Logical Mapping: An Intermedia Synchronization Model for Multimedia Distributed Systems

Saul E. Pomares Hernandez, Luis A. Morales Rosales,
Jorge Estudillo Ramirez and Gustavo Rodriguez Gomez
National Institute of Astrophysics, Optics and Electronics (INAOE)
Luis Enrique Erro No. 1, Zip 72840, Tonantzintla, Puebla, Mexico
Email: {spomares, lamorales, jestudillo, grodrig}@inaoep.mx

***Abstract*— The preservation of temporal dependencies among different media data, such as text, still images, video and audio, and which have simultaneous distributed sources as origin, is an open research area and an important issue for emerging distributed multimedia systems, such as Teleimmersion, Telemedicine, and IPTV. Although there are several works oriented to satisfy temporal dependencies in distributed multimedia systems, they are far from resolving the problem. In this paper we propose a logical synchronization model able to specify at runtime any kind of temporal relationship among the distributed multimedia data involved in a temporal scenario. The synchronization model is based on a new concept that we call *logical mapping*. A logical mapping, in general terms, translates a temporal relation based on a timeline to be expressed according to its causal dependencies. The logical mappings allow us to avoid the use of global references, such as a wall clock and shared memory. We note that the proposed intermedia synchronization model does not require previous knowledge of when, nor of how long, the media involved of a temporal scenario is executed. Finally, in order to show the viability of the proposed model, a syncrhonization approach is presented.**

***Index Terms*— distributed systems, temporal synchronization, causal ordering, group communication.**

I. Introduction

Emerging distributed multimedia systems, such as Teleimmersion, Telemedicine, and IPTV deal with heterogeneous data, which is generally grouped into two broad categories: *continuous media* (e.g. audio and video) and *discrete media* (e.g. texts, data and images). One open research area in multimedia distributed systems involves intermedia synchronization. Intermedia synchronization concerns the preservation of temporal dependencies among the application data from the time of generation to the time of presentation. For example, in a one-to-many distance learning scenario, a professor can use a Whiteboard tool to send slides in still images (discrete media) and two continuous medias (e.g. audio and video) to give details about them. In this scenario, the intermedia synchronization must ensure that at any reception process ($Client$) (Figure 1), the slides must be presented according to the phrases pronounced with their corresponding video. A simple on-site education scenario such as "As we can see in the next figure"(associated with the correspondent concurrent presentation slide action) is not simple to reproduce on a distance learning system since the communication channels are asynchronous and independent, and the media involved is heterogeneous [1]. The term $next$ in the scenario establishes, at an application level, a temporal dependency with the remaining media involved. In such case, if temporal dependencies are not ensured, it is possible that the receiver can associate, without distinction, this phrase to the current, the preceding or the following slide. Satisfying temporal dependencies is even more complex if we consider many-to-many multimedia scenarios, where geographically distributed participants communicate simultaneously. The present work mainly focuses on this last kind of scenario, where in principle, there is no global reference, such as a wall clock or shared memory.

Several works attempt to give a possible solution to the intermedia synchronization [2]–[6]; however, as we will show, these works are far from resolving the problem. In this paper we propose a logical synchronization model that specifies at runtime any kind of temporal relationship among the multimedia data involved: interval-interval relations for continuous media, point-to-point relations for discrete media, and point-interval relations when discrete and continuous media coexist.

In order to specify distributed temporal scenarios that can include the three types of relations, our model considers two abstract levels. At the higher level, the multimedia data (discrete and continuous) is represented as $intervals$. At the lower level, the model considers that an interval is a set of finite atomic sequential elements, called information units (we refer to them in the rest of the paper simply as $messages$). Our model translates temporal multimedia scenarios to be expressed as segments (intervals) arranged according to the precedence and/or simultaneous interval relations that we define (Definitions 3 and 4). This translation results in the creation of what we call a *logical mapping*. In terms of intermedia synchronization, a logical mapping determines "when" a reception process should reproduce the media data.

At the end of this paper, we present an approach to carry out the synchronization model. The main objectives of the approach is to present a manner of how the model can be implemented and to show its viability.

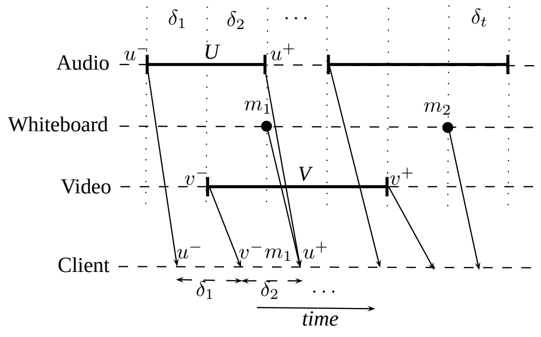


Figure 1. A multimedia scenario example

This paper is structured as follows. Section 2 presents the most relevant related work. Next, in Section 3, the system model is described and the background information is provided. In Section 4, the intermedia synchronization model is presented. In Section 5, a synchronization approach based on this model is presented. Some conclusions are given in Section 7.

II. Related Work

Many works related to multimedia synchronization exist. We classify them according to the mode of execution: offline mode and inline mode. The offline mode refers to when the specification of a temporal scenario is manually made, and it preceds the media data transmision. The most representative works in this category are: the object composition Petri net model (OCPN) introduced by Little and Gafford [7], the timing and synchronization model of SMIL [2], and the time ontology model of OWL [4]. The main characteristic of the works in this category is that they require complete previous knowledge of the media data, the temporal dependencies and/or the order of actions before carrying out the temporal specification.

On the other hand, we have the inline mode. In this mode, the temporal specification is carried out at runtime. Most of the works in this category are primarily based on the identification and preservation of physical time constraints by using a common reference (wall clock, shared memory, mixer, etc.) [1], [3]. These works usually try to answer the synchronization problem by measuring and ensuring, based on a timeline, the period of physical or virtual time elapsed (δ_t) between certain points. Such points can be, for example, the *begin* (x^-), *end* (x^+) and/or discrete events (m_1) of the media involved (Fig. 1). In this type of works, the time of presentation δ_t among events must be ensured at any Client, .

Few works address intermedia synchronization at runtime without the use of a global reference, as is the case of the present work. Avoiding global references is desirable when the media data have different sources and the transmission delay is not negligible. These kind of works are primarily based on the identification of logical dependencies. There are two main approaches based on logical dependencies: Shimamura's model [5] and the work of Plesca et al. [6].

Shimamura et al. in [5] establish six logical precedence relations at an object level (*top, tail, full, partial, inclusive* and *exclusive)*. These relations are specified based on the causal dependencies of the *begin* (v^-) and *end* (v^+) points of the objects. The objects are represented by intervals composed by messages. In order to obtain a fine level synchronization, Shimamura introduces an interval segmentation mechanism that arbitrarily divides the objects into predetermined fixed length segments. This mechanism uses two logical relations: the *precedes* relation and the *concurrent* interval relation. The *precedes* relation of Shimamura is defined as:

$$U \rightarrow V \text{ if } u^+ \rightarrow v^-$$

while the *concurrent* relation is defined as:

$$U \parallel V \text{ if } \neg(u^+ \rightarrow v^-) \wedge \neg(v^+ \rightarrow u^-)$$

We note that this mechanism can be inaccurate since it does not clearly establish, at a segment level, a translation of the possible timeline temporal interval relations identified by Allen [8] (*before, meets, overlaps, starts, finishes, includes, equals*). Shimamura's work, as a consequence of an arbitrary segmentation and a broad definition of the concurrent interval relation, determines six of Allen's seven basic relations as "concurrent"(see Table I). A pair of concurrently related segments (intervals) means that no order can be established between the messages that compose them. In other words, it is not possible to clearly determine when the media data must be executed.

TABLE I.
Allen's relations with their corresponding Shimamura'relations

Allen's Relations	Shimamura's Relations
$UbeforeV$	$UprecedesV$
$UequalsV$	$UconcurrentV$
$UmeetsV$	
$UoverlapsV$	
$UduringV$	
$UstartsV$	
$UfinishesV$	

Plesca et al. [6] have considered a practical approach for intermedia synchronization by using causal dependencies. Plesca's work uses causal messages as synchronization points to satisfy temporal dependencies among continuous media. This work, in a heuristic manner, introduces causal synchronization points and shows that these points can be useful, but it does not resolve the problem of when causal messages must be introduced.

III. Preliminaries

A. The System Model

Processes: The application under consideration is composed of a set of processes $P = \{i, j, \ldots\}$ organized into a group that communicates by reliable broadcast asynchronous message passing. A process can only send one message at a time.

Messages: We consider a finite set of messages M, where each message $m \in M$ is identified by a tuple $m = (p, x)$, where $p \in P$ is the sender of m, and x is the local logical clock for messages of p, when m is broadcasted. The set of destinations of a message m is always P.

Events: Let m be a message. We denote by $send(m)$ the emission event and by $delivery(p,m)$ the delivery event of m to participant $p \in P$. The set of events associated to M is the set $E = \{send(m) : m \in M\} \cup \{delivery(p,m) : m \in M \wedge p \in P\}$. The process $p(e)$ of an event $e \in E$ is defined by $p(send(m)) = p$ and $p(delivery(p,m)) = p$. The set of events of a process p is $E_p = \{e \in E : p(e) = p\}$.

Intervals: We consider a finite set I of intervals, where each interval $A \in I$ is a set of messages $A \subseteq M$ sent by participant $p = Part(A)$, defined by the mapping $Part : I \to P$. We denote by a^- and a^+ the endpoint messages of A, and due to the sequential order of $Part(A)$, we have that for all $m \in A : a^- \neq m$ and $a^+ \neq m$ implies that $a^- \to m \to a^+$. We note that when $|A| = 1$, we have that $a^- = a^+$; in this case, a^- and a^+ are denoted indistinctly by a.

B. Background and Definitions

Happened-Before Relation for Discrete Media. The Happened-Before relation is a strict partial order [9] (i.e. irreflexive, asymmetric and transitive) denoted by $e \to e'$ (i.e. e causally precedes e') defined as follows:

Definition 1: The causal relation "$\to$" is the least partial order relation on E satisfying the two following properties:

1) For each participant p, the set of events E_p involving p is totally ordered: $e, e' \in E_p \Rightarrow e \to e' \vee e' \to e$
2) For each message m and destination $p \in P$ of m, the emission of m precedes its delivery; i.e. $send(m) \to delivery(p,m)$

By using "$\to$", Lamport defines that two events are concurrent as follows:

$$e \parallel e' \text{ if } \neg(e \to e' \vee e' \to e)$$

Partial Causal Relation (PCR). The PCR relation was introduced in [10] (Definition 2). It considers a subset $M' \subseteq M$ of messages. The PCR induced by M' takes into account the subset of events $E' \subseteq E$ that refer to *send* or *delivery* events of the messages belonging to M'. In our work, the PCR relation is a non strict partial order (i.e. *reflexive*, *asymmetric*, and *transitive*).

Definition 2: The partial causal relation "$\to_{E'}$" is the least partial order relation satisfying the two following properties:

1) For each participant $p \in P$, the local restrictions of $\to_{E'}$ and $\to$ to the events of E'_p coincide: $\forall e, e' \in E'_p : e \to e' \Leftrightarrow e \to_{E'} e'$
2) For each message $m \in M'$and $j \in P$, the emission of m precedes its delivery to $j : j \in P \Rightarrow send(m) \to_{E'} delivery(j,m)$

Happened-Before Relation for Intervals. Lamport establishes in [11] that an interval A happens before another interval B if all elements that compose interval A causally precede all elements of interval B. This definition is used in the model presented in Section 4. However, according to the definition of $Intervals$ presented in Section 3.1, the causal interval relation "$\to_I$" can be expressed only in terms of the endpoints as follows:

Definition 3: The relation "$\to_I$" is accomplished if the following two conditions are satisfied:

1) $A \to_I B$ if $a^+ \to_{E'} b^-$
2) $A \to_I B$ if $\exists C \mid (a^+ \to_{E'} c^- \wedge c^+ \to_{E'} b^-)$

where a^+ and b^- are the endpoints of A and B, respectively, c^- and c^+ are the endpoints of C, and $\to_{E'}$ is the partial causal order (Definition 2) induced on $E' \subseteq E$, where E', in this case, is the subset composed by the endpoint send events of the intervals in I. When $|A| = 1$, we have that $a^- = a^+ = a$; and in this case, we consider $a \to a$.

Finally, we present the simultaneous relation for intervals, defined as follows:

Definition 4: Two intervals, A and B, are said to be simultaneous " $|||$" if the following condition is satisfied:

- $A \,|||\, B \Rightarrow a^- \parallel b^- \wedge a^+ \parallel b^+$

The definition above means that one interval A can take place at the "same time" as another interval B.

IV. Intermedia Synchronization Model

According to Bertino et al. [12], a temporal synchronization model should possess three basic features: the ability to represent the relationships in a temporal scenario, the support for the generation of synchronization specifications, and finally the management of indeterminacy. In order to attain the first capability in distributed systems without using global references, we propose to specify temporal scenarios based on the identification of logical precedence dependencies among the media involved. Since we group the media data as continuous and discrete, our temporal scenarios are composed by interval-interval relations, point-point relations, and point-continuous relations. All possible temporal relations based on a timeline have been previously established by Allen [8](continuous-continuous) and Vilain [13] (point-point and point-continuous). Our logical model represents these temporal relations by translating them in terms of the precedence relation and simultaneous relation previously defined (Definitions 3 and 4); we call this translation *logical mapping*. Informally, the logical mapping decomposes a temporal relation into data segments (intervals) that are arranged according to their precedence dependencies. A detailed description is given in the next section.

A. Logical Mapping

The logical mapping translation (Table II) involves every pair of intervals of a temporal relation. Each interval is labeled as X or Y such that for every pair, $x^- \to y^-$ or $x^- \parallel y^-$. Once the X and Y intervals are identified, they are segmented into four subintervals [1]: $A(X,Y)$, $C(X,Y)$, $D(X,Y)$, and $B(X,Y)$. These data segments,

[1]We consider in our model that an interval can be empty. In such case, the following properties apply:
– $\emptyset \to_I A \vee A \to_I \emptyset = A$ and $\emptyset \,|||\, A \vee A \,|||\, \emptyset = A$

considering our definition, become new intervals[2]. Finally, we proceed to construct the general causal structure $S(X,Y) = A(X,Y) \rightarrow_I W(X,Y) \rightarrow_I B(X,Y)$, where $W(X,Y)$ determines if overlaps exist between the present pair. We note that the data segments can be constructed at runtime as the events occur in the system, based on the causal-effect relation of the endpoints.

TABLE II.
LOGICAL MAPPING

$\forall(X,Y)$	$\in$	$I \times I$
$A(X,Y)$	$\leftarrow$	- if $x^- \rightarrow y^-$, $\{x \in X : delivery(Part(Y),x) \rightarrow send(y^-)\}$ - otherwise, $\emptyset$
$C(X,Y)$	$\leftarrow$	- if $y^+ \rightarrow x^+$, $\{x \in X : send(x) \rightarrow delivery(Part(X),y^+)\} - A(X,Y)$ - otherwise, $X - A(X,Y)$
$D(X,Y)$	$\leftarrow$	- if $x^+ \rightarrow y^+$, $Y - \{y \in Y : delivery(Part(Y),x^+) \rightarrow send(y)\}$ - otherwise, Y
$B(X,Y)$	$\leftarrow$	- if $y^+ \rightarrow x^+, X - \{A(X,Y) \cup C(X,Y)\}$ - otherwise, $Y - D(X,Y)$
$W(X,Y)$	$\equiv$	$C(X,Y) \vert\vert\vert D(X,Y)$
$S(X,Y)$	$\equiv$	$A(X,Y) \rightarrow_I W(X,Y) \rightarrow_I B(X,Y)$

The model identifies five logical mappings: $precedes, overlaps, ends, starts$, and $simultaneous$. These five logical mappings, as we will show, are sufficient to represent all possible intermedia temporal relations. Notice that the resulting logical mappings represent synchronization specification units, which are automatic processable descriptions of a given temporal relation. For clarity, we next present the establishment of the logical mappings according to the type of temporal relation.

1) Logical Mapping for Continuous Media: Allen identifies seven basic relations and their inverses (See Table III) for atomic intervals [8]. These temporal relations establish all the possible ways that a pair of continuous media can be temporally related based on a global clock. In order to find a logical representation for continuous media temporal relations, we translate, according to Table II, Allen's basic relations and we identify five possible logical mappings (Table III, second column), which we call: *precedes, overlaps, ends, starts,* and *simultaneous*. These five logical mappings, as shown in Table III, are sufficient to represent all possible interval-interval relations. It is interesting to see in the right column of Table III that the logical mappings can be represented at a message level by the endpoints of the subintervals that compose them. This fact is important since for the construction of efficient synchronization mechanisms based on this model, they will only need to work with the precedence dependencies of the endpoints to ensure synchronization specifications.

2) Logical Mapping for Discrete Media: Vilain establishes three possible basic point-point temporal relations based on a timeline, which are *before* $(<)$, *simultaneous* $(=)$ and *after* $(>)$ (See Table IV, left column). After applying Table II to each possible temporal relation, we identify two logical mappings, which are *precedes* and *simultaneous* (See Table IV, right column). We can see that these logical mappings are the same as defined at an interval level for the continuous media. They differ in the representation endpoints. For example, when $x < y$ (*before* relation), we have $A = \{x\}$ and $B = \{y\}$, and therefore, we obtain the logical mapping $A \rightarrow_I B$. We have for this logical mapping an endpoint representation $a \rightarrow b$.

3) Logical Mapping for Discrete and Continuous Media: Vilain's point-interval algebra [13] establishes five basic temporal relations (See Table V, left column). We note that our model considers an additional temporal relation called the *simultaneous* relation. This relation is not considered by Vilain because by using a timeline, a single discrete element (a point) cannot occur with more than one element of an interval at the same time (See Table V, v *during* U point-interval relation). By translating each temporal relation, we establish five possible logical mappings (*precedes, overlaps, ends, starts,* and *simultaneous*). We can see in Table V that these logical mappings are sufficient to represent all possible point-interval temporal relations. It is interesting to note that by considering the continuous media and the discrete media as intervals, the model can indistinctly deal with both of them without a need for any particular considerations.

In the next section, we show how, by using the logical mappings, it is possible to make temporal synchronization specifications, which can be simple to maintain and to re-use.

B. Synchronization Specification

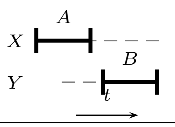
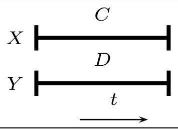
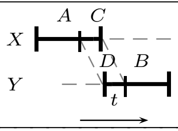
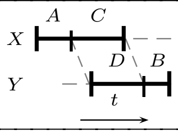
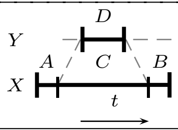
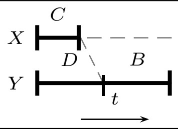
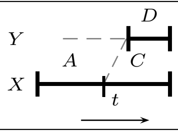
The second ability that a synchronization model should possess is the support for the generation of synchronization specifications. This ability concerns three aspects: specification maintainability, specification reusability and consistency checking. Our work generates a synchronization specification SS for a temporal scenario TS (set of intervals), expressed $SS(TS)$, by translating each pair of intervals in its correspondent logical mapping. For example, extracting from the distance learning scenario example previously presented, the stream of Video and the data sent by the Whiteboard, we have TS= $\{V,\{m_1\},\{m_2\}\}$. For this scenario, we find the following specification.

$$\begin{aligned} &SS(TS) = \\ &\{A(V,m_1) \rightarrow_I (C(V,m_1) \,|||\, D(V,m_1)) \rightarrow_I B(V,m_1), \\ &\qquad A(V,m_2) \rightarrow_I B(V,m_2), \\ &\qquad A(m_1,m_2) \rightarrow_I B(m_1,m_2)\} \end{aligned}$$

In order to explain how the logical mapping is performed at runtime, we take the pair $V,\{m_1\}$ depicted in Fig. 2. The process of creating a logical mapping in our work is made by identifying the causal boundaries of the concerned intervals from left to right as the time elapses and the events occur.

[2]Henceforth, we can refer to them only as A, C, D, and B when there is no ambiguity in the context

TABLE III.
ALLEN'S INTERVAL-INTERVAL RELATIONS AND THEIR LOGICAL MAPPING

Allen's Relations	**Logical Mapping**	**Scenario Example**	**Logical Mapping Expressed on Endpoints**
$UbeforeV$	*precedes*: $A \rightarrow_I B$		$a^+ \rightarrow b^-$
$UequalsV$	*simultaneous*: $(C \parallel\mid D)$		$c^- \| d^-, c^+ \| d^+$
$UmeetsV$	*overlaps*:		
$UoverlapsV$	$A \rightarrow_I (C \parallel\mid D) \rightarrow_I B$		$a^+ \rightarrow c^-, a^+ \rightarrow d^-$ $c^- \| d^-, c^+ \| d^+$ $c^+ \rightarrow b^-, d^+ \rightarrow b^-$
$UduringV$			
$UstartsV$	*starts*: $(C \parallel\mid D) \rightarrow_I B$		$c^- \| d^-, c^+ \| d^+$ $c^+ \rightarrow b^-, d^+ \rightarrow b^-$
$UfinishesV$	*ends*: $A \rightarrow_I (C \parallel\mid D)$		$a^+ \rightarrow c^-, a^+ \rightarrow d^-$ $c^- \| d^-, c^+ \| d^+$

In this example (See Figure 2), we first identify $X = V$ and $Y = \{m_1\}$ since $x^- \rightarrow y$. Then, we proceed to determine segment $A(V, m_1)$. The segment $A(V, m_1)$ will be constructed, according to Table II, for each element $x \in X$ such that $delivery(Part(Y), x) \rightarrow send(y)$). After interval segment $A(V, m_1)$ is identified, we proceed to identify the elements of the segments $C(V, m_1)$ and $D(V, m_1)$. In this case, since $y \rightarrow x^+$, segment $C(V, m_1)$ will be composed by each element $x \in X$ such that $send(x) \rightarrow delivery(Part(X), y^+)$ minus the elements of segment $A(V, m_1)$. Now, since $\neg(x^+ \rightarrow y^+)$ we have that $D(X, Y) = Y = \{m_1\}$. Finally, we establish the segment $B(V, m_1)$ as equal to $X - \{A(V, m_1) \cup C(V, m_1)\}$, since $y \rightarrow x^+$.

Interpreting the synchronization specification $S(V, m_1) = A(V, m_1) \rightarrow_I (C(V, m_1) \parallel\mid D(V, m_1)) \rightarrow_I B(V, m_1)$ for the pair $V, \{m_1\}$ (see Fig. 2) means the following: Interval $A(V, m_1)$ identifies the subset of messages $v \in V$ that causally precede the sending of discrete media m_1 and that should be reproduced before $D(V, m_1) = \{m_1\}$. Interval $C(V, m_1)$ identifies the messages that are concurrent to $D(V, m_1)$ and that can be reproduced in any order with respect to $D(V, m_1)$. Finally, interval $B(V, m_1)$ identifies the messages that should be executed after $D(V, m_1)$. In terms of physical time and their endpoints, the period of time between the execution of $a^+ \in A(V, m_1)$ and $b^- \in B(V, m_1)$ establishes the valid period of time for when the discrete media $d \in D(V, m_1)$ should be executed at any reception process.

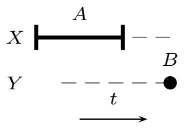
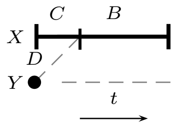
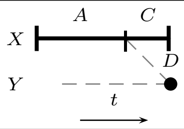
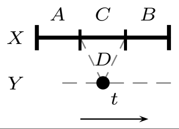
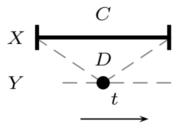
TABLE IV.
VILAIN'S POINT-POINT RELATIONS AND THEIR LOGICAL MAPPING

Basic Point Relations	**Logical Mapping**	**Logical Mapping Expressed on Endpoints**
$x < y$	*precedes*: $A \rightarrow_I B$	$a \rightarrow b$
$y > x$		
$x = y$	*simultaneous*: $C \parallel\mid D$	$c \parallel d$

For the aspect concerning the specification maintainability, we note that in our model, if an element (interval) is moved, we only need to redesign the part of the specification where that element modifies the precedence dependencies. For example, if we move m_1 such that $m_1 \rightarrow v^-$ the first specification line results in $A(V, m_1) \rightarrow_I B(V, m_1)$; nevertheless, the rest remains unchanged.

The reusability rends possible the use of an existing specification. Our model considers specification reusabil-

TABLE V.
VILAIN'S POINT-INTERVAL RELATIONS AND THEIR LOGICAL MAPPING

Vilain's Relations	**Logical Mapping**	**Scenario Example**	**Logical Mapping Expressed on Endpoints**
U *before* v	*precedes*: $A \rightarrow_I B$		$a^+ \rightarrow b$
v *after* U			
U *starts* v	*starts*: $(C \mid\mid\mid D) \rightarrow_I B$		$c^- \parallel d$, $c^+ \rightarrow b^-$ $d \rightarrow b^-$
U *finishes* v	*ends*: $A \rightarrow_I (C \mid\mid\mid D)$		$a^+ \rightarrow c^-$, $a^+ \rightarrow d$ $c^- \parallel d$
v *during* U	*overlaps*: $A \rightarrow_I (C \mid\mid\mid D) \rightarrow_I B$		$a^+ \rightarrow c^-$, $a^+ \rightarrow d$ $c^- \parallel d$, $c^+ \parallel d$ $c^+ \rightarrow b^-$, $d \rightarrow b^-$
Not defined in Vilain's relations U *simultaneous* v	*simultaneous*: $C \mid\mid\mid D$		$c^- \parallel d$, $c^+ \parallel d$

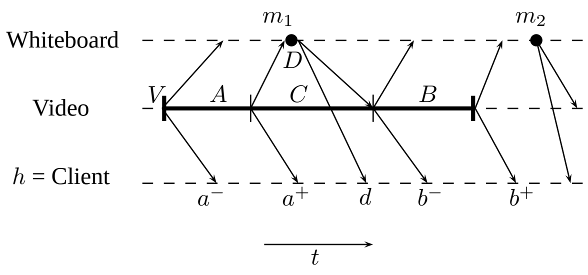


Figure 2. Temporal Scenario example

ity not only when new intervals are inserted in a temporal scenario, but also in the contrary case when some element(s) are removed. When new intervals are inserted, the specification is redesigned by adding the new precedence dependencies that each new element generates with the set of the intervals in the TS; but the rest of the specification remains unchanged. For example, if we insert the interval U of the Audio stream of Fig. 1 in the previous example, we need to modify $SS(TS)$ by adding the synchronization specification for the pairs of intervals $(U, m_1), (U, m_2)$ and (U, V), but we do not modify the rest. For the case when some element is removed, we simply erase the specification where such element belongs. With respect to the consistency checking, our work is based on ensuring that the set of intervals of a temporal scenario are locally reproduced by a reception process according to the precedence dependencies established by the temporal specification.

Finally, for the aspect of management of indeterminacy, we notice that our model does not need previous knowledge of the behavior of the system, i.e. we don't need to know in advance the composition of the temporal scenario. This model can be used for real-time cooperative systems since the precedence dependencies among the media involved can be established at an execution time according to the behavior of the system.

C. Logical Mapping vs Other Works

As we show, our five logical mappings are sufficient to logically represent any type of temporal relation. The logical mappings obtained constitute the expressive power of our intermedia synchronization model. As far as we know, this is the first inline distributed synchronization model that includes the three types of possible temporal relations: interval-interval relations, point-to-point relations, and point-interval relations. The works of Shimamura [5] and Plesca [6] only consider interval-interval relations. On the other hand, our model is more expressive than Shimamura's work for continuous media since it clearly identifies more logical dependencies among the media data, which results in a more accurate interval segmentation. With respect to Plesca's work, causal messages in the media continua are introduced in a heuristic manner. The logical mapping model, on the other hand, clearly establishes in a determinist way when the causal messages must be introduced.

V. Carrying out the synchronization model

Taking the resultant logical mapping for the pair $V, \{m_1\}$ depicted in Fig. 2 which is $A(V, m_1) \rightarrow_I (C(V, m_1) \mid\mid\mid D(V, m_1)) \rightarrow_I B(V, m_1)$, we proceed to show how it is possible to carry out the intermedia synchronization. To achieve this, the first task to perform is the interval causal delivery, and secondly the correction of the synchronization error.

Interval causal delivery. According to interval declaration and $Definition$ 3 presented in Section III, the interval causal delivery can be done by ensuring only the causal delivery of the interval endpoints. In this case, for a $h = Client$, we need to ensure that:

- *delivery*$(h, a^+) \rightarrow$ *delivery*(h, c^-),
- *delivery*$(h, a^+) \rightarrow$ *delivery*(h, d),
- *delivery*$(h, c^+) \rightarrow$ *delivery*(h, b^-) and
- *delivery*$(h, d) \rightarrow$ *delivery*(h, b^-)

Several causal ordering protocols exist, most of them are based in the method of Vector Clocks introduced by Mattern [14]. In this work, in order to ensure the causal delivery of endpoints, we take the approach proposed by Birman et al. [15]. Birman's work attaches per message a vector VT of size n, where n is the number of processes in the system. At this time, more efficient causal protocols [16], [17] exist compared with Birman's work; nevertheless, we have chosen this protocol since it is a classical protocol that continues as a point of reference.

Correction of the synchronization error. For this task, the endpoint messages (causal messages) are used as synchronization points. Each time that an endpoint is received by a participant, two correction actions are executed. First, the deliveries of messages are delayed when they do not satisfy the causal dependencies, and secondly, messages are discarded when the maximum waiting time (Δ_{il}) is exceeded. In our case, we establish the value of Δ_{il} according to the maximum synchronization error tolerated between the type of media involved [18].

Waiting time period. In order to determine how much time a message $m = (i, t)$ must wait from its reception to its delivery, we define the following function:

```
$wait(m) : i = Part(m)$
$\{\ \forall\ l = 1...n,\ \ l \neq i$
    $a) VT(m)[l] \leq (t' = VT(p)[l])$ or      /*$m' = (l, t')$*/
    $b) receive\_time_p(m) + remainder\_time_{m'}(i, l)$
                                          $\leq current\_time(p)\ \}$
```

Where $receive\_time_p(m)$ returns the reception time of m at participant p. The function $current\_time(p)$ has the existing time at p and the $remainder\_time_{m'}(i, l)$ returns the possible wait time for the reception of m'. In general, the function $remainder\_time_{m'}(i, l)$ for a message m' is defined as follows:

$$remainder\_time_{m'}(i, l) =$$
$$\begin{cases} \Delta_{il} - (last\_rcv_p(i) - last\_rcv_p(l)), & \text{if } > 0, \\ 0 & \text{otherwise.} \end{cases}$$

The Δ_{il} is the maximum waiting time established between the media data with sources i and l, and the $last\_rcv_p(k)$ gives the time when participant p received the last message by a participant k.

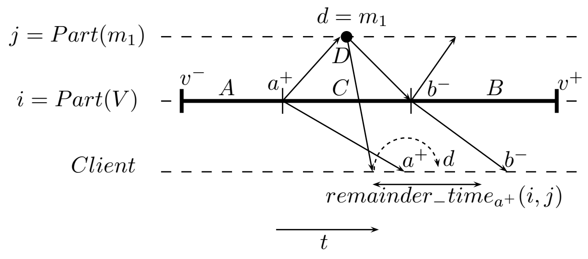


Figure 3. Example of synchronization error correction

Therefore, the $wait(m)$ function establishes that a message m sent by participant i ($i = Part(m)$) and received by a participant p will be immediately delivered, only after that, each message $m' = (l, t') : m' \rightarrow m$ has been either delivered at p (first condition) or the maximum waiting time Δ_{il} has elapsed (second condition). If the second condition is satisfied, then message m' is discarded by participant p, and every message $m'' = (l, t'')$, such that $t'' > VT(p)[l]$ and $t'' < t'$ is also discarded .

In Figure 3, we show an example of the behavior of the $wait(m)$ function. In this case, the delivery of message m_1 is delayed until the delivery of a^+ since a^+ precedes the sending of m_1 ($a^+ \rightarrow m_1$). Since a^+ has not been delivered to p, vector $VT(p)$ does not satisfy the first condition of the $wait()$ function. In this scenario, message a^+ is delivered because it arrives before the expiration of the maximum waiting time Δ_{ij}; otherwise, this message will be discarded.

A. Measuring the Synchronization Error

In order to evaluate the impact of the synchronization mechanism proposed in this paper, we define two equations, the $rcv\_synch\_error_p(m)$ and $dlv\_synch\_error_p(m)$. The function $rcv\_synch\_error_p(m)$ estimates the synchronization error by p at the reception of a causal message m, and $dlv\_synch\_error_p(m)$ estimates the synchronization error at the moment of the delivery of m, which is measured after applying the correction mechanism.

The $rcv\_synch\_error_p(m)$ is defined as follows:

$$rcv\_synch\_error_p(m) =$$
$$\frac{\sum_{l=1, l \neq i}^{n} receive\_time_p(m) - last\_rcv_p(l)}{(n-1)} \tag{1}$$

The $dlv\_synch\_error_p(m)$ is defined as follows:

$$dlv\_synch\_error_p(m) =$$
$$\frac{\sum_{l=1, l \neq i}^{n} delivery\_time_p(m) - last\_dlv_p(l)}{(n-1)} \tag{2}$$

Where $delivery\_time_p(m)$ returns the delivery time of m at participant p, and $last\_dlv_p(l)$ gives the time when participant p has delivered the last message seen from participant l. We note that these equations are only used to measure the synchronization error between continuous-continuos and continuos-discrete media since the logical

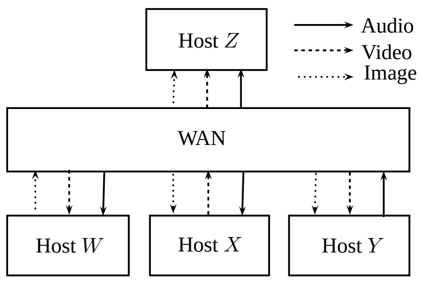


Figure 4. Emulation architecture

mapping model cannot establish a direct point of reference for discrete-discrete media. The synchronization between discrete-discrete media can only be done in an indirect manner if relations exist among the media of type discrete-continuous followed by continuous-discrete. We invite it to the reader to corroborate this assertion.

B. Emulation Results

We have emulated the previous synchronization approach considering some WAN network characteristics, such as asynchronous communication, variable transmission delay and partial order delivery. To emulate a WAN network behavior, we use the NIST Net emulation tool [19]. In order to emphasize the potential situation of unphased media data, we built an emulation environment, see Figure 4, that consists of a group of four distributed hosts, three of them transmitting live multimedia data, while the other functions only as *Client* (Host Z). Each host has two input and one output communication channels. The sending hosts only transmit one media (audio, video or images), see Figure 4. For the audio and video, we use MPEG-4 encoders of the MPEG4IP project [20]. For the case of audio, the silence compression MPEG-4 CELP mode is used. In this mode, we send a $begin$ message each time that the TX_Flag equals to 1 (indicates voice activity), and a frame is send as an end message each time that TX_Flag $\neq$ 1 (indicates no or low voice activity). For the remaining audio frames, we send them as $fifo\_p$ messages. For the case of video, the video was encoded as a single video object into a single video object layer with a rate of 25 frames/s, the group of pictures (GoP) pattern is set to IBBPBBPBBPBB. The I frames are sent as $begin$ messages, and the last B frames of the patterns are send as end messages. The rest of the video frames are sent as $fifo\_p$ messages. To emulate the sending of images, we randomly generate and transmit discrete events, each approximately 1200 ms based on a random variable with normal distribution. These data are sent as $discrete$ messages. In this scenario, each host has the synchronization mechanism running, and only the $Client$ measures the synchronization error by using equations 1 and 2. For simplicity, we consider only one maximum waiting time for every pair of media data $\Delta = 240$ms, which is the maximum synchronization error established for image-audio in real time in [18].

With this configuration, we present two tests, one with optimal conditions, while in the other, the transmission delay surpasses the synchronization error tolerated. The traffic conditions are presented in Table VI.

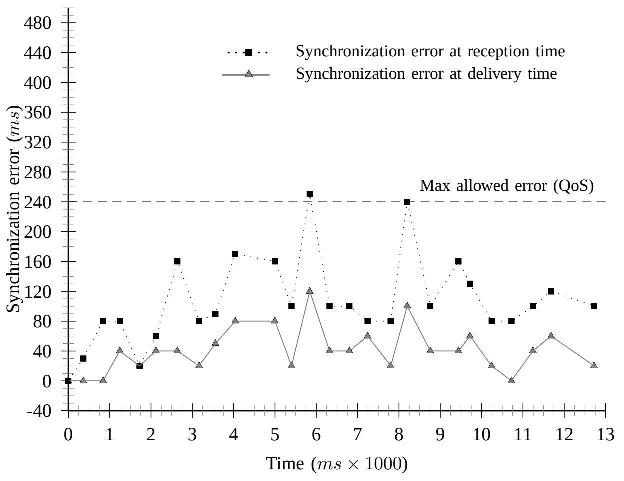


Figure 5. Test 1 graphics results: Audio and video (300 ± 120 ms); Image (200 ± 50 ms)

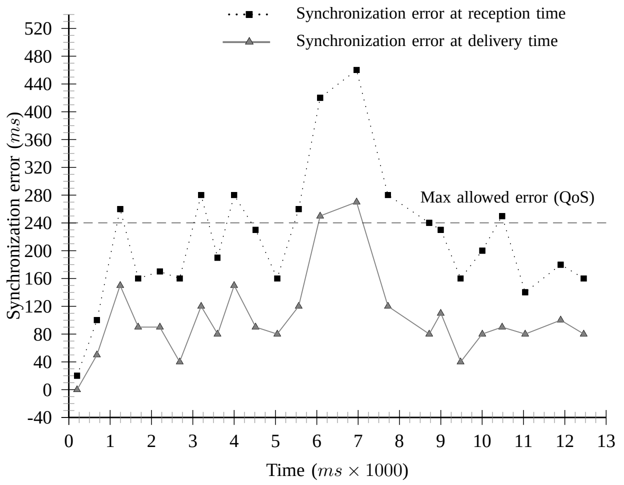


Figure 6. Test 2 graphics results: Audio and video (300±180 ms);Image (200 ± 100 ms)

Test 1. Under optimal conditions, we can see in Figure 5 that the estimated error at the reception time remains acceptable; nevertheless, our mechanism, at the moment of the media delivery, reduces the synchronization error that in this case is close to zero during nearly the entire transmission.

Test 2. For the second test, we can see in Figure 6 that at the reception time, the synchronization error surpasses in several points the maximum error tolerated. This effect can produce, at an application level, the loss of coherency among the media played. In this case, after applying our mechanism, the error is decreased, making it acceptable in nearly all cases.

VI. CONCLUSIONS

We have presented a distributed intermedia synchronization model that deals with continuous media and discrete media. The synchronization model presented avoids the use of a global reference by identifying

TABLE VI.
TRAFFIC CONDITIONS ESTABLISHED FOR EMULATION

Data Type	**Source**	**Target**	**Test 1 (ms)**	**Test 2 (ms)**
Video/Audio	Host X/Y	Host Z	300 ± 120	300 ± 180
Still images	Host W	Host Z	200 ± 50	200 ± 100

the causal dependencies of the media involved. The core of the synchronization model is the construction of logical mappings that are capable of specifying any kind of temporal relation (continuos-continuos, discrete-continuos, and discrete-discrete). We show that through the use of the logical mappings, our intermedia model has the ability to represent at runtime the relationships in a temporal scenario. We note that a logical mapping precisely establishes synchronization points that indicate when the media data should be reproduced. Finally, an approach to implement the synchronization model has been presented. The approach has been emulated considering WAN network characteristics. The emulation results show the viability of our model since the synchronization error has been reduced among the media data in the system.

REFERENCES

[1] I. S. P. Balaouras and L. Merakos, "Potential and Limitations of a Teleteaching Environment based on H.323 Audio-visual Communication Systems," *Computer Networks*, p. 945, 2000.

[2] D. Bulterman, and et al. Editors, "World Wide Web Consortium, Synchronized Multimedia Integration Language," SMIL 3.0, http://www.w3.org/TR/2007/WD-SMIL3-20070713.

[3] C. Perkins, *RTP Audio and Video for Internet*. Addison Wesley, 2003.

[4] S. Bechhofer, and et al. Editors, "World Wide Web Consortium, Time Ontology in OWL," http://www.w3.org/TR/2006/WD-owl-time-20060927.

[5] K. Shimamura, K. Tanaka, and M. Takizawa, "Group Communication Protocol for Multimedia Applications," in *Proceedings of the IEEE ICCNMC'01*, 2001, p. 303.

[6] C. Plesca, R. Grigoras, P. Quinnec, and G. Padiou, "Streaming with Causality: a Practical Approach," in *Proceedings of the 13th annual ACM international conference on Multimedia*, 2005, p. 283.

[7] T. Little and A. Ghafoor, "Synchonization and Storage Models for Multimedia Objects," *IEEE Journal on Selected Areas in Communications*, p. 413, 1990.

[8] J. Allen, "Maintaining knowledge About Temporal Intervals," *Communications of the ACM*, p. 832, 1983.

[9] L. Lamport, "Time Clocks and the Ordering of messages in Distributed Systems," *Communications ACM*, p. 558, 1978.

[10] J. Fanchon, K. Drira, and S.E. Pomares Hernandez, "Abstract Channels as Connectors for Software Compo nents in Group Communication Services," in *Proceedings of the Fifth Mexican International Conference on Computer Science*, 2004, p. 88.

[11] L. Lamport, "On Interprocess Communications: I. Basic Formalism," *Distributed Computing*, p. 77, 1986.

[12] E. Bertino and E. Ferrari, "Temporal Synchronization Models for Multimedia Data," *IEEE Transactions on Knowledge and Data Engineering*, p. 612, 1998.

[13] M. Vilain, "A System for Reasoning about Time," in *Proceedings 2nd. (US) National Conference on Artificial Intelligence*, 1982, p. 197.

[14] F. Mattern, "Virtual Time and Global States of Distributed Systems," in *Parallel and Distributed Algorithms: Proceedings of the International Workshop on Parallel and Distributed Algorithms*, 1989, p. 215.

[15] K. Birman, A. Schiper, and P. Stephenson, "Lightweight Causal and Atomic Group Multicast," *ACM Transactions on Computer Systems*, p. 272, 1991.

[16] R. Prakash, M. Raynal, and M. Singhal, "An Adaptive Causal Ordering Algorithm Suited to Mobile Computing Environment," *J. Parallel Distrib. Comput.*, p. 190, 1990.

[17] S.E. Pomares Hernandez, J. Fanchon, and K. Drira, *The immediate dependency relation: an optimal way to ensure causal group communication*. University Press and World Scientific Publications, 2004, vol. 6, p. 61.

[18] A. Hac and C. Xue, "Synchronization in Multimedia Data Retrieval," *International Journal of Network Management*, p. 33, 1997.

[19] M. Carson and D. Santay, "Nist net - a Linux-based Network Emulation Tool," *ACM SIGCOMM Computer Communication Review*, p. 111, 2003.

[20] D. Mackie, B. May, and B. Eisenberg, "Mpeg4ip Open Source Project," http://mpeg4ip.sourceforge.net.

Saul Eduardo Pomares Hernandez is an Associate Researcher in the Computer Science Department at the National Institute of Astrophysics, Optics and Electronics, in Puebla, Mexico. He completed his PhD degree at the laboratory for Analysis and Architecture of Systems of CNRS, France in 2002, and his M.S. at the Centre of Research and Advanced Studies (CINVESTAVIPN) in Mexico. Since 1998, he has been researching in the field of distributed systems, partial order algorithms, multimedia synchronization, and security.

Luis Alberto Morales Rosales is currently a PhD student in the Computer Science Department at the National Institute of Astrophysics, Optics and Electronics, in Puebla, Mexico. He received his M.S. from the same institute. His research efforts focus on partial order algorithms, multimedia synchronization, fuzzy control systems, and distributed systems. His Postgraduate Studies have been and continue to be supported by the National Council of Science and Technology of Mexico (CONACYT).

Jorge Estudillo Ramirez is currently a PhD student in the Computer Science Department at the National Institute of Astrophysics, Optics and Electronics, in Puebla, Mexico. He received his M.S. from the same institute. His research efforts focus on partial order algorithms, security protocols, multimedia synchronization and mobile networks. His Postgraduate Studies have been and continue to be supported by the National Council of Science and Technology of Mexico (CONACYT).

Gustavo Rodriguez Gomez is an Associate Researcher in the Computer Science Department at the National Institute of Astrophysics, Optics and Electronics, in Puebla, Mexico. He completed his PhD from the same institute in 2002, and his M.S. in mathematics at the Science Faculty of the National Autonomous University of Mexico (UNAM). He has been researching in the field of scientific computing, simulation of continuous processes, analysis and design of scientific software, and numerical solution of partial and ordinary differential equations.