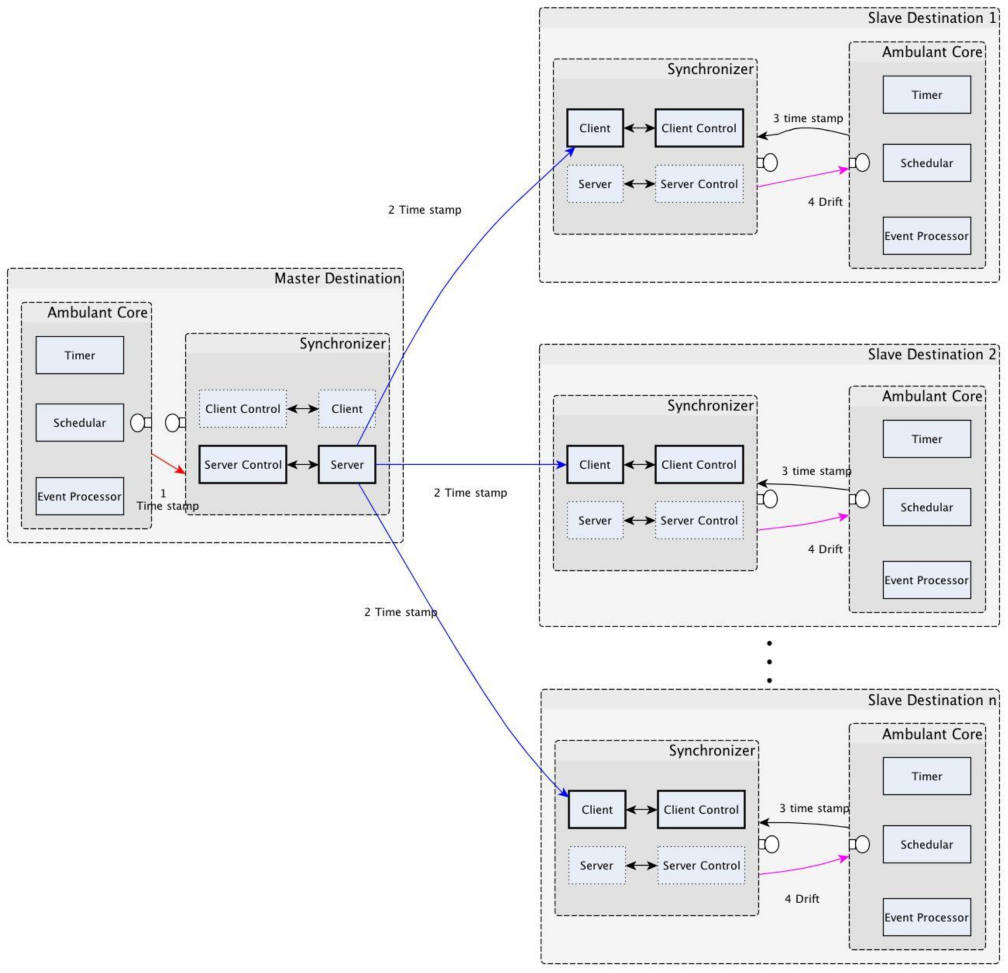


Figure 4.5 Master-slave technique with ambulant player

4.2.2 Synchronization Manager Technique

Synchronization-Manager technique does not classify destinations into master and slaves; therefore all the destinations can be handled fairly. It involves a synchronization manager in order to synchronize the primary media among all destinations. The role of synchronization manager can be performed by one of the receivers. Each destination then sends timestamp of the current presentation to the synchronization manager. The synchronization manager gathers timestamps from the destinations, and it calculates the drift between the presentations. The synchronization manager adjusts its own presentation state and will send the drift value to the other destination which will also adjust presentation state locally. Synchronization manager technique is fair compared to the master-slave technique as the burden of adjusting the presentation state is not only on one destination but mutually shared by destinations.

The pictorial description of the Master-Slave technique is presented in Fig 4.6. The number associated with the exchanged information shows the sequence of information exchange. Each destination receives the local timestamp value of the current presentation state and sends it to synchronization manager. The synchronization manager receives a timestamp of its own local presentation as well as from the other destinations. It calculates the drift and adjusts its own presentation state and sends the drift value to the participating destination. After receiving the drift value, the participating destinations will adjust the presentation state accordingly.

1. *Synchronizer at all destinations, including manager, reads time stamp to ambulant core.*
2. *Synchronizer at all destinations except manager, send this time stamp to manager.*
3. *Synchronizer at manager calculates the drift and sends back to all destinations.*
4. *Synchronizer at all destinations including manager, will set the drift at ambulant core*

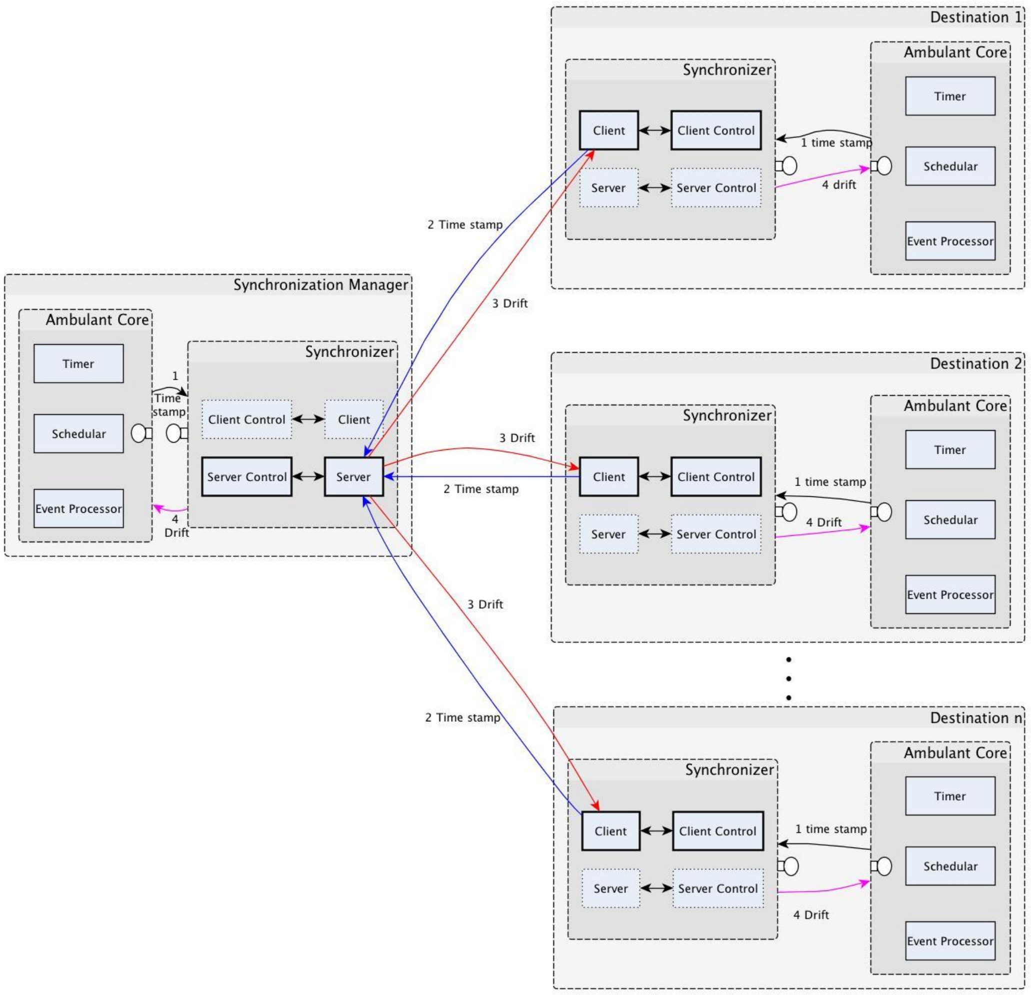


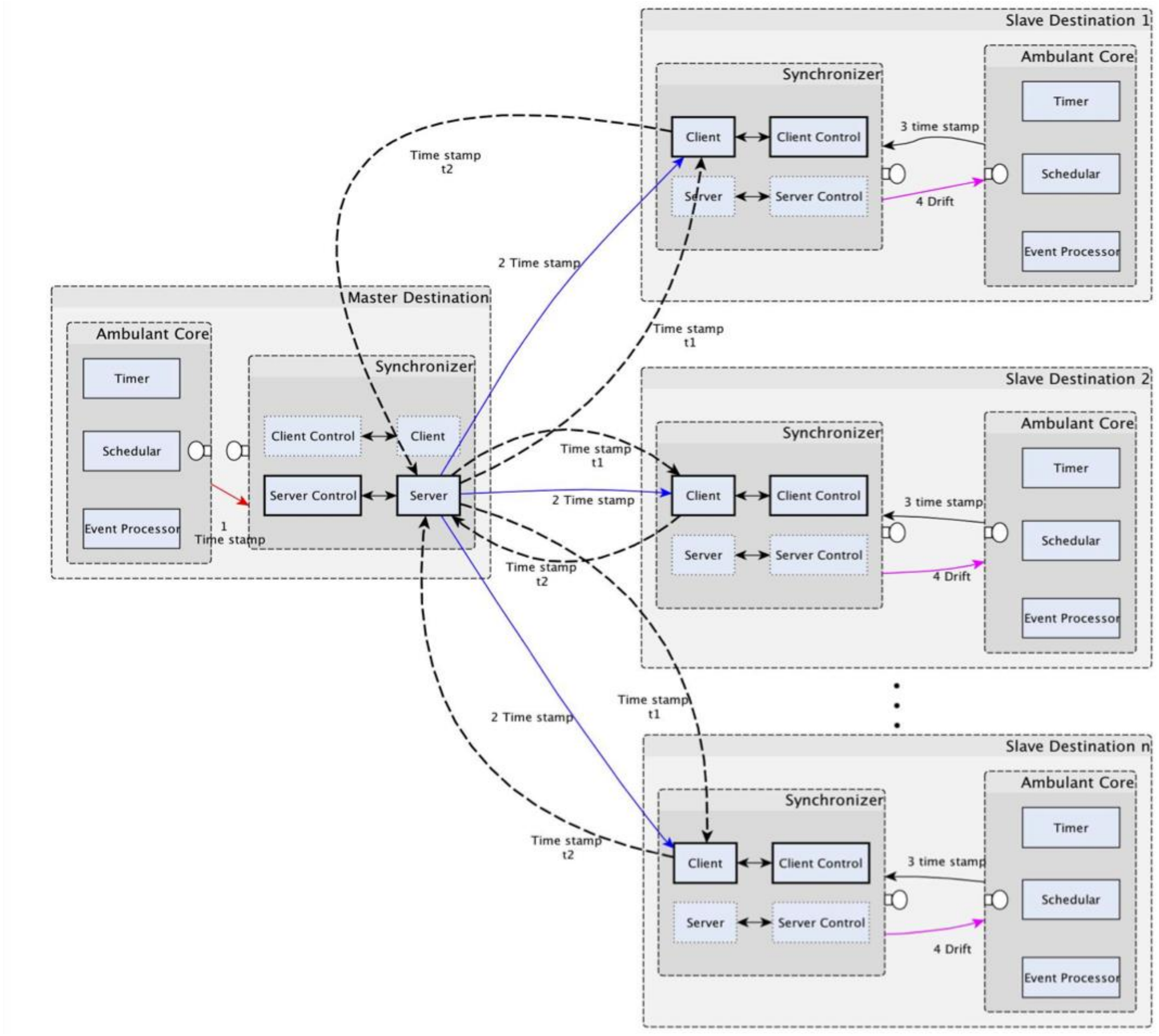
Figure 4.6 Synchronization-manager technique with ambulant Player

4.2.3 Time Synchronization of the destinations

As we are considering that the participating destinations are not globally time synchronized, so the calculated drift after comparing timestamp values may not be accurate. The global time synchronization of destination is an expensive procedure in term of processing. In order to investigate the effect of time synchronization in the above stated algorithms we develop a light weight estimated clock synchronization mechanism. The updated algorithms of master-slave and synchronization manager techniques are presented in the following section.

A. Master-slave technique with estimated time synchronization:

Every slave destination sends its clock value to the designated master destination periodically. The master destination after receiving this value will sends back the same value immediately. When the slave destination receives the clock value from the master destination it compares it with its current clock value and calculates the RTT value between itself and master destination. It will set the clock drift between itself and the master destination by half of RTT value. In every future calculation of the drift between the presentation state between master and slave destination, it will add this clock drift value to drift.



t1: All slave destinations send the clock value to master destination

t 2: Master destination sends back the same clock value to respective slave destinations

∗ Rest of communication is similar to Figure 4.5.

Figure 4.7 Master slave with estimated time synchronization

In this manner the calculated drift value will be accurate. Also the slave destination clock is virtually synchronized with the master destination. We called it the estimated clock synchronization. This kind of clock synchronization is estimated and may not be accurate as global clock synchronization, but it will give us the leverage to evaluate the effect of clock synchronization in our algorithm. The pictorial representation of the master-slave algorithm is presented in Fig 4.7.

B. Synchronization manager technique with estimated time synchronization

Every destination sends its clock value to the designated synchronization manger destination periodically. The synchronization manger destination after receiving this value will send back the same value immediately. When the destination receives the clock value from the synchronization manager destination it compares it with its current clock value and calculates the RTT value between itself and synchronization manger destination. It will set the clock drift between itself and the synchronization manger destination by half of the RTT value. In every future calculation, the destination will add this clock drift value in the presentation drift value.

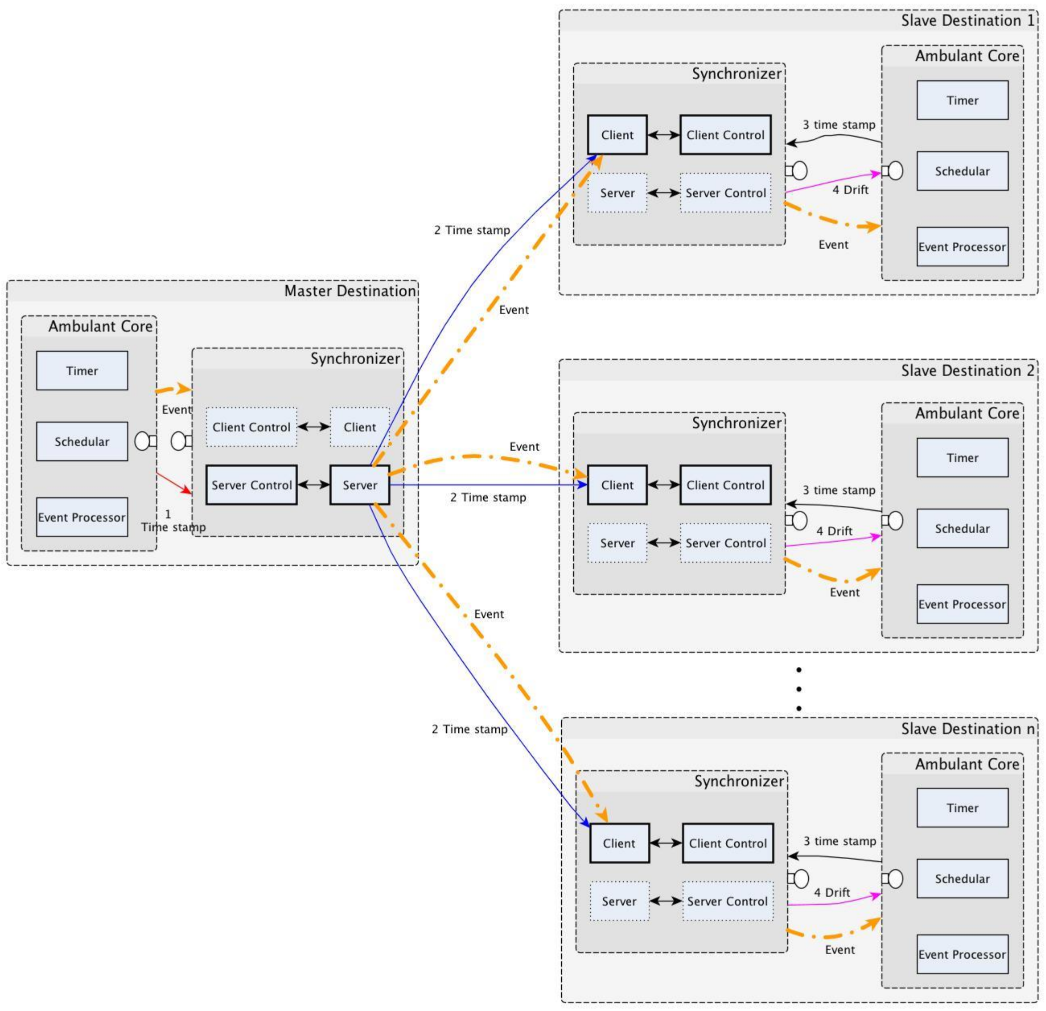
In this manner the calculated drift value will be accurate. Also the destination clock is virtually synchronized with the synchronization manger destination. We called this estimated clock synchronization. This kind of clock synchronization is estimated and may not be accurate as global clock synchronization, but it will give us the leverage to evaluate the effect of clock synchronization in our algorithm.

4.2.4 Event synchronization

For the interactive synchronous shared experience, it is necessary to exchange the user interaction events between the destinations, to keep the presentation states consistent and synchronous. These user interaction events in SMIL presentation can include play, pause, resume, stop and jump to a specific node. The exchange of the timestamp information is to keep the presentation across destinations synchronized. To keep the state of the presentation synchronized the event exchange has to be performed. We extend master slave and synchronization algorithms for event exchange. In the following section we will discuss each one by one.

A. Interactive master slave synchronization technique

Every user interaction event is captured by the ambulant player at master destination. This event is sent to the synchronizer module which sends this event to all the participating slave destinations. The client at the slave destination receives the event and will send it to scheduler module of the ambulant player at slave destination, which eventually executes this event at destination. To make the synchronous execution of the event the master destination sends a timestamp of the presentation state at the instance of occurrence of event. The pictorial representation of the interactive master slave algorithm is presented in Fig 4.8.



1. *Ambulant core at master sends user event to its synchronizer.*
2. *Synchronizer at master broadcasts this event to all destinations*
3. *Synchronizer at all slave destinations send the received event to their synchronizers*

∗ Rest of communication is similar to Figure 4.5.

Figure 4.8 Interactive master slave technique

Interactive synchronization manager technique

For synchronization of user interaction, every event is captured by the ambulant player at synchronization manager. This event is sent to the synchronizer module which sends this event to all the participating destinations. The client at the participating destination receives the event and will send it to scheduler module of the ambulant player, which eventually executes this event at destination. To make the synchronous execution of the event the synchronization manager destination sends timestamp of the presentation state at the instance of occurrence of event.

Chapter 5:

Experiments and Evaluation

In this section, we describe the experimental setup and then discuss the procedures used to validate our system. We conducted a series of experiments with different algorithms and evaluated synchronization performance on them. Each algorithm was run for 3 times to ensure its consistency on multiple runs. The results have proven that the algorithms provide consistent output in repeated trials.

In particular this chapter answers two research questions stated in chapter 1:

Research Question 1.2: *How much synchronization can existing stream-based media synchronization techniques achieve in cases of document-based media presentation?*

Document-based media provide functionality to synchronize the media contents of different objects at the same location. They do not provide the functionality of synchronizing the play-out of the contents across the destinations. This functionality has not been previously implemented in document-based media. We design and implement a platform which can allow us to implement the inter-destination synchronization algorithm for document-based media.

As inter-destination synchronization algorithms for document based media have not been studied before. It was required to study algorithms related to stream-based media present in literature. An adaptation of these techniques for document-based media was required before implementation. We discussed these techniques in chapter 4. Here we discuss performance measurement and evaluation of these techniques.

Research Question 1.3: *How much synchronization can we achieve in case of user interactions in document based media presentation? How much is the presentation at different destinations de-synchronized in case of user interaction and how long will it take to re-synchronize them?*

By using the implemented platform for inter-destination synchronization in document based media, user interactions like, 'play', 'pause', 'stop', 'resume' and 'navigation control' are executed across all participants. Here it is evaluated that for the navigation control user action how much the media content at distinct location is de-synchronized and after how much of an interval the contents are re-synchronized again.

Although, the design of our solution supports more than two participate in shared experience. But to compare algorithms we ran the solution with two participants only as to measure the differences, we can record and analyze the presentation of only two participants.

5.1. Experiment Setup

In this section we will elaborate the process of running the video presentation on ambulant players, how we recorded the video presentation, and finally how we analyzed them to conclude results. It is very important to know all these steps so we will elaborate all one by one.

SMIL provide well-tested mechanism for alternative content selection based on available bandwidth. Rather focusing on alternative content selection, we decided to focus on the synchronization of the videos between two ambulant players. Our decision to measure the play-out difference only in videos was derived due to two reasons:

- In document-based media all type of media (video, audio, text etc.) use same document clock for synchronization purpose. In our synchronization techniques we are synchronizing this document clock across distributed participants. So measuring play-out differences in only one type of media is enough.

- It is easier to measure the play-out difference in case of videos. To measure play-out differences we need clearly distinguishable brakes in continuous media. Video with frequent scene changes as source media ensures we have these clearly distinguishable brakes. Tools are available to measure the frame differences in two videos. In case of other media like audio measuring the play-out difference is difficult due to two reasons. At first place it is difficult to produce audio with clearly distinguishable brakes and second unavailability of specialized tools to measure play-out difference in audio data. It is possible to measure the play-out differences in these media using subjective evaluation by involving end users, but it was out of the scope of this thesis.

For experiment we used two computers one with *Intel Core (TM) 2 Quad 2.4 GHz* processor, with *8GB* memory, running *32-bit windows 7* operating system, while the other with *Intel T2130 1.8 GHz* processor, *1GB* memory and running 32-bit *windows 7* operating system. Both were connected with high speed internet with delay less than 1 millisecond.

Starting Presentations:

Two participants of the shared experiences run the ambulant player at two different destinations. At one destination the parameter settings are to run the player as master and on the other as slave (for master slave technique). Both the destinations then opened a SMIL presentation from the same url. The presentation then started at master destination will automatically start the presentation at the slave destination. After the start of the presentation it’s up to the synchronization technique and algorithm to synchronize the play out for the complete duration of the presentation.

Recording Presentations:

We start measuring play-out difference by recording the two subject videos as shown in Figure 4.1. The recording of the two videos is used later to compare them for play-out difference. The camera used for recording the two videos, along with the frame rate of the videos, defines the achievable accuracy of the play-out difference measurement. We used the camera with the 15 frame per second rate. The recording set up is pictorially described in Fig 5.1.

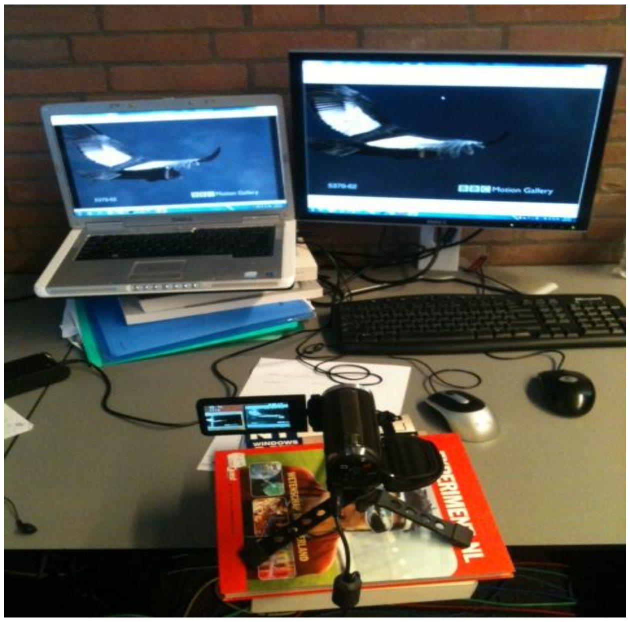


Figure 5.1 Experiment set up

Measuring play out difference:
For measuring the play out difference we used two methods, manual and automated by using software. We discuss both the methods one by one.

Manual Evaluation: For manual evaluation of the play out difference we use the *virtualdub* software [46]. Its primarily is a video capture/processing utility. It gives the facility to analyze the video frame by frame. We used this ability to measure the play out differences manually. We went through the whole recorded video of the two subject videos. Figure 4.1 shows the sequence of the screen shots to elaborate the m annual evaluation process.

When we went through the recorded video frame by frame, we could see the scene changes in the subject videos and we noted the frame number of the recorded video on the scene changes on the both of the subject videos on left and right. For example if a particular scene change occur on the left screen on frame X and the same scene change occur on the right video on frame Y, then play out difference of the two subject video will be X-Y. If the frame rate of the subjected video is 15 frames per second it mean that frame duration is 66.67 milliseconds. The play out difference of the two subjective video is (X-Y) * 66.67 second. Manual evaluation procedure is pictorially described in Fig 5.2. In Fig a sequence of four frames in recorded video are presented. In frame1 we have same scene on both the screen and both presentations are synchronized. In frame2 and frame3 the scene is changed on left screen while on right screen scene change has not occurred. In frame2 and frame3 both the presentation on left and right screen are not synchronized. In frame4 the scene changed in right frame as well which means that presentations on left and right screen are synchronized again. We are measuring the de-synchronization and re-synchronization only on scene changes as it is easy to measure the presentation differences at scene changes.

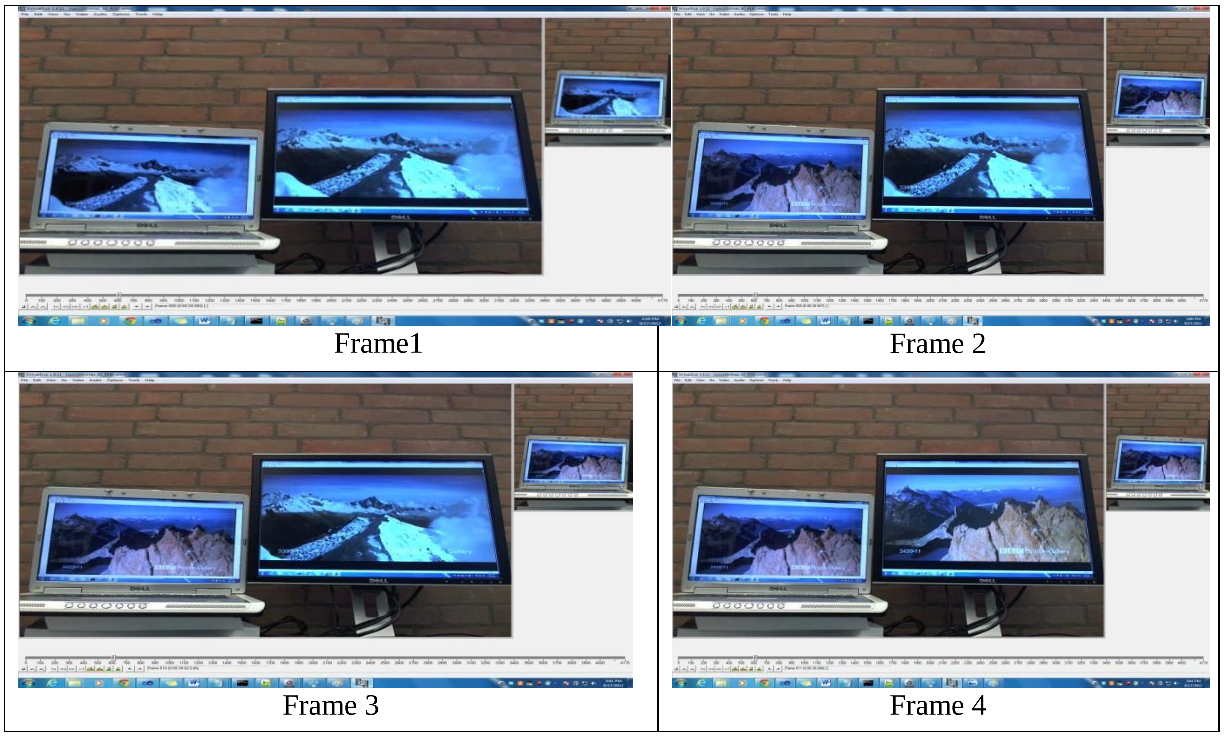


Figure 5.2 Manual evaluation procedures

Automated Evaluation: We also performed automated evaluation of the same recorded video by using software [38]. This software used the image processing routines to detect the scene changes from the video. After detecting subject videos scene changes in the left and right subject videos the software run module to extract the data relevant to these detected scene changes, This relevant data include the frame numbers of the scene changes on the left and right subject videos. We compare the frame numbers of the particular scene change on the left and right video to calculate the play out difference with same formula stated above in manual evaluation.

The software used the probabilistic methods for scene change detection and extracting data related to scene changes. More over our video recording system was not etiquette enough, so the software had some false positive and false negative errors regarding scene change detection. We deleted the false positive entries by counter checking manually in *virtualdub*. We did nothing regarding the false negative entries as it will not affect the measurements. The effect of false negative is that we had less scene changes in the video as compare to manual evaluation but enough to conclude some result.

5.2. Experimental Measurements and Results

For the experiments we had two system one with considerably less resources then the other one. We ran the same presentation on both destinations without enabling synchronizer to measure how much play out difference they experience without our synchronization mechanism. Then we ran different algorithms of synchronization techniques and compare them with the unsynchronized results. We also compared the different algorithms results with each other. In this section we will discuss the results with discussions.

Measuring Moving average: For the clarity in comparisons we not only calculate the play out differences at every scene change, but we also calculate the moving average of the play out differences.

The moving average is calculated with the following formula.

$$PDn = \frac{\sum_{i=1}^{n} \sum PDn}{n} \text{For } n \leq 4$$

$$PDn = \frac{\sum_{i=n-4}^{n} (PDn)}{5} \quad \text{For } n \geq 5$$

Where PDn is the play out difference at scene change number n.

Measuring Accumulative Moving average: In some cases, for the clarity in comparisons we not only calculate accumulative moving average with following formula.

$$PDn = \frac{\sum_{i=1}^{n} (PDn)}{n}$$

Where PDn is the play out difference at scene change number n.

5.2.1. Master Slave Synchronization Technique

Master Selection Policy: For master slave synchronization technique we ran measure the play out differences with two different settings. First we designated the faster computer as master and in second setting the slower as master.

Results for the faster master are shown in Figure 5.3. The dotted lines represent the absolute play out differences in milliseconds at scene changes. The solid lines represent the moving average of play out differences in milliseconds. The detail of moving average calculation is given in section 5.2. As moving average is the average of the current and previous four values so the moving average curve will be slightly below the absolute difference curve. This is more evident in case of unsynchronized presentations because the play-out difference is increasing in every scene change.

The average play out difference for the unsynchronized video is increasing all the time and after four and half mints presentation is over 2.4 second .From the graph we can infer that the absolute play out difference lie between 150 milliseconds and 450 milliseconds. The average play out difference is between 250 and 300 milliseconds for whole duration of the video. The results are promising in the sense that the play out difference is within the window of 250 and 300 millisecond irrespective of the length of the video and it is not increasing with the duration of the video.

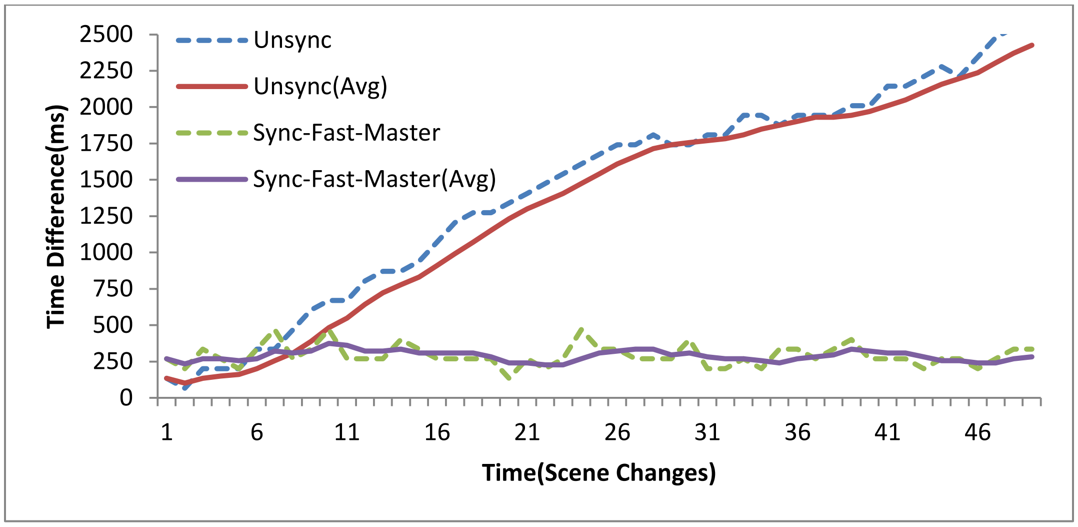


Figure 5.3Result of Master Slave (Fast Master) synchronization technique

The results are even better in case if slow destination as master and are shown in Figure 5.4. In this case the absolute play out difference is between o and 250 milliseconds and the average play out difference is below 150 milliseconds.

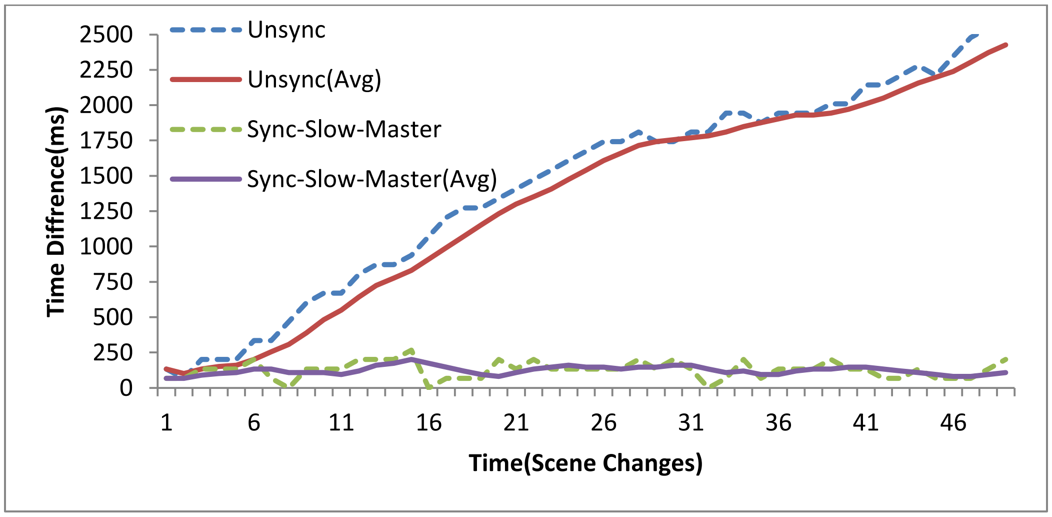


Figure 5.4 Results of Master Slave (Slow Master) synchronization technique

In Figure 5.5 we summarize the results of two master selection policies. From the Figure it is evident that the slow destination as master performs considerably better than the fast destination as master. The reason of this difference in case of fast destination as master and slows as slave is the method of play out adjustment at the slave destination. In case of slow as slave the master is always ahead of the slave due to its speed and slave always have to catch up the master. In order to synchronize with the fast master the slow slave has to skip the frames in order to play out the future frames and scheduler took time to fetch and display the future frames. In case of slow master and fast slave the slave always have to increase the frame duration or has to display the frame more than one time which do not require any fetching of the future frames so it's easy for slave to synchronize.

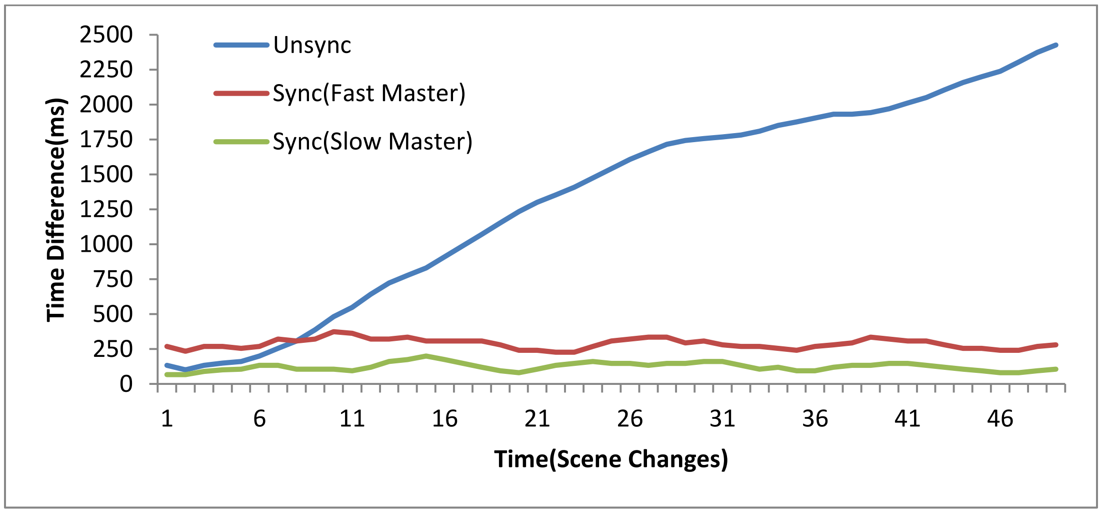


Figure 5.5 Comparison of Master Slave synchronization techniques

In Table 5.1 we presented the comparison of two master selection policies in term of minimum, maximum and average frame difference with unsynchronized case. The average play out difference in case of unsynchronized version is around 1400 milliseconds, in case of fast master are around 250 milliseconds and in case of slow master is around 125 milliseconds. So, master slave synchronization technique in general has promising results with slow master even better than the fast master case.

	Play out difference(ms)		
Technique	**Min**	**Max**	**Avg**
Unsync	67	2546	1431
Sync Fast-master	134	469	289
sync slow-master	0	268	125

Table 5.1 Comparisons of master Slave synchronization techniques

5.2.2. Synchronization Manager Technique

For synchronization manager technique we also have two settings like in master slave techniques namely fast manager and slow manager.

Fast Manager: In case of fast manager case we designated the faster computer as the synchronization manager and slow as a normal destination. Figure 5.6 presents the play out difference in this case. Throughout the duration of the video the play out difference between two destinations remained under 250 milliseconds. The average play out difference is around 100 milliseconds.

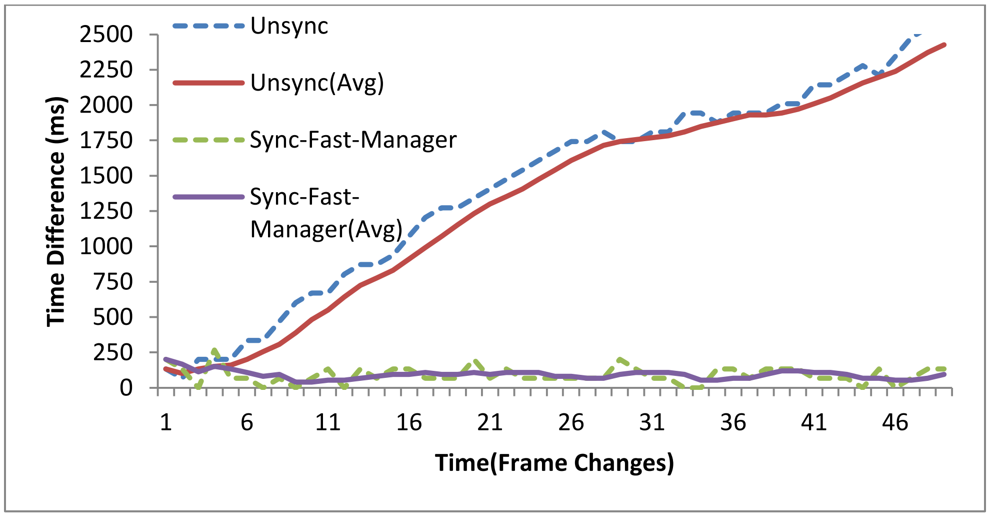


Figure 5.6 Results of Synchronization manager (fast manager) technique

Slow Manager: Play out differences in case of slow destination as manger is presented in Figure 5.7. The absolute play out difference remains in fewer than 250 milliseconds and the average play out difference are around 150 milliseconds.

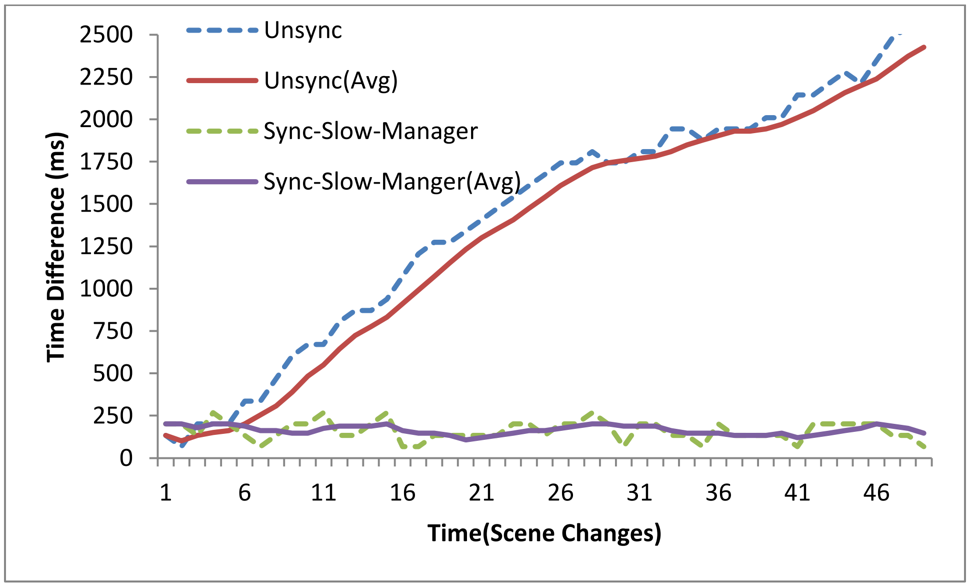


Figure 5.7 Results of Synchronization manager (slow manager) technique

We also compared the two synchronization manager cases in Figure 5.8. The Fast synchronization master remains better than the slow synchronization manager. The reason is that in case of synchronization manager the responsibility of adjusting the play out to remain synchronized lies on both the synchronization manager and other destination. Synchronization manager have to receive the timestamps from all other destinations and then distribute the calculated drift value as well. This put extra processing responsibility on synchronization manager and the fast synchronization manager performed well in this scenario.

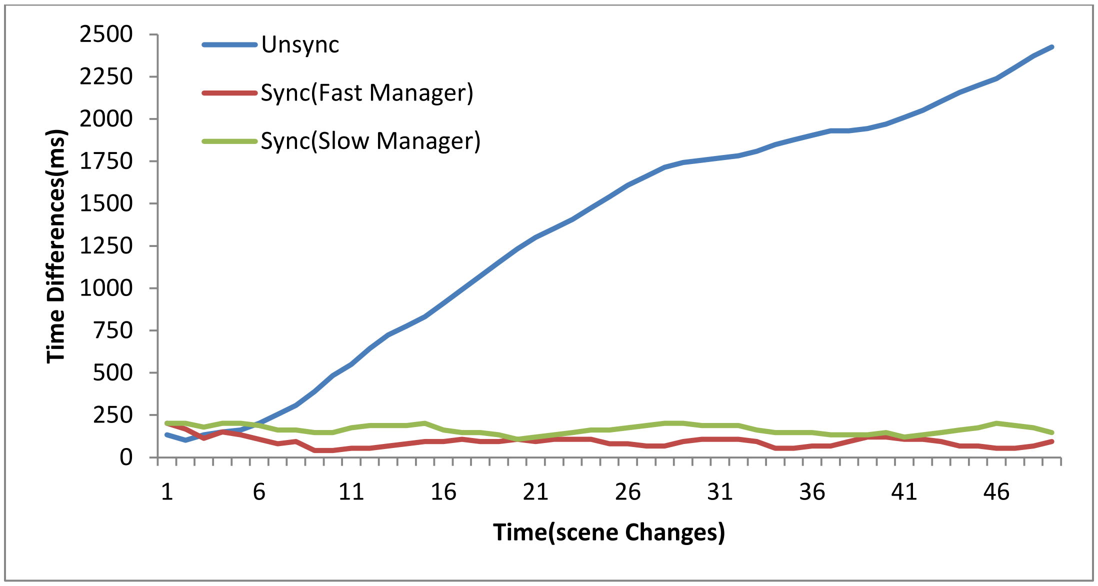


Figure 5.8 Comparison of synchronization manager techniques

Table 5.2 present the comparisons of the minimum, maximum and average play out difference in both the case of synchronization manager technique and the unsynchronized case.

	Play out difference(ms)		
Technique	**Min**	**Max**	**Avg**
Unsync	67	2546	1431
Sync Fast Manager	0	268	88
Sync Slow Manager	67	268	161

Table 5.2 Comparison of synchronization manager techniques

Comparison between master Slave and Synchronization Manager Techniques

Comparison between the master slave and synchronization manager technique is presented in Figure 9. This Figure is the answer of the research Question 1.2.

Research Question 1.2: *How much synchronization can existing stream-based media synchronization techniques achieve in cases of document-based media presentation?*

The acceptable play out difference in synchronous shared experience is not studied much in the literature, but the lower bound of the acceptable play out difference is 200 milliseconds. This value is taken from lip synchronization research [1]. In some literature [42] acceptable play out difference for synchronous shared experience are 500 milliseconds. In this section we will discuss the value achieved by our algorithms.

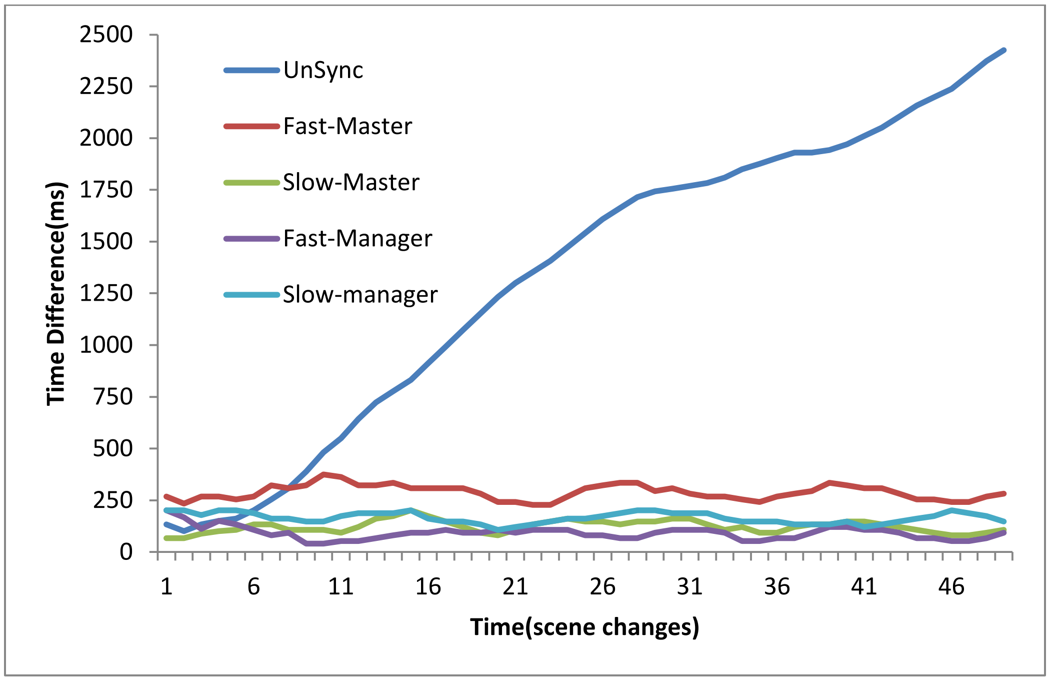


Figure 5.9 Comparison of synchronization techniques (moving average)

The comparison of the four techniques is presented in Fig 5.9. Every curve presents the moving average play-out difference of the presentation. This moving average play-out difference is calculated by averaging current and last five values. For an end user the synchronization quality will not depend only on last four five scene changes rather for whole duration of the presentation. In Fig 5.12 we presented comparison of these techniques in term accumulative average play-out difference. It is the average play out difference of the current and all previous scenes.

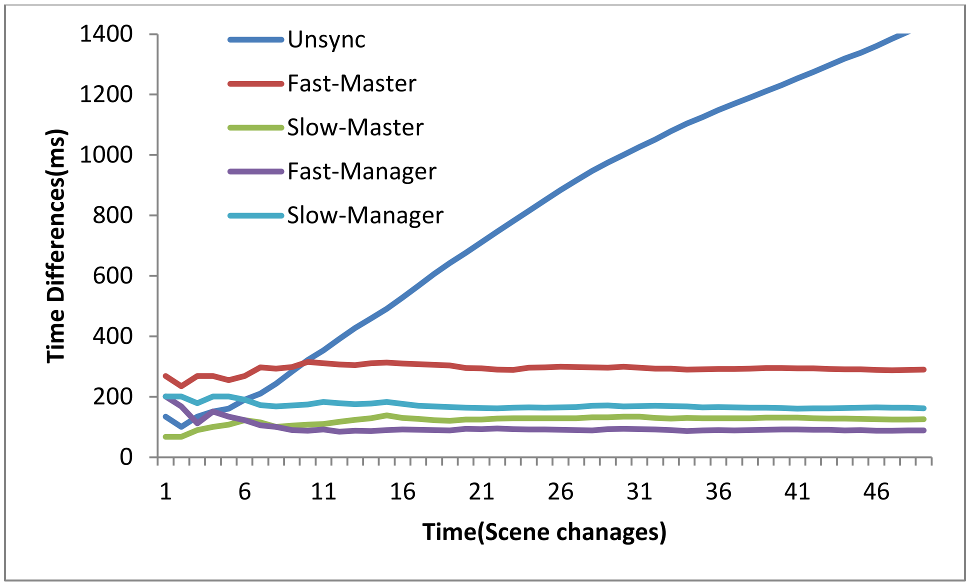


Figure 5.10 comparisons of synchronization techniques (cumulative moving average)

The average play out difference for the unsynchronized video is increasing all the time and after four and half mints video it's over 2.4 second. In case of master-slave (fast master) the results are promising in the sense that the play out difference is within the window of 250 and 300 millisecond irrespective of the length of the video and it's not increasing with the duration of the video. But it's a bit higher than the minimum acceptable difference. However the play out difference value for the other algorithms is less than 200 milliseconds.

From the Figure 5.10, it is evident that the slow destination as master performs considerably better than the fast destination as master. The reason of this difference in case of fast destination as master and slows as slave is the method of play out adjustment at the slave destination. In case of slow as slave the master is always ahead of the slave due to its speed and slave always have to catch up the master. In order to synchronize with the fast master the slow slave has to skip the frames in order to play out the future frames and scheduler took time to fetch and display the future frames. In case of slow master and fast slave the slave always have to increase the frame duration or has to display the frame more than one time which do not require any fetching of the future frames so it's easy for slave to synchronize.

Similarly the fast synchronization master remains better than the slow synchronization manager. The reason is that in case of synchronization manager the responsibility of adjusting the play out to remain synchronized lies on both the synchronization manager and other destination. Synchronization manager have to receive the timestamps from all other destinations and then distribute the calculated drift value as well. This put extra processing responsibility on synchronization manager and the fast synchronization manager performed well in this scenario.

As whole the synchronization manager techniques performed better because the responsibility of the play out adjustment is shared between the destinations, while in case of master slave techniques only slave have to adjust the play out differences.

5.2.3. Comparison with and without time synched destination

As discussed in the previous section the only the master-slave technique with fast master did not perform poorly as compare to other techniques. We did experiment with this technique in order to improve the results. We did implement a light weight protocol to virtually synchronize the document clock of the two destinations. The detail is discussed in Chapter 4.3.

As a result of this protocol every slave destination has a RTT value between the master and itself. On receiving the timestamp value the slave destination will add this value in the received timestamp in order to accurately synchronize with master destination. So the clock value is not changed at slave destination but the difference in the clock is somehow accommodated. In Figure 5.13 we present the result of master-slave with Virtual clock synchronization and master-slave without Virtual clock synchronization. The former perform slightly better than later. For better visualization the comparison of these two readings with accumulative moving average is also presented in Fig 5.12.

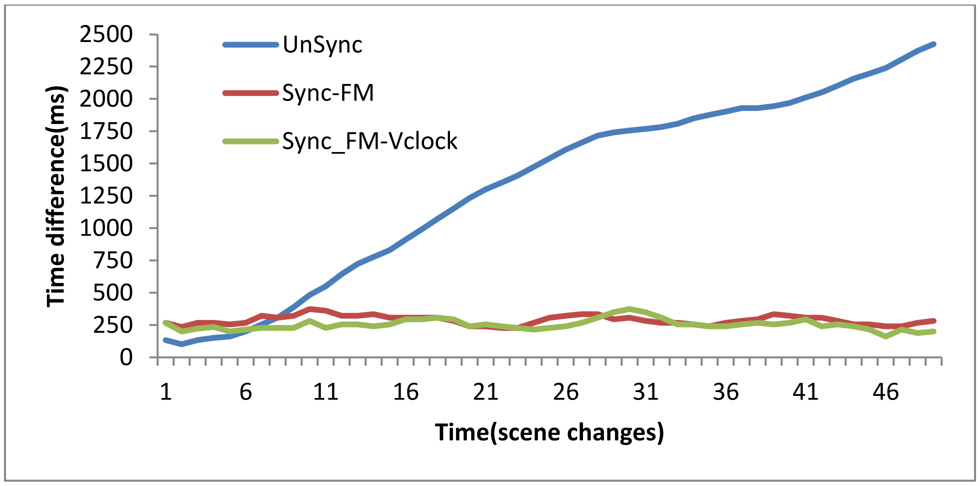


Figure 5.11 master slave with virtual clock (moving average)

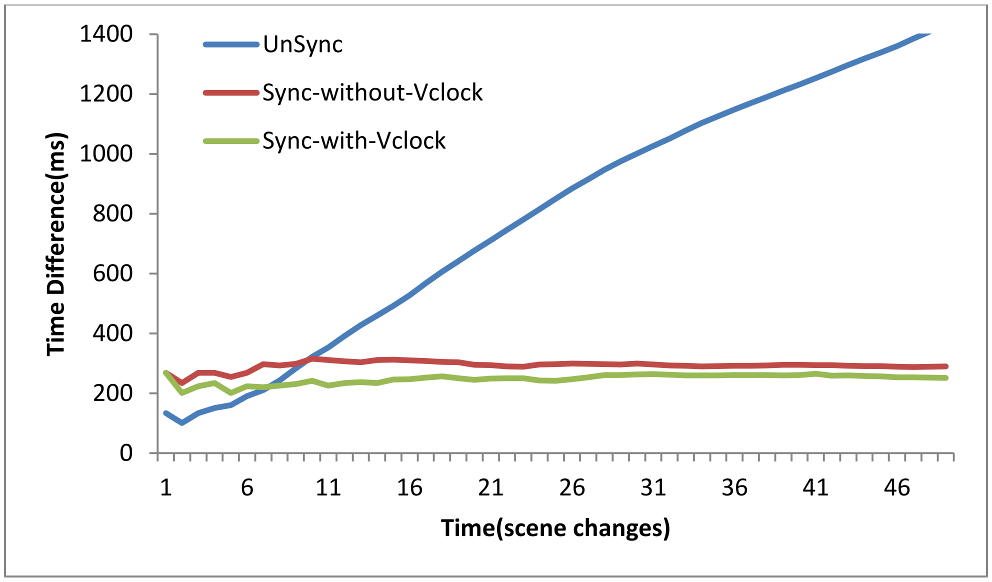


Figure 5.12 master slave with virtual clock (cumulative moving average)

5.2.4. Event synchronization

We performed experiment for the event synchronization. Most of the pause, resume and stop event were very well synchronized. The only interesting results were of the navigation to a specific segment of the presentation. In this case the navigation can be to a future location which can be the other node of the SMIL presentation. In this case to start the presentation of the new element may include processing the event, resolution of url associated with the node to navigate and then playing the presentation from the mentioned place. As a result of all the processing required to execute a navigation event some de-synchronization between the play-out were expected. In Figure 5.15 we presented the result of navigation event synchronization. The peaks represent higher degree of de-synchronization between play-out when navigation event occurred at one location. It is evident that the presentation were not synchronized momentarily when the event occurred at one destination but then synchronized again as a result of our synchronization mechanism.

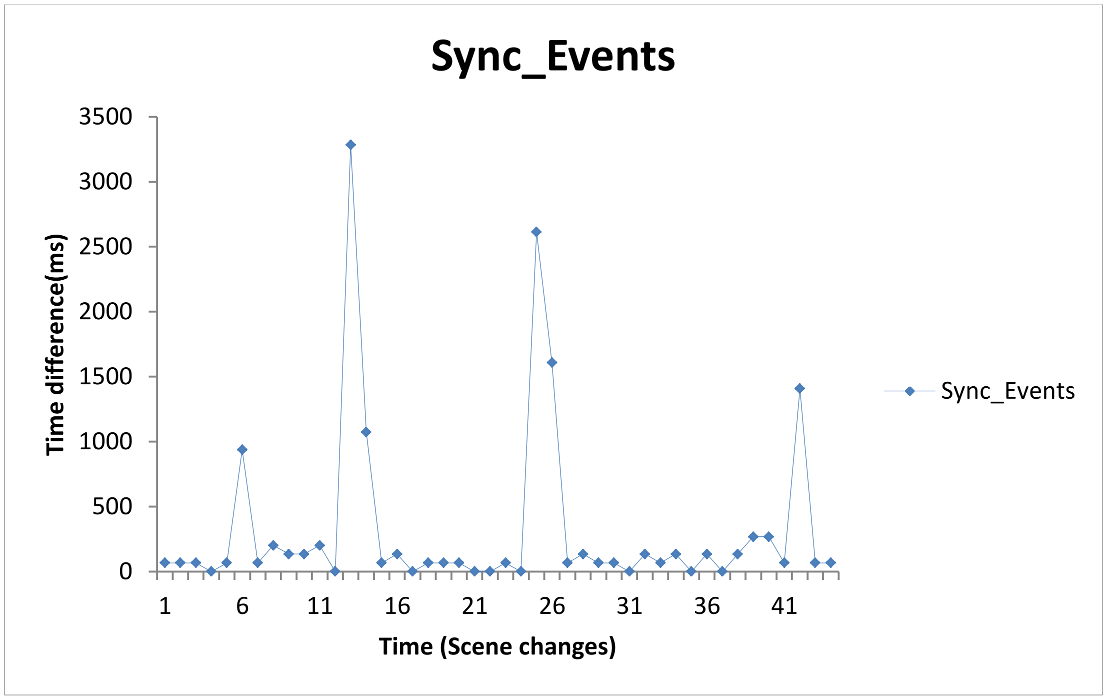


Figure 5.13 Event synchronization

Figure 5.13 presents only the play-out differences between distributed participants, but does not show how much time our synchronization mechanism took to resynchronize the play-out. Resynchronization period measurements are presented in Table 5.3. De-sync Time presents the time of event execution on one participant's presentation and the Re-sync Time presents time when the presentations are synchronized again after navigation event occurred at second participant's presentation. Re-sync period is the time interval when the presentations were not synchronized between participants. In half of navigation events the Re-sync period is 1 second which is very much acceptable. The maximum Re-sync period is 3 second which is on higher side. We have not done any user studies in this thesis but we are optimistic that 3 second R-sync period will be acceptable. The reason is that in case of navigation event execution users are already expecting some delay.

Event No	De-sync Time (Seconds)	Re-sync Time (Seconds)	Re-sync Period (Second)
1	48	49	1
2	84	87	3
3	140	143	3
4	175	176	1

Table 5.3 Measurements of resynchronization period of navigation events

Chapter 6:

Conclusion

Synchronous shared experience has been studied in recent past in perspective of stream-based media. Inter-destination synchronization mechanism is fundamental requirement of synchronous shared experience. Inter-destination synchronization in stream-based media has not been applied to work with the group of participants which varied available bandwidth and computing resources. Document- based media provides functionality of alternative content selection on bases of available resource. This makes document-based media a potential solution for inter-destination synchronization when participants of synchronous shared experience have varied bandwidth and computing resource.

This thesis explored certain technical challenges to inter-destination synchronization in document-based media. Given a varied bandwidth and computing resources of participants of a group for synchronous shared experience, media components need to be played-out in synchronous manner across participants. This thesis investigated techniques for inter-destination synchronization for stream-based media and adopted them to work with document based media. Firstly we designed a scalable and efficient signaling architecture for exchange on necessary information between participants. We did a prototype implementation of designed architecture. Such architecture was required for implementation and evaluation of inter-destination synchronization techniques for document-based media.

Secondly we studied existing inter-destination synchronization techniques for stream-based media in literature. We identified classes of techniques and adopted two of them to work with document-based media. We implemented and evaluated two classes of techniques in this thesis. In order to make synchronous shared experience interactive, action of the participants need to be synchronized across participants. These actions are user interactions which can change state of presentation. Finally we extended inter-destination synchronization techniques to synchronize user interaction actions across participants.

Total of three research questions were identified and address in this thesis work. In next section we restate and summarize results we achieve while answering these question.

Research Question 1.1: *Can we create a platform which facilitates the implementation and evaluation of existing stream-based media synchronization techniques for document-based media presentations? Which techniques are relevant for synchronization of document-based media presentation?*

To achieve synchronous shared experience in a group where participants have varied bandwidth and computing resources inter-destination mechanism was required. For inter-destination in document-based media we required a mechanism to play-out media objects at participant location. Instead of building the play-out engine by our self we used ambulant player to play-out document-based media. ambulant implements functionality of SMIL, which provides inter-media synchronization mechanism for different media objects in document-based media. It also provides the functionality of alternative content selection based on available bandwidth on participant's computer.

We extended ambulant implementation to provide inter-destination synchronization. For inter-destination synchronization in document-based media we required three components. The first one was play-out document-based media. ambulant provides this functionality. The second one was required to exchange signaling information between the participants, necessary for synchronization. We designed and implemented this component in ambulant and at moment it works well for five participants. To scale and evaluate it for higher no of participants is one of the future directions of this this thesis work. The third component was to control the play-out of media at participant's location in ambulant player in order to make it synchronous with other participants. We extended ambulant implementation to make play-out synchronous if the ambulant identified any play-out differences among participants. Identification of play-out differences was made possible by exchange of signaling information among participants. Above stated three components made it possible to design and implement architecture to support inter-destination synchronization in document-based media.

Research Question 1.2: *How much synchronization can existing stream-based media synchronization techniques achieve in cases of document-based media presentation?*

The acceptable play out difference in synchronous shared experience is not studied much in the literature, but the lower bound of the acceptable play out difference is 200 milliseconds. This value is taken from lip synchronization research [1]. In some literature[42] acceptable play out difference for synchronous shared experience is 500 milliseconds.

We implemented two basic algorithms for inter destination synchronization techniques, namely Master-Slave Technique and Synchronization-Manager Technique. For both techniques the play-out differences are within acceptable range. In general synchronization-manager technique performed better than master-slave technique. The performance of the master-slave technique with fast system as master and slow as slave was considerably below as compare to other scenarios.

We implemented a light weight virtual clock synchronization mechanism to synchronize the participants with master or synchronization-manager in case of master-slave and synchronization-manager technique respectively. Master-slave technique with fast master, whose performance was below than the other techniques, was improved as a result of this virtual clock synchronization of master and slave participants. Other techniques were not affected by this virtual clock synchronization mechanism.

We evaluate the system by processing the videos which recorded the play-out differences between participants. It is possible that the quality of the presentation may decrease as a result of our synchronization algorithms. Investigation of such effects was out of the scope of this thesis and is a very useful future direction of this research work. Subjective evaluation by involvement of the end users is also a future direction of this work.

Alternative content selection mechanism in SMIL is well tested functionality. So we did not focus how our system will work in varied bandwidth for selection of the alternative contents. Rather we focused more on providing inter-destination synchronization mechanism in document based media, which is necessary for synchronous share experience. To test and measure the performance of our system in varied bandwidth in context of alternative content selection can be interesting future direction of this work.

Research Question 1.3: *How much synchronization can we achieve in case of user interactions in document based media presentation? How much is the presentation at different destinations de-synchronized in case of user interaction and how long will it take to re-synchronize them?*

We implemented the event synchronization mechanism among distributed participants. These user generated events include , ’play’, ‘pause’, ‘stop’, ‘resume’ and ‘navigation control’. During the execution of user interaction the presentations across distributed participants remained well synchronized.

During execution of navigation event which causes skipping of presentation, presentation across distributed participants desynchronized momentarily but resynchronized again by our synchronization mechanism. This de-synchronization period lies between 1 second and 3 second. De-synchronization period of 3 seconds is on higher side but as user normally expects some delay in navigation event so we are optimistic that it will be accepted by most of users especially when they are at distributed location.

Subjective evaluation by user experience can validate these results, but it was out of scope of this thesis. However we consider subjective evaluation by user experience as useful future direction of this thesis work.

References

[1] Kouvelas, I. and Hardman, V. and Watson, A. Lip synchronisation for use over the Internet: analysis and implementation Global Telecommunications Conference, 1996. GLOBECOM '96. 'Communications: 1996 2 893 -898 vol.2

[2] E. Cronin, B. Filstrup, S. Jamin, A.R. Kurc, An efficient synchroniza- tion mechanism for mirrored game architectures, Multimedia Tools Appl. 23 (l) (2004) 7–30.

[3] C. Diot, L. Gautier, A distributed architecture for multiplayer interactive applications on the Internet, IEEE Network 13 (4) (1999) 6–15

[4] C.M. Huang, C. Wang, J.M. Hsu, Formal modeling and design of multimedia synchronization for interactive multimedia presenta- tions in distributed environments, in: International Conference on Consumer Electronics 1998, ICCE 1998, Digest of Technical Papers, June 1998, pp. 458–459.

[5] Y. Ishibashi, S. Tasaka, H. Miyamoto, Joint synchronization between stored media with interactive control and live media in multicast communications, IEICE Trans. Commun. E85-B (4) (2002) 812–822.

[6] C.M. Huang, C. Wang, C.H. Lin, Interactive multimedia synchroni- zation in the distributed environment using the formal approach, IEE Proc. Soft. 147 (4) (2000) 131–146.

[7] Narayanan Shivakumar, Cormac J. Sreenan, B. Narendran, and Prathima Agrawal, “The concord algorithm for synchronization of networked multimedia streams,” in international Conference on Multimedia Computing and Systems, 1995.

[8] C.J. Sreenan, Jyh-Cheng Chen, P Agrawal, and B Naendran, “Delay reduction techniques for playout buffering,” IEEE Transactions on Multimedia, vol. 2, no. 2, June 2000.

[9] Ramachandran Ramjee, Jim Kurose, Don Towsley, and Henning Schulzrinne, “Adaptive playout mechanisms for packetized audio applications in wide-area networks,” in Proceedings of the Conference on Computer Communications (IEEE Infocom), Toronto, Canada, June 1994, pp. 680–688, IEEE Computer Society Press, Los Alamitos, California.

[10] Van Jacobson and Michael J. Karels, “Congestion avoidance and control,” in SIGCOMM Symposium on Communications Architectures and Protocols. ACM, Nov. 1998.

[11] Jean-Chrysostome Bolot, “End-to-end packet delay and loss behavior in the Internet,” in SIGCOMM Symposium on Communications Architectures and Protocols, Deepinder P. Sidhu, Ed., San Francisco, California, Sept. 1993, ACM, pp. 289–298, also in Computer Communication Review 23 (4), Oct. 1992.

[12] Sue B. Moon, Jim Kurose, and Don Towsley, “Packet audio playout delay adjustment: performance bounds and algorithms,” ACM/Springer Multimedia Systems, vol. 5, no. 1, pp. 17–28, Jan. 1998.

[13] Marco Roccetti, Vittorio Ghini, Giovanni Pau, Paola Salomoni, and Maria Elena Bonfigli, “Design and experimental evaluation of an adaptive playout delay control mechanism for packetized audio for use over the internet,” Multimedia Tools and Applications, vol. 14, no. 1, May 2001.

[14] Felipe Alvarez-Cuevas, Miquel Bertran, Francesc Oller, and Josep M. Selga, “Voice synchronization in packet switching networks,” IEEE Network, vol. 7, no. 5, pp. 20–25, Sept. 1993.

[15] Nikolaos Laoutaris and Ioannis Stavrakakis, “An analytical design of optimal playout schedulers for packet video receivers,” submitted for publication in Computer Communications journal. An earlier version was presented at the 2nd International Workshop on Quality of Future Internet

Services (QofIS2001), Coimbra, Portugal, 2001, 2002.

[16] Donald L. Stone and Kevin Jeffay, “An empirical study of a jitter management scheme for video teleconferencing,” Multimedia Systems, vol. 2, no. 2, 1995.

[17] Kurt Rothermel and Tobias Helbig, “An adaptive stream synchronization protocol,” in Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Durham, New Hampshire, Apr. 1995, Lecture Notes in Computer Science, pp. 189–202, Springer.

[18] Ernst Biersack, Werner Geyer, and Christoph Bernhardt, “Intra and interstream synchronisation for stored multimedia streams,” in ICMCS (IEEE Multimedia Conference), Hiroshima, Japan, June 1996.

[19] Maria C. Yuang, Shih T. Liang, Yu G. Chen, and Chi L. Shen, “Dynamic video playout smoothing method for multimedia applications,” in Proceedings of the IEEE International Conference on Communications (IEEE ICC), Dallas, Texas, June 1996, p. S44.

[20] Maria C. Yuang, Po L. Tien, and Shih T. Liang, “Intelligent video smoother for multimedia communications,” IEEE Journal on Selected Areas in Communications, vol. 15, no. 2, pp. 136–146, Feb. 1997.

[21] Y. Ishibashi, A. Tsuji, S. Tasaka, A group synchronization mechan- ism for stored media in multicast communications, in: Proceed- ings of the Sixth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), vol. 2, Kobe, Japan, April 1997, pp. 692–700

[22] Boronat F., Guerri J.C., and Lloret J., “An RTP/RTCP based approach for multimedia group and inter-stream synchronization”, Multimedia Tools and Applications Journal, Vol. 40 (2), 285-319, June 2008.

[23] Y. Ishibashi, S. Tasaka, A group synchronization mechanism for live media in multicast communications, in: Global Telecommunica- tions Conference, 1997 (IEEE GLOBECOM’ 97), November 1997, pp. 746–752.

[24] Boronat F., Lloret J., and García M., Multimedia group and interstream synchronization techniques: A comparative study. Inf. Syst. 34, 1, 108-131, March 2009.

[25] C. Diot, L. Gautier, A distributed architecture for multiplayer interactive applications on the Internet, IEEE Network 13 (4) (1999) 6–15.

[26] Y. Ishibashi, S. Tasaka, A distributed control scheme for causality and media synchronization in networked multimedia games, in: Proceedings of the 11th International Conference on Computer Communications and Networks, Miami, USA, October 2002, pp. 144–149.

[27] M. Mauve, J. Vogel, V. Hilt, W. Effelsberg, Local-lag and timewarp: providing consistency for replicated continuous applications, IEEE Trans. Multimedia 6 (l) (2004) 47–57.

[28] Y. Ishibashi, S. Tasaka, A comparative survey of synchronization algorithms for continuous media in network environments, in: Proceedings of the 25th IEEE Conference on Local Computer Networks, Tampa, FL, USA, November 2000, pp. 337–348.

[29] H. Liu, M. El Zarki, A synchronization control scheme for real-time streaming multimedia applications, in: Proceedings of the 13th Packet Video Workshop, Nantes, France, April 2003.

[30] L. Ehley, B. Furht, M. Ilyas, Evaluation of multimedia synchroniza- tion techniques, in: Proceedings of the International Conference Multimedia Computing and Systems, ICMCS 94, Boston, MA, USA, May 1994, pp. 514–519.

[31] M.J. Perez-Luque, T.D.C. Little, A temporal reference framework for multimedia synchronization,

IEEE J. Sel. Areas Commun. 14 (1) (1996) 36–51.

[32] Boronat, F.; Montagud, M.; Vidal, V.; , Master Selection Policies for Inter-destination Multimedia Synchronization in Distributed Applications, in Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2011 IEEE 19th International Symposium on. 269 – 277

[33] Köhler, D. & Müller, H. Steinmetz R. (Ed.) Multimedia playout synchronization using buffer level control
Multimedia: Advanced Teleservices and High-Speed Communication Architectures, Springer Berlin / Heidelberg, 1994, 868, 167-180

[34] J. Eidson, M. Fischer, and J. White. IEEE 1588 standard for a precision clock synchronization protocol for networked measurement and control systems. In34 th Annual Precise Time and Time Interval (PTTI) Meeting, pages 243–254,2002.

[35] D. Mills. Network Time Protocol (Version 4) Specification, Implementation and Analysis. Network, 2006.

[36] M. Mauve, J. Vogel, V. Hilt, and W. Effelsberg. Local-lag and timewarp: Providing consistency for replicated continuous applications. IEEE Transactions on Multimedia, 6(1):47, 2004.

[37] Y. Zhang, L. Chen, and G. Chen. Globally synchronized dead-reckoning with local lag for continuous distributed multiplayer games. In NetGames ’06. ACM, 2006.

[38] R.N. Mekuria, H.M. Stokking, M.O. van Deventer. Automatic Measurement of Play-out Differences for Social TV, InteractiveTV, Gaming and Inter-destination Synchronization.*Adjunct proceedings of the EuroITV 2011*, 2011.

[39] Y. Liu, P. Shafton, D. A. Shamma, J. Yang, Zync: the design of synchronized video sharing, in Proceedings of the 2007 conference on Designing for User experiences, 2007.

[40] http://ambulantplayer.org/

[41] Dick C.A. Bulterman and Lloyd Rutledge; SMIL 3.0: Interactive Multimedia for the Web, Mobile Devices and Daisy Talking Books, published by Springer-Verlag, Heidelberg, December 2008

[42] Ishan Vaishnavi, Coherence in synchronous shared experience, PhD dissertation, 2011

[43] S.Ud din, D.C.A Bulterman, Synchronization Techniques in Distributed Multimedia Presentation, published in MMEDIA 2012, The Fourth International Conferences on Advances in Multimedia, 2012.

[44] Dick C.A. Bulterman, Jack Jansen, Kleanthis Kleanthous, Kees Blom and Daniel Benden; ambulant: A Fast, Multi-Platform Open Source SMIL Player, International Multimedia Conference: Proceedings of the 12 th annual ACM international conference on Multimedia, 10(6), p.g. 492-495, 2004.

[45] S.Ud din, D.C.A Bulterman, Synchronization Techniques in Distributed Multimedia Presentation, published in MMEDIA 2012, The Fourth International Conferences on Advances in Multimedia, 2012.

[46] http://www.virtualdub.org/

[47] Cesar, P., Vaishnavi, I., Kernchen, R., Meissner, S., Hesselman, C., Boussard, M., Spedalieri, A.,Bulterman, D.C.A., Gao, B.: Multimedia adaptation in ubiquitous environments: benefits of structured multimedia documents. In: Proceeding of the ACM Symposium on Document Engineering, pp. 275–284 (2008)