Managing synchronization and time factors in multimedia presentation

H Saiedian and M Awad

We propose a simple and efficient way to synchronize multimedia presentation. First, we introduce a conceptual multimedia system architecture in which synchronization actions are performed globally by a Global Synchronization Manager (GSM) which is part of the Multimedia Database Management System, and locally by Media Specific Synchronizers. Next, we propose a presentation scenario convention that is easy to create and manipulate. We assume that synchronization is driven by real time and not by synchronization events, and that coordination of scenario entries with Global Time (GT) is managed by the GSM. User interaction with the presentation creates entries in the GSM log of special events which in coordination with the relationship matrix will adjust the actual time included in the scenario entries. Entries in the presentation scenario contain presentation actions that are fired (activated) when the time field of that entry rendezvous with the GT.

multimedia data, system architecture, synchronization, time factor, interaction

BACKGROUND

Multimedia data are a combination of heterogeneous types of data such as text, tables, images (digitalized images and complex vector graphics), sound, voice, animation and video. Among these, only text and images are by themselves *time-independent* media, whereas sound, voice, animation and video are all *time-dependent*. By time-dependent and time-independent we mean whether or not the 'state' of the data presented depends on time. For example, an image or a text appearing on a display terminal is considered 'existentially' complete as soon as it appears on the terminal, while displaying a video clip or playing some voice is not considered complete until the recorded voice message or the clip reaches its end. Since voice data and video data interact in some way with time, and their complete existence depends on the passing of time, and furthermore, they vary with time, they are called *time-varying* data, while on the other hand, text and image data are *time-invariant*.

In order for a document appearing on the screen or a presentation using these different media data to be called a multimedia document or a multimedia presentation, two or more of these media must be involved. Such involvement will include some kind of interaction between these different media data[1]. This interaction is the 'soul' of multimedia applications in general. Interaction of this kind is divided into two types[2]:

(1) Sequential interaction.
(2) Overlapped interaction.

The first type, as the name indicates, has a *sequential* nature. An example of such an interaction could be a display of some text describing a historical site, then as the text display ends, a video clip showing that site is displayed. Another example is showing an image of a person followed by displaying his name next to his image, then playing a sample of his voice. This sequential interaction is simple and straightforward, as the start of any action is activated by the end of another, or in other words, the end of an action marks the beginning of another action. Building a model or an application for such an interaction is fairly simple. As a conceptual model for such interaction you need to define the concept of [action-start] (or [action-activation]) and another for [action-end] (or [action-deactivation]), and find a method to join such concepts to form a model for a sequential document (or application) presentation. Creating an application to apply such interaction is easy as most of the facilities used for creating real applications are sequential in nature.

The second type of interaction, i.e., *overlapped interaction*, is the one we are interested in. A few points have to be mentioned about this interaction:

- Overlapped interaction includes time-dependent media, but time-independent media might be included. The state of time-independent data does not change with time, but overlapped presentation can be done, for example, by displaying a text as soon as an image appears. Furthermore, when time-independent data interact with time-dependent data they might perform overlapped interaction. For example, while a video clip is displayed on the screen, specific text showing names of persons or sites appearing throughout the video presentation must be displayed when those persons or sites appear. Such 'simultaneous appearance', or in other words 'instantaneous display', should be managed as multimedia applications depend thoroughly on such overlapped interaction.
- Overlapped interaction depends by its entirety on the idea of 'synchronization'[3], which is the method used to manage different aspects of media interaction. Synchronization ensures that actions performed by a system

Department of Computer Science, University of Nebraska at Omaha, Omaha, NE 68182-0243, USA

are done in a logically correct order so that these actions will result in the final job as intended[4,5].

- 'Synchronization' has something to do with time. In fact we can say that synchronization in concept means managing the time factor in the time-related components of an event. If we agree on this, then we will have some kind of formal definition of the problem we are facing in multimedia field. It is as follows: 'How to synchronize actions (or events) so that their final result will be the intended result.' To be specific, we need to find a model that if applied to a multimedia application will synchronize the overlapped interaction of its components.
- When we say 'synchronizing multimedia actions' we mean the application ('software') side of it and not the hardware aspect. Although hardware synchronization is essential and no actual application synchronization is done without it, this is a different area of research that could be included either as part of the operating system field or the parallel computer architecture field.

The rest of this paper is organized as follows: in the next section we will include a description of the multimedia system architecture. We will then introduce the presentation scenario conventions, and the Global Synchronization Manager (GSM) log of special events in the third section. In the fourth section we will explain how basic synchronization tasks are carried out, and how user interaction is handled. In the last section we conclude the paper with some remarks about future work.

A CONCEPTUAL MULTIMEDIA SYSTEM ARCHITECTURE

In the traditional database systems in which only structured data are stored, user manipulates the data stored in the physical databases through a Database Management System (DBMS) that coordinates user's interactions with the database. This may include producing either a response to the user's query, or blocking evaluation of the request because of locks being issued on the database resources. Notice that no synchronization actions are needed when feedback responses are performed simply because data are in no need of time synchronization. A typical database management system environment is shown in Figure 1.

For multimedia database systems, we propose a system architecture that takes timing synchronization into consideration, and allows better communications between the user, the Multimedia Database Management System (MMDBMS) and the different media databases. Users interact with the MMDBMS by sending requests that pass through the Multimedia User Interface (MMUI) which in turn transforms it to a form understandable by the MMDBMS. User requests are communicated directly to the MMUI either by clicking on an icon appearing in a multimedia application, or requesting execution of an already designed multimedia application that involves different media. MMUI transforms such a request into a message containing all necessary information to the MMDBMS. We will designate a part of the MMDBMS as Global Synchronization Manager (GSM). The GSM receives the message, analyses its contents, divides it into different categories, and sends the results of this analysis to Media Specific Synchronizers (MSS) according to the different categories. For example, information concerning the video part of the presentation is sent to the video data synchronizer.

Messages sent to each MSS are analysed and the exact information requested is determined. Each MSS is considered a communication interface that, as shown in Figure 2, deals in one direction with the GSM, and in the other direction with the specific media database. Nodes appearing at the lower end of Figure 2 showing Media$_1$, . . . , Media$_n$ may include, besides the different databases, some media specific devices. For example, voice synthesizers and voice recognition devices are connected to voice databases, and video-related devices and image recognition and interpretation devices are connected to video and image databases respectively. These devices, along with the multimedia database themselves, receive messages from the MSS asking for particular (intrinsic) data. Feedback information is sent back to the MSS showing the availability of the data and the locking status of the database (if any) and, most importantly, information about the timing duration of the data requested. MSS receives all such information, keeps the local part of it and sends global information necessary for synchronization to the GSM which is the place where

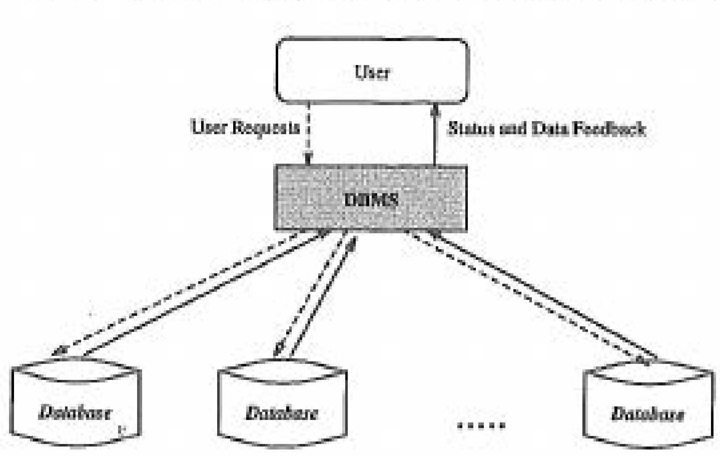

Figure 1. A conceptual database system model

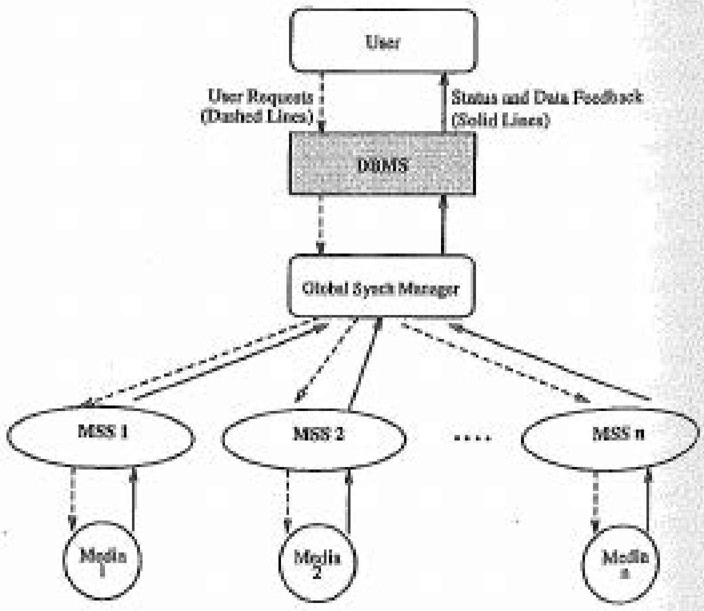

Figure 2. A conceptual multimedia synchronization model

all synchronization activities are managed. As a result, the GSM governs actual passage of multimedia data to the user according to the timing information received by each MSS, and according to user application needs transmitted through the MMUI by the user.

The four major aspects of this architecture that define the synchronization actions are:

(1) The GSM and the MSS.
(2) The messages transmitted through the architecture.
(3) The way these messages are interpreted when received.
(4) A timing model that all parts of the architecture agree upon.

These parts are explained and justified throughout the paper, but first we need to justify the need for two types of synchronization activities. In multimedia database systems, the user's request passes through the MMDBMS that sends it to the media involved. Before the user receives any feedback, two types of timing synchronization have to be done:

(1) Global synchronization,
(2) Media specific synchronization.

Global synchronization is needed to control feedback to the user and the overall multimedia presentation. The global synchronizer should determine the exact intended media interaction involved in each specific presentation. Data flow to the user application is done according to the messages received by the MMDBMS (via the GSM) from both directions. In other words, the GSM has to interpret the messages received from the user, and also determines the order of multimedia data flow to the user according to the messages received from the MSS.

In the overlapped media interaction the GSM has the following responsibilities:

(1) Managing the presentation scenario: this results in the actual feedback received by the user.
(2) Determination of the media involved in the user's request: this is done right after receiving the user's request. The GSM breaks the request into sub-requests. Each individual sub-request has two parts:
    (a) the medium involved, which is used as an address to direct processing information to the appropriate (MSS);
    (b) the processing information, which includes the address of the actual data needed, and the specific action to be performed on it whether to start, pause, resume or stop the flow of these data.

Media Specific Synchronizers receive the messages from the GSM and control the flow of data according to the contents of the messages.

According to our conceptual architecture, there is a global timer that is initialized when the presentation starts, and runs until the end of the presentation. This global time will be referred to as GT. It is managed and controlled by the GSM, and is consistent with the actual presentation introduced to the user, which means that the global timer will be initialized by the GSM when the presentation starts, and runs throughout the presentation incrementally even though overlapped interaction between different media parts might be performed, or even if parts of the presentation will pause for a period of time because of either the user interaction or for the GSM to perform some synchronization action. From the user's point of view, this is the only time that exists, and he coordinates his interaction with the presentation according to this global time. For each MSS, there is a local timer that is initialized when that media part of the presentation starts. This local time will be referred to as LT. A MSS performs the following actions on the local timer: (a) *Initialize;* (b) *Pause;* and (c) *Resume*. Initialization is done once when that part of the presentation is activated, and runs until the GSM sends a message to that specific MSS to stop his local timer temporarily which could happen in two cases:

(1) If the user asks for that explicitly by clicking on some icon to pause that part; or
(2) If the GSM needed that for global synchronization. An example of this will be a case when the GSM finds out that a user had asked for the video part of a presentation to pause, while there is a voice message associated with this video presentation. To avoid the loss of synchronization between the video display and the voice the GSM has to stop the act of playing the voice by sending a message to the Voice Synchronization Manager to stop his local time temporarily.

A Resume action on the local timer will be performed if the MSS received a message from the GSM to do so because the previous case has been resolved.

PRESENTATION SCENARIO AND GSM LOG

To describe the multimedia presentation in terms of individual specific events and to show the sequence and the time at which they overlap, we offer the following scenario specifications which, we think, is adequate to fulfil the timing synchronization requirements for the multimedia presentation according to our proposed system architecture.

Two or more different media parts of the presentation interact in a sequential or overlapped fashion will be described as a part of the scenario according to one of the following specifications:

(1) If the finish of presentation part P^i (e.g., video part) activates the beginning of part P^j (e.g., audio part) at time t_k then we form a tuple $\langle P^i_f, P^j_s, t_k \rangle$ where:

    - P^i_f: the finish f of presentation part i rendezvous
    - P^j_s: the start s of presentation part j at t_k.

    An example: $\langle V_f, A_s, t_k \rangle$.

(2) If part P^i starts while part P^j is active at time t_k, we form a tuple $\langle P^i_s, P^j_c, t_k \rangle$ where:

- P^i_s: the start s of presentation part (i) rendezvous
- P^j_c: the continuation c of presentation part j at t_k.

An example: $\langle V_s, A_c, t_k \rangle$

(3) If part P^i finishes (ends) while part P^j is active at time t_k, we form a tuple $\langle P^i_f, P^j_c, t_k \rangle$ where:

- P^i_f: the finish (f) of presentation part (i) rendezvous
- P^j_c: the continuation of presentation part (j) at t_k.

An example: $\langle V_f, A_c, t_k \rangle$

To model a specific presentation scenario Σ we form a set of tuples delimited by P_s and P_f referring to presentation start and presentation finish, e.g.,

$$\Sigma = [P_s, \langle V_s, t_1 \rangle, \langle A_s, V_c, t_2 \rangle, < T_s, A_c, V_c, t_3 \rangle,$$
$$\langle T_c, A_f, V_f, t_4 \rangle, \langle T_f, t_5 \rangle, P_f]$$

refers to a presentation session (Σ) in which the video part will start at t_1. While the video is in display, audio part starts at time t_2. Text part will be activated at time t_3 while the audio and video parts are still playing. Audio and video parts will end at time t_4. Text part will stay until time t_5 when it ends, which ends the presentation session. The GSM maintains a log of special events. These include:

(1) User stops the flow of some part of the presentation. This is indicated in the log by [PAUSE, PART, GT], where PART shows that part of the presentation, and GT shows the exact global time when that event happened. Example: [PAUSE, V, t_3].
(2) User resumes the flow of some paused part of the presentation. This is indicated in the Log by [RESUME, PART, GT], where PART shows that part of the presentation, and GT shows the exact global time when that event happened. Example: [RESUME, V, t_4].
(3) A presentation part has ended before its intended finish time. This happens when the user decides to end a part of the presentation before its previously planned finish time. This is indicated in the Log by [EARLY, PART, GT]. Example: [EARLY, V, t_5].

In addition to the log of special events, the GSM maintains a Relationship Matrix showing whether presentation part P^i should run in coordination with part P^j, or if they are separate. Clearly, if they relate to each other, any action on one of them will affect the other. The matrix shows 'r' if they are related, and 's' if they are separate (i.e., unrelated). See Figure 3.

BASIC SYNCHRONIZATION AND USER INTERACTIONS

Our conceptual multimedia system architecture will be able to handle the basic synchronization problems which define the regular multimedia data flow and media interaction synchronization. Also, it will handle the user interaction with the presentation, which includes:

(1) Pause a specific presentation part.

	P^1	P^2	...	P^n
P^1	–	r		r
P^2	r	–		s
⋮			...	
P^n	r	s		–

Figure 3. Relationship matrix

(2) Resume that presentation part.
(3) Close a presentation part earlier than planned.

Before explaining how these operations are done, we need to point out that according to our model:

- Synchronization depends on time, and time only.
- Synchronization points are defined by time coordinates.
- By time, we mean a universal time convention, so if we say that the video presentation part has been forwarded for 3 seconds or that the audio part has been rewound for 3 seconds we mean 3 seconds according to the user timing regardless of the nature of the medium involved. This timing convention is carried out by each MSS. The MSS will make sure that 3 seconds in real time means so and so in his medium's timing standards, and accordingly work on his data. For example, fast forward t seconds of a video part means skip $t \times n$ frames where n is number of frames displayed per second. Again, all this information is kept, and carried out in each MSS according to his medium.

Synchronization of the overall presentation is done using the scenario information, the GT and the LTs as described below:

(1) The GSM creates a copy of the scenario to be used in the current presentation. So the word scenario from now on will refer to this copy.
(2) At the beginning of the presentation session the GSM encounters the P_s entry in the scenario, which is a signal for him to initialize his GT to zero, and start global synchronization duties.
(3) If the time portion of an entry in the scenario rendezvous with the GT, the GSM checks first with his log. If the log is empty, or the presentation parts included in that entry do not relate to the ones in the log, the GSM sends messages to the corresponding MSSs according to the following:
    (a) If the entry contains P^i_s, the GSM sends a message to the corresponding MSS showing the data needed

(P^i), and asking him to initialize his LT to zero to start the flow of data.

(b) The GSM skips any P^i_c in that entry.

(c) If the entry contains P^i_f, the GSM sends a message to the corresponding MSS showing the part indicated and asking him to stop the flow of data.

(4) A request from the user to pause part P^i of the presentation will create an entry in the log of the GSM [PAUSE, P^i, t_k] and will drive the GSM to check the relationship matrix. The GSM will create entries [PAUSE, P^j, t_k] for all j where P^i and P^j are related. Then the GSM will send messages for all corresponding MSSs to stop their LTs, and to pause the flow of data. The MMUI will be responsible for showing the last image of any video part or animation sequence, and to quit any voice, or sound effect, if included.

(5) A request from the user to resume part P^i of the presentation will create an entry in the log of the GSM [RESUME, P^i, t_l], and will force the GSM to do the following:

(a) Search the log of special events for any [PAUSE, P^i, t_k] entry. If one is found the GSM will find the time difference $d_x = t_l - t_k$.

(b) Search the relationship matrix for the parts related to P^i, and add entries [PAUSE, P^j, t_l] for all such parts.

(c) Go through his copy of the scenario, and for each entry whose time field is greater than or equal to t_l, and has either P^j_s or P^j_f for all P^j related to P^i, adjust the time field in that entry by adding d_x to it (of course this includes P^i itself). The GSM then sends a message to the MSSs involved to resume the flow of data.

- If an entry in the scenario has its time field adjusted, and one of its constituents was an unrelated part P^t then if this entry contained P^t_s the GSM will look for the entry containing P^t_f and adjust its time, and vice versa.
- If this part is already active, and its finish time has been adjusted, the MMUI will make sure that the last section of that presentation keeps appearing to the user until the GT rendezvous with its finish action time field.

(6) A request from the user to end part P^i of the presentation earlier than its intended finish time will force the GSM to create an entry in his log of special events showing [EARLY, P^i, GT], then look into the relationship matrix, and create a similar entry for each related presentation part. Next, the GSM will change the time field for every entry in the scenario which has P^j_s or P^j_f for all P^j related to P^i, and for which the time field contents are greater than or equal to the GT, it will change it to zero.

If such an entry includes the start or finish action of an unrelated part then a new entry in the scenario is created which contains that unrelated part with the time field included in the original entry.

CONCLUSIONS AND FUTURE WORK

We have proposed a conceptual multimedia system architecture, a presentation scenario convention, and explained how these part with coordination with the GSM log of special events, and the relationship matrix will enforce multimedia presentation synchronization. Actions in the entries of the presentation scenario are activated when a rendezvous between GT managed by the GSM, and the time field in that entry is encountered. The user's interaction activities are managed by controlling changes in the time field of the entries containing related actions.

The proposed architecture solves the synchronization problem when the user needs to pause, resume or close a part of the presentation. An extension of this architecture will be to provide support for further complicated actions like fast forward or rewind voice and video parts, and a future intended work plan will focus on letting the user create his own multimedia presentation interactively, and the ability of the user to add additional presentation parts to an already designed one during the presentation. At the end of this step, an object oriented framework will be created according to an object-oriented view of the architecture.

REFERENCES

1 **Little, T and Ghafoor, A** 'Synchronization and storage models for multimedia objects' *IEEE J. Selected Areas in Communications* Vol 8 No 3, (April 1990) pp 413–427

2 **Hoepner, P** 'Synchronizing the presentation of multimedia objects – ODA extensions' in **Kjelldahl, L (ed)** *Multimedia: systems, interaction and applications* 1st Eurographics Workshop, Stockholm, Sweden (April 1991) pp 87–100.

3 **Gibbs, S, Dami, L and Tsichritzis, D** 'An object-oriented framework for multimedia composition and synchronization' in **Kjelldahl, L (ed)** *Multimedia: systems, interaction and applications* 1st Eurographics Workshop, Stockholm, Sweden (April 1991) pp 102–111

4 **Steinmetz, R** 'Synchronization properties in multimedia systems' *IEEE J. Selected Areas in Communications* Vol 8 No 3 (April 1990) pp 401–412

5 **Gibbs, S, Breiteneder, C, Dami, L, De Mey, V and Tsichritzis, D** 'A programming environment for multimedia applications' in **Tsichritzis, D (ed)** *Object Framework* University of Geneve (1992)