Temporal Representation for Multimedia Systems

Minglu Li, Yongqiang Sun, and Huanye Sheng

Dept. of Computer Science and Engineering
Shanghai Jiao Tong University
Shanghai, 200030, P.R.China
hysheng@sjtu.edu.cn

Abstract

This paper focuses on temporal representation for multimedia systems. On the basis of an interval-based indeterministic synchronization model presented in [1], we propose a new Generalized Synchronization Model (GSM), which has fewer, more powerful interval operators. Then we outlines a description language paradigm as a preferred approach to specify multimedia systems, and present a new Multimedia Synchronization Description Language (MSDL), which is based on the GSM. Additionally, we demonstrate how the MSDL can be used to describe a large number of present and future multimedia systems.

1 Introduction

Multimedia systems integrate a variety of media with different temporal characteristics, including time dependent media, such as audio, video, and animation, and time independent media, such as text, graphics, and images [6]. In monomedia environment, all media show the same basic temporal behavior. Time does not need any particular attention. Now, with the arising multimedia systems, various temporal relations between media objects become more and more important.

Assuring the correct temporal appearance of the media objects is called synchronization. We have provided a survey of a selection of most common existing temporal models for multimedia synchronization in [18]. Steinmetz [6] analyze synchronization properties in multimedia systems based on Hoare's CSP; a Milner's CCS based specification of multimedia composition is described in [7]; Little and Ghafoor [2,3] propose a synchronization model based on timed Petri nets called OCPN; and a specification of multimedia synchronization using path expressions is presented in [4]. These models are appropriate for the representation of multimedia synchronization, if the duration of all media objects is deterministic. However, as soon as interactive media added, they require more expressiveness, because interactive multimedia objects introduce a huge number of unpredictable duration. Wahl and Rothermel [1] propose an interval-based indeterministic synchronization model based on 10 interval operators, which is appropriate for the representation of interactive multimedia. In this paper, this model is called Wahl-Rothermel Synchronization Model (WRSM).

This paper focuses on temporal representation for multimedia systems. In section 2, a summary of all temporal relations is given, and the question Which out of all possible temporal relations might be needed for multimedia? is examined. We give a brief overview of the WRSM in section 3. Next, section 4 enhances the WRSM, and proposes a new Generalized Synchronization Model (GSM) which has powerful, yet fewer interval operators. It is followed by a formal description of the Multimedia Synchronization Description Language (MSDL) in section 5, which is then used to design multimedia systems. Finally, some concluding remarks and further research are given in section 6.

2 Temporal relations in multimedia

To systematically develop a complete set of temporal relations for multimedia, we examine how many and which relations are theoretically possible. Depending on their elementary units, two basic classes of temporal models can be distinguished [8]. In the first class, time is expressed by means of points in a one-dimensional time space [9] whereas, in a second model class, intervals are

0-8186-7528/96 $5.00

the atomic units of the time space [5]. This section introduces the temporal models, their elementary units and the relations between them, and evaluates the relevant temporal relations in multimedia.

2.1 Point-based model

In point-based temporal model, the elementary units are events, which are points in a time space. Given two events in history, only three relations can hold between them. An event can be before (<), simultaneous to (=) or after (>) a second event. The relations < , =, > are called the basic point relations.

In contrast to relations in the past, relations between future events might be indefinite. For example, we know that an event *e1* cannot occur after an event *e2*. This means that *e1* is before or simultaneous to *e2*. This is denoted as $e1<e2 \vee e1=e2$ or as $e1\ \{<,=\}e2$. Note that *e1* is before or simultaneous to *e2*, and it is not known which of the relations will become true. Typically, indefinite relations are represented as disjunctions of basic point relations. Since there are three basic point relations, $2^3=8$ disjunctions exist each representing an indefinite relation. For example, instead of $e1\ \{<,=\}e2$, we use $e1 \leq e2$. The 8 indefinite relations are : $\emptyset$, $\leq$, <, =, >, $\geq$, $\neq$, ? , where ? is the full set of basic point relations $\{<, =, >\}$, $\emptyset$ is the empty set { }, and the others are self-explaining.

2.2 Interval-based model

Intervals are the basic units of a time model class suggested by [5][10][11]. There are 13 basic interval relations. An enumeration of the 13 basic interval relations is given in [5].

In analogy to the point relations, 2^{13} indefinite interval relations can be defined as disjunctions of the basic interval relations.

2.3 Translations between the models

As any interval can be characterized by its starting and end point, any basic interval relations can be represented by a conjunction of point relations on its margins. Also, a number of indefinite interval relations can be represented by a point relation conjunction on its margins. Table 1 summarizes how many indefinite interval relations are representable by point relation conjunctions.

2.4 Relevant point relations in multimedia

point relations	numbers of interval relations
<, =, >	13
<, =, >, ?	29
≤, <, =, >, ≥, ?	82
≤, <, =, >, ≥, ≠, ?	187

Table 1. Interval relations generated by point relations

From section 2.1, we learned that 8 indefinite pointed relations exist. However, are all of them necessary to specify time in multimedia? Obviously, the basic point relations <, =, > occur in multimedia because presentation events might be before, simultaneous to or after other events.

To evaluate the indefinite point relations ≤, ≥, ≠, ?, we have to consider the fact that small inaccuracies are tolerated in multimedia. E.g. in a video-audio presentation, the audience does not notice the skew introduced if the audio is presented too early or too late by some milliseconds [12][13]. So, we do not need to specify the temporal behavior at exactly one point in time rather it is sufficient to specify the temporal behavior close to each point in time. This implies that there is not perceptible difference in the presentation if somebody specifies for two events *e1* and *e2* that $e1<e2$ or in the second case $e1 \leq e2$. This holds because the audience cannot distinguish whether *e1* is simultaneous to *e2* or *e1* is *1* millisecond before *e2*. Therefore, it is sufficient to be able to express only one of the relations < or ≤. In this paper, we operate with the relations < and >, and do not need the relations ≤ and ≥. Analogically, the relation ≠ differs from ?-relation only in one point in time. Since there is not perceptible difference between the relations, we do not need the relation ≠ if we have the relation ? . Observe that we need the relation ? if any basic point relation can hold between two events.

Finally, the point relation $\emptyset$ might be used to specify in consistencies because $\emptyset$ means that no temporal relation between two events exists. Consequentially, only either of the events can occur. In this paper, we focus on consistent scenarios. So we do not consider the inconsistent relation $\emptyset$.

To summarize, the relation <, =, > and ? are relevant point relations in multimedia environment. Powerful point-based temporal models should be able to express at least this set of relations.

3 Wahl-Rothermel synchronization model

Section 2 showed the 4 multimedia-relevant point relations generate 29 interval relations. In [1], Wahl and Rothermel propose an interval-based indeterministic

synchronization model which can cover the 29 interval relations. In this section, we make a brief overview of the main features of the WRSM.

If an interval operator is defined for each interval relation, the 29 operators represent a complete model. However to use 29 different operators, seems to complicate a presentation specification. Therefore, it would be good to have fewer, more powerful operators which are still intuitive to multimedia authors.

Fortunately, the number of operators can be reduced by exploiting regularities between the interval relations. The regularities can be given as follows.

1) Some interval relations are inverse to each other.

2) Some interval relations differ only by an offset from each other.

Then, several interval relations can be combined to one operator and the number of operators can be reduced from the original 29 to 10. Figure 1 shows the generic pattern for each of the 10 operators. In Figure 1, we use timed Petri nets [2][3] to represent time specifications.

In Figure 1, the delay parameter δi is described as a subset of non-negative real numbers $\mathbf{R_0}^+$. We use the notation 0 if the delay is zero, + if the delay has a positive value, and * if the delay is positive or zero.

To avoid having several specification methods for the same interval relation, we require $\delta i \neq 0$ for some of operators in Figure 1. Then, the 10 operators are a complete set to specify any of the 29 interval relations generated by <, =, > and ? .

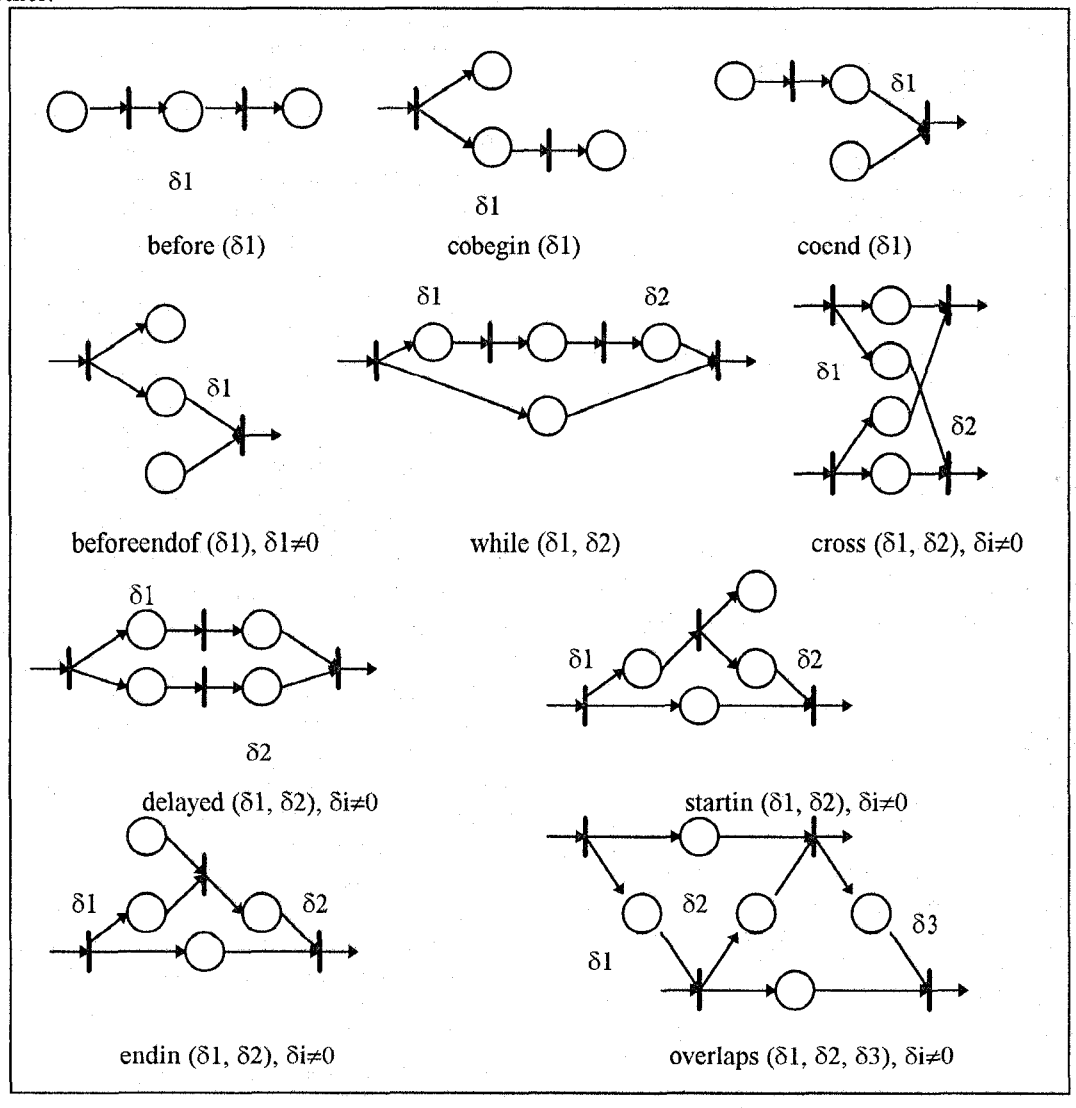

Figure 1. Generic interval operators in WRSM

For example, if we specify *media_1 while (0,*) media_2* , media object 1 is displayed during the presentation of media object 2. The parameter $\delta 1$ describes the delay between the beginning of *media_1* and *media_2*. In this example, both media start simultaneously because $\delta 1$=0. The second delay parameter $\delta 2$ says how much time after the end of *media_1* the media object 2 has to be terminated. In this case, both objects might end simultaneously or *media_2* ends any time after *media_1* has finished because $\delta 2$=*. This relation occurs for example if *media_2* has to be terminated interactively.

The 10 operators are a complete set to specify any of the 29 interval relations which are relevant for interactive multimedia. However, this does not imply that all operators are needed to define a complete model for a multimedia environment. Sometimes, only a selection of the operators is necessary. For example, the path expressions synchronization model [4] and the Object Composition Petri Net (OCPN) synchronization model [2,3], are both based on Allen's [5] 13 temporal relations which can be indicated by a timeline representation. They suppose the duration of all media objects is preknown and fixed, the temporal model may be restricted to the operators taking at most 1 delay parameter. Note that the requirements 'preknown and fixed duration' are very strict and prohibit any kind of interaction or flexibility. With the emerging interactive multimedia systems, it is expected that a larger subset of the interval operators is needed because interactive media objects introduce a huge number of unpredictable duration.

4 Generalized synchronization model

In section 3, the 10 operators are used to represent a complete synchronization model. However, to use 10 different operators, seems to complicate a presentation specification. Therefore, it would be good to have fewer, more powerful operators which are still intuitive to multimedia authors and users. This section gives two approaches to enhance the WRSM. On the basis of the enhancements, we propose a new Generalized Synchronization Model (GSM).

4.1 Pseudo-dynamic delay θ

First, we introduce a new pseudo-dynamic delay parameter θ which is described as a subset of real number **R**. θ can be defined as follow.

$$\theta = Sb - Sa \quad (1)$$

where, *Sa* denotes the starting of the interval *a* and *Sb* the starting of the interval *b*.

The semantic of θ is that, if before the execution of two parallel media objects it is unpredictable, but if the two parallel media objects once begin executing, it is determinable. Hence, θ is called pseudo-dynamic delay.

Theorem 1: Using θ, the 4 operators including pseudo-dynamic delay parameters: *coend, beforeendof, cross,* and *endin*, can be represented by the others.

Proof. We can give the enumeration of the 4 transformations as follows.

$$a\ coend(\delta 1)\ b ::= \text{if } \theta>0 \text{ then } b\ delayed(\theta, \delta 1)\ a \text{ else } a\ while(\theta, \delta 1) \quad (2)$$

$$a\ beforeendof(\delta 1)\ b ::= \text{if } \theta>0 \text{ then } a\ cobegin(\theta)\ b \text{ else } a\ startin(\theta, \delta 1)\ b \quad (3)$$

$$a\ cross(\delta 1, \delta 2)\ b ::= \text{if } \theta>0 \text{ then } b\ startin(\theta, \delta 2)\ a \text{ else } a\ startin(\theta, \delta 1)\ b \quad (4)$$

$$a\ endin(\delta 1, \delta 2)\ b ::= \text{if } \theta>0 \text{ then } a\ overlaps(\theta, \delta 1, \delta 2)\ b \text{ else } a\ while(\theta, \delta 2)\ b \quad (5)$$

□.

4.2 Operator of highest degree of expressiveness

In terms of Theorem 1, we can use the 6 operators to represent the proposed synchronization model by introducing the pseudo-dynamic delay θ, such as *before, cobegin, while, delayed, startin,* and *overlaps*. Which is the operator of the highest degree of expressiveness?

Theorem 2: In the 6 operators, *cobegin* is an unique operator of the highest degree of expressiveness.

Proof. In one hand, we must justify that *cobegin* can represent the other operators.

We introduce two delay parameters γ and φ, and give the definitions as follows.

$$\gamma = Ea - Sb \quad (6)$$

$$\varphi = Eb - Ea \quad (7)$$

where, *Ea* denotes the end of the interval *a*, and *Eb* the end of the interval *b*.

Using γ and φ, we can give the enumeration of the 5 transformations as follows.

$$a\ before(\delta 1)\ b ::= a\ cobegin(+)\ b, \text{ when } \gamma = -\delta 1 \text{ and } clearly\ \varphi>0 \quad (8)$$

$$a\ while(\delta 1, \delta 2)\ b ::= b\ cobegin(\delta 1)\ a, \text{ when } \gamma>0 \text{ and } \varphi=\delta 2 \quad (9)$$

$$a\ delayed(\delta 1, \delta 2)\ b ::= b\ cobegin(\delta 1)\ a, \text{ when } \gamma = * \text{ and } \varphi=-\delta 2 \quad (10)$$

$a\ startin(\delta 1, \delta 2)\ b ::= a\ cobegin^{-1}(\delta 1)\ b,$
when $\gamma > 0$ and $\varphi = *$ (11)

$a\ overlaps(\delta 1, \delta 2, \delta 3)\ b ::= a\ cobegin(\delta 1)\ b,$
when $\gamma = \delta 2$ and $\varphi = \delta 3$ (12)

where, we use the notation '+' if the delay has a positive value, '*' if the delay is any real number.

On the other hand, we can prove that each of the operators exception of *cobegin* can not represent the others by giving counterevidences. This is very easy, so we may omit it.

□.

4.3 GSM

Intuitively, we think of that sequential processing is different from parallel processing. As a result, the GSM can be described by following theorem.

Theorem 3: The complete synchronization model can be represented by the two operators: *before* and *cobegin*, and the pseudo-dynamic delay θ.

Proof. We can directly give the result by using Theorem 1 and limiting $\gamma > 0$ in Theorem 2.

□.

5 Multimedia synchronization description language

5.1 Atomic language G^0

An atomic language G^0 to specify the generalized synchronization model presented in section 4.3, can be defined as follow.

$G^0 ::= A \mid A\ ;\ A \mid A \parallel A \mid$ if E then A else $A \mid \{ A \}$ (13)

where, A is a statement which specifies the execution of a media object; E is a Boolean expression defined as follow.

$E ::=$ if $\theta > 0$ then *TRUE* else *FALSE* (14)

and the operators ';' , '||' and { }are named and listed below.

$\{ A \}$ *Composition:* composed statement A is composed of some statements. The composed statement terminates when all participating statements terminate.

$A \parallel B$ *Parallel*: statement A and B are started at a common startpoint and are executed concurrently. The composed statement terminates when all participating statements terminate.

$A\ ;\ B$ *Sequential*: it is only possible to execute B if A is executed before. The endpoint of A equals the startpoint of B. The composed statement terminates when the last statement in the sequence terminates.

Theorem 4: Atomic language G^0 has the same expressiveness as the GSM.

Proof. First, it is clear that the definition of E includes pseudo-dynamic delay θ from equation (14). Then, the two temporal relations and their operators in Theorem 3 can be transformed into the statements in G^0 (see Table 2).

relations in GSM	statements in G^0
$a\ before(\delta 1)\ b$	$A\ ;\ \Delta 1\ ;\ B$
$a\ cobegin(\delta 1)\ b$	$A \parallel \{ \Delta 1\ ;\ B \}$

Table 2. Using G^0 to represent GSM

□.

5.2 MSDL

The Multimedia Synchronization Description Language (MSDL) aims at design of multimedia systems. The statements of the MSDL are divided two sorts: declaration statements and action statements. The declaration sort defines which multimedia objects make the system up and where they are stored. The action sort is concerned with the way these objects are presented. An action statement can be a primitive or a combination of primitives. The MSDL does not describe how primitive action statements are executed, because the execution of such action statement is intrinsically related with the underlying hardware and this clearly is an implementation concern.

The MSDL deals mainly with the relations between the action statements. The atomic language G^0 must be included in the MSDL, i.e. G^0 is a subset of the MSDL. G^0 is used to determine the way which the action statements are executed, in sequence or in parallel. We can give the abstract syntax of the MSDL as follows.

The MSDL G is composed by a set of declaration statements D followed by a set of action statements A.

$G ::= D\ ;\ A$ (15)

where, ';' denotes sequential execution.

To declare a multimedia object, it is necessary to give its identifier, its type and where it is stored.

Let $d \in D$, $h \in H$, $s \in S$, $c \in C$, then

$D ::= d \mid D\ ;\ D$ (16)

$d ::= h :: s < c$ (17)

where, H is a set of identifiers of multimedia objects which can be specified by character strings; S is a set of

types of multimedia objects which include almost all human-computer interaction objects.

S ::= {*NULL, text, numeric-data, graphic, image, video, audio, animation,button, menu, dialog-box, window, ...*} (18)

C is a set of characteristics of multimedia objects, e.g. the name of the file where the object is stored.

Action statements can be defined as follow.

$A ::= a \mid A ; A \mid A \parallel A \mid A >< A \mid$ if E then A else $A \mid$ while E do A $\mid \{ A \}$ (19)

$a ::= f_t (h, t > Pi) \mid f_{s1} (h > Pi) \mid f_{s2} (h1, h2 > Pi)$ (20)

where, $a \in A$; E is a Boolean expression which can be made up by any Boolean combinator; L is a set of labels which label the action statements; $Pi \subseteq P$; $f_t \subseteq F_t$; $f_{s1} \subseteq F_{s1}$; $f_{s2} \subseteq F_{s2}$; $F = F_t \cup F_{s1} \cup F_{s2}$; F is a set of action functions. P is set of parameters of action functions.

F_t is a set of temporal action functions which can be of the timed parameter t, t can be used to specify the duration in which f_t is executing. If the value of t is not preknown, we denote it as '*', and f_t can be terminated by itself or by the other parallel action function.

F_t ::= {*Show, Play, Delay, WaitEventOn,...* } (21)

F_{s1} is a set of monadic spatial action functions which can change the spatial layout or appearance of one multimedia object.

F_{s1} ::= {*Assign, Scaling, Reverse, Mirror, Move, Rotation,...*} (22)

F_{s2} is a set of binary spatial action functions which can deal with the spatial relations between two multimedia objects.

F_{s2} ::= {*Or, And, Xor, Cover,...*} (23)

Then, we give the informal semantic of the combinators as follows.

$\{ A \}$ *Composition:* composed statement A is composed of some statements. The composed statement terminates when all participating statements terminate.

$A \parallel B$ *Parallel_last*: statement A and B are started at a common startpoint and are executed concurrently. The composed statement terminates when all participating statements terminate.

$A >< B$ *Parallel_first*: statement A and B are started at a common startpoint and are executed concurrently. The composed statement terminates when the first (in time) participating statement terminates.

$A ; B$ *Sequential*: it is only possible to execute B if A is executed before. The endpoint of A equals the startpoint of B. The composed statement terminates when the last statement in the sequence terminates.

if_then_else and *while_do* statement have the same semantic as ones in the other programming languages.

The formal semantic of the MSDL is the subject of a forthcoming paper.

5.3 Example I: using the MSDL to design multimedia interface

A multimedia interface description is used to demonstrate the MSDL works. Figure 2 shows how an author specifies an interactive multimedia presentation applying the MSDL.

A graphic `Graph1`, which is stored in file `"C:\CAD\GRAPHIC.DXF"`, is shown 10 seconds. While the graphic is being presented, a user can play the music fragment `Audio1`, pressing the button `Button`, which is executed for 5 seconds. The appearance of `Button` is stored in file `"C:\ICON\BUTTON.ICO"`, `Audio1` is stored in file `"C:\WAVE\MUSIC.WAV"`. When `Audio1` completes, the video `Video1` in parallel with audio `Audio2` is shown. `Video1` is stored in the `VideoDiscPlayer` device, while `Audio2` in the `CD_Player` device. If a user event does not occur, when `Graph1` terminates, `Video1` and `Audio2` begin.

```
Graph1::graphic<C:\CAD\GRAPHIC.DXF;
Audio1::audio  <C:\WAVE\MUSIC.WAV;
Video1::video  <VideoDisc_Player;
Audio2::audio  <CD_Player;
Button::button <C:\ICON\BUTTON.ICO;

Show(Graph1,*>P1)><
  if WaitEventOn(Button,10>P2)=TRUE
    then Play(Audio1,5>P3);

Play(Video1,*>P4)||
  Play(Audio2,*>P5)
```

Note: where, $P i \subseteq P$, $i = \{1, 2, ...,5\}$.

Figure 2. Example for designing multimedia interface

5.4 Example II: using the MSDL to design interactive multimedia slide show

Figure 3 describes an interactive multimedia slide show specification. Consider a slide sequence `Slide[N]` and a talk sequence `Talk[N]`. Any slide and its corresponding talk are presented simultaneously. Before each talk, there is a silence of 3 seconds to give the spectator a first impression of the slide. After a talk is finished, the user can look at the slide silently or continue to the next slide interactively.

```
X        ::integer<*;
N        ::integer<CONSTANT;
Slide[N]::image   <SLIDE.DBF;
Talk[N]  ::audio  <TALK.DBF;
Button   ::button <BUTTON.ICO;

Assign(x,0);

while X<N do
  Show(Slide[X],*>P1) >< {
    Delay(NULL,3);
    Play(Talk[X],*>P2);
    if WaitEventOn(Button,
       *>P3)=TRUE
      then Assign(X,X+1);
    }
```

Note: where, $P_i \subseteq P$, $i = \{1, 2, 3\}$.

Figure 3. Example for designing interactive slide show

6. Conclusion and further research

We have shown a generalized model for multimedia synchronization, and proposed a new Multimedia Synchronization Description Language (MSDL). We have also demonstrated the MSDL is appropriate for specifying interactive multimedia systems, because it uses high-level abstractions and has a high degree of expressiveness.

Some progress has already been made in research of multimedia synchronization and description language [14-18]. The work on the MSDL is now moving from the conceptual analysis to detail system design to be followed by a prototype implementation. Some of issues that we intend to address in the near future are described below.

Formal semantic: It is an important issue to discuss the formal semantic of the MSDL, because there are large number of interactions in multimedia systems, correspondingly there are large number of operators which must be supported in the MSDL.

Implementation: A translator or an interpreter which allows direct implementation of the MSDL specification will be useful. Once interactive multimedia systems can be realized from the MSDL specification, work can progress in the opposite direction: designing a visual programming environment which can generate the MSDL specification automatically.

References

[1] T. Wahl and K. Rothermel, Representing time in multimedia systems, *In Proc. of the IEEE International Conference on Multimedia Computing and Systems*, Boston, MA, USA, May 1994

[2] T.D.C.Little and A.Ghafoor, Synchronization and storage models for multimedia objects, *IEEE Journal on Selected Areas in Communications*, 8(3):413-422, April 1990

[3] T.D.C.Little and A.Ghafoor, Interval-based conceptual models for time-dependent multimedia data, *IEEE Transactions on Knowledge and Data Engineering*, 5(4):551-563, Aug. 1993

[4] P.Hoepner, Synchronizing the presentation of multimedia objects, *Computer Communications*, 15(9):557-564, Nov. 1992

[5] J.F.Allen, Maintaining knowledge about temporal intervals, *Communications of the ACM*, 26(11):832-843, Nov. 1983

[6] R.Steinmetz, Synchronization properties in multimedia systems, *IEEE Journal on Selected Areas in Communications*, 8(3):401-412, April 1990

[7] S.B. Eun et al, Specification of multimedia composition and a visual programming environment, In *Proc. of ACM Multimedia'93*, pp167-173, Aug. 1993

[8] P.van Beek, Reasoning about qualitative temporal information , *Artificial Intelligence*, 8:297-326, Dec. 1992

[9] M.Vilain and H.A.Kautz, Constraint propagation algorithms for temporal reasoning , In *AAAI-86*, pp132-144, 1986

[10] B.C.Bruce, A model for temporal reference and its applicants in a question answering program , *Artificial Intelligence*, 3:1-12, 1972

[11] C.L.Hamblin, Instants and intervals , In *1st Conf. Int. Soc. for the Study of Time*, pp324-331, Springer, 1972

[12] T.D.C.Little and F.Koa, An intermedia skew control system for multimedia data presentation , In *3rd Intl. Workshop on Network and Operating Systems Support for Digital Audio and Video*, pp121-132, Nov. 1992

[13] K.Rothermel and G.Dermler, Synchronization in Joint-Viewing Environments , In *3rd Intl. Workshop on Network and Operating Systems Support for Digital Audio and Video*, Nov. 1992

[14] Minglu Li, Yongqiang Sun, and Huanye Sheng, Synchronization support for multimedia communications, In *Proceedings of 4th International Conference for Young Computer Scientists*, Beijing, China, July 1995

[15] Minglu Li, Yongqiang Sun, and Huanye Sheng, An interval-based nondeterministic model for multimedia synchronization, In *Proc. of International Workshop on Advanced Parallel Processing Technologies*,Beijing, China, Sept. 1995

[16] Minglu Li, Yongqiang Sun, and Huanye Sheng, Formal description of multimedia presentations, In *Proceedings of 4th International Conference on CAD & CG*, Wuhan, China, Oct. 1995

[17] Minglu Li, Yongqiang Sun, and Huanye Sheng, MSDL: an approach to design multimedia interfaces, To appear in *1st International Conference on Multimodal Interface*, Beijing, China, June 1996

[18] Minglu Li, Yongqiang Sun, and Huanye Sheng, Temporal models for multimedia synchronization, Submitted to *Journal of Computer Science and Technology*.