Synchronization in Multimedia Documents*

Peter King[1], Helen Cameron[1], Howard Bowman[2], and Simon Thompson[2]

[1] Department of Computer Science, University of Manitoba, Winnipeg, Manitoba, R3T 2N2, Canada
[2] Computing Laboratory, University of Kent at Canterbury, Canterbury, Kent, CT2 7NF, United Kingdom

Abstract. This paper presents a taxonomy of possible synchronization relationships between pairs of items in multimedia documents. Several existing approaches to the synchronization of entire items are reviewed. We then discuss classes of synchronization based upon dynamic events or conditions occurring within media items and their internal structure. We present a taxonomy of seventy-two possible such relations, which are illustrated by numerous examples and which are formalized in the authors' temporal logic notation, *Mexitl*. The ideas are then applied to provide a description of the lip-synchronization problem.

1 Introduction

Multimedia documents, their description and means for their authoring, are the subject of a considerable volume of current research and development work. The term "multimedia" refers to a document containing continuous or time dependent components, which are termed media items; see Erfle [Erf93]. A major part of the task facing the author of such a document is the specification of the temporal relationships among the media items in such a document. The present paper discusses our approach to this question.

In this paper we are primarily interested in temporal constraints between pairs of media items. Many authors have made use of the well-known taxonomy described by Allen [All83], who presents a complete set of thirteen such binary relationships or constraints between media items, as a starting point for defining a set of temporal relations. Shih *et al* [SHT96] have pointed out that the Allen relations are not sufficient from the viewpoint of authoring since they do not contain any precise timing information, which is typically a basis for item synchronization in multimedia documents. To this drawback, we add that the Allen relations treat each of the two media items as continuous and indivisible. There is no (direct) provision for synchronizing one object upon events or conditions occurring dynamically in the second object. The main objective of this paper is to address this latter point.

* The authors acknowledge the assistance of the British Council in providing travel grants, and the Natural Sciences and Engineering Council of Canada for research grants.

We will present what we consider to be a complete set of binary media item synchronization relationships. We will do this by presenting a set of relationships as the Cartesian product of various classes of primitive synchronization conditions which may occur in the two items which are to be related.

Each such synchronization condition will be discussed from three viewpoints. First, we will present an informal description of the synchronization relation. Second, we will present examples from hypothetical multimedia documents, illustrating the synchronization relation. Third, we will give a formal description of the synchronization relation. While most of our discussion will be limited to synchronization relations between a pair of media items, the ideas certainly extend to several items.

The formal descriptions of the synchronization relations will be given in the *Mexitl* notation. *Mexitl* [BCKT97a,BCKT97b] is a formal notation developed by the authors for use in specifying multimedia document. *Mexitl* is based on an interval temporal logic. It is a central component of the long-term goals of the authors: to make use of formal methods in the development of an authoring tool, which would thereby provide a means to address issues such as consistency verification, modeling, prototyping, and specification refinement. Although *Mexitl* itself is not the primary subject matter of this paper, it is nonetheless a convenient vehicle for the formal expressions we wish to provide, and equally, we seek to support the claim that *Mexitl* is a complete formalism in the pragmatic sense mentioned above. Accordingly, a brief description of *Mexitl* will be included.

The remainder of the paper will be organized as follows. Section 2 reviews the Allen set, and also makes use of it to introduce the *Mexitl* language. We will also elaborate on some of the shortcomings of the Allen set in this application area. Section 3 presents an informal introduction to our view of item synchronization, and presents a number of illustrations. Section 4 defines synchronization events and synchronization items, and uses these notions to provide a new taxonomy of possibilities for media item synchronization. Section 5 uses these notions to develop a more complex example, that of lip-synching, and also includes a *Mexitl* version of this problem. Section 6 concludes.

2 Synchronization of Continuous Items: Allen and *Mexitl*

In this section, we are concerned with synchronizing two continuous items, where the synchronization is defined in a manner which is independent of any events or conditions occurring in the two items. The items, therefore, are to be regarded as indivisible. The taxonomy of binary relations between two temporal intervals introduced in Allen [All83] is well known, and has been used by a number of authors [JLSI97,KS95,Kin96,MHM96,SHT96]. We re-introduce it here for completeness and as a means of introducing some of the components of the *Mexitl* notation.

2.1 *Mexitl*

We will present only the briefest introduction to *Mexitl*, concentrating on those features needed in this paper. The reader is referred to [BCKT97a,BCKT97b] for further details. *Mexitl* is an interval temporal logic, that is, *Mexitl* propositions are interpreted over finite sequences of states, known as intervals. In applying *Mexitl* to multimedia descriptions, the intervals are to be regarded as time intervals, with the component states occurring at equal time intervals (of whatever granularity is desired). *Mexitl* propositions may be regarded as logical statements which thus have the value **True** or **False** over the interval in question. Alternatively, they may be interpreted using an imperative dictum, which has the effect of assigning values to free variables occurring in the various formulae.

Mexitl supports a type definition mechanism, whereby various classes of media items can be introduced, such as video-clip, audio-segment, etc. It also supports a facility for declaring named instances of those types. *Mexitl* also supports actions, which are discrete and atomic. The display of a media item is represented by an interval proposition which corresponds to the sequence of actions comprising that item. Other aspects of *Mexitl* will be introduced as they are needed.

2.2 *Mexitl* and the Allen Relations

Of the thirteen Allen relations, six are converses of other relations and we therefore concern ourselves with seven relations: equal, meets, before, during, starts, finishes and overlaps. These relations are defined in *Mexitl* using the following operators: $\wedge$, $\mathbf{;}$, $\Diamond$, $\Diamond_{\mathrm{i}}$ and $\Diamond_{\mathrm{a}}$, which have the following meanings:

$a \ \wedge \ b$: true over an interval over which both a and b hold;
$a \ \mathbf{;} \ b$: true over an interval that can be split into two contiguous subintervals such that a holds over the first subinterval and b holds over the second;
$\Diamond a$: true if a holds over a suffix subinterval;
$\Diamond_{\mathrm{i}}\, a$: true if a holds over a prefix subinterval;
$\Diamond_{\mathrm{a}}\, a$: true if a holds over an arbitrary subinterval.

Most of the Allen relations can be specified using these operators:

aEb	"equals"	$a \ \wedge \ b$
aMb	"meets"	$a \ \mathbf{;} \ b$
aBb	"before"	$a \ \mathbf{;} \ \Diamond b$
aDb	"during"	$\Diamond_{\mathrm{a}}\, a \ \wedge \ b$
aSb	"starts"	$a \ \wedge \ \Diamond_{\mathrm{i}}\, b$
aFb	"finishes"	$\Diamond a \ \wedge \ b$

For the remaining relation, we introduce **mylen**, which yields the length of the current interval, and $\hat{x}$, which yields the length of media item x. We have

$$\text{aOb “overlaps”} \ \Diamond_{\mathrm{i}} \ a \ \wedge \ \Diamond \ b \ \wedge (\mathbf{mylen} < (\hat{a} + \hat{b}))$$

Before proceeding, we present some other basic *Mexitl* operators:

$\neg$	"not"	The first three operators are familiar
$\Rightarrow$	"implies"	from propositional logic.
False		We also derive **True** $\equiv$ $\neg$**False**.
$=$		Equality of expressions.
$\bigcirc P$	"next"	P holds over the interval which begins in the next state.
$\Box P$	"always"	P holds on all terminal subintervals.
$\boxed{\scriptstyle \mathrm{i}}\, P$	"always initially"	P holds on all initial subintervals.
$\boxed{\scriptstyle \mathrm{a}}\, P$		P holds on all arbitrary subintervals.

These operators are not independent; for example, we have

$$
\begin{aligned}
\boxed{\scriptstyle \mathrm{a}}\, P &\equiv \neg\ \Diamond_{\mathrm{a}} \neg P \\
\Box P &\equiv \neg\ \Diamond \neg P \\
\Diamond_{\mathrm{i}}\, P &\equiv P \ \mathbf{;}\ \mathbf{True} \\
\Diamond_{\mathrm{a}}\, P &\equiv \mathbf{True}\ \mathbf{;}\ P\ \mathbf{;}\ \mathbf{True}
\end{aligned}
$$

2.3 Revised Interval Temporal Relations

Although the Allen relations form a complete set, and accordingly the previous subsection may be regarded as a demonstration that *Mexitl* is complete in this regard, they are not entirely suitable for use in multimedia applications, since they lack provision for precise timing specifications. One might, for example, wish to use a stronger form of "overlaps" in which b starts precisely three time units after a does. Following Shih *et al* [SHT96], who introduce somewhat different temporal relations to address such questions, we now list such timed relations indicating how each would appear in *Mexitl*.

A time parameter is added to each of the three relations before, overlaps, and during.

n-before(a, b, n): b starts n cycles after a finishes.
n-overlaps(a, b, n): a overlaps b and b starts n cycles after a does.
n-during(a, b, n): b occurs during a and starts n cycles after a does.

In fact, n-overlaps and n-during are both special cases of

n-intersects(a, b, n): b starts n cycles after a does and before a finishes.

Synchronization in the case of the other four Allen relations is position dependent rather than time dependent, and they are, therefore, unchanged.

In order to specify these relations in *Mexitl*, we make use of the $\mathbf{len}(n)$ proposition, which is true over any interval of length n. We write

$$
\begin{aligned}
n\text{-before}(a, b, n) &\equiv a\ \mathbf{;}\ \mathbf{len}(n)\ \mathbf{;}\ b \\
n\text{-overlaps}(a, b, n) &\equiv \Diamond_{\mathrm{i}}\, a\ \wedge\ (\mathbf{len}(n)\ \mathbf{;}\ b)\ \wedge\ (\hat{b} > \hat{a} - n > 0) \\
n\text{-during}(a, b, n) &\equiv a\ \wedge\ \Diamond_{\mathrm{i}}(\mathbf{len}(n)\ \mathbf{;}\ b) \\
n\text{-intersects}(a, b, n) &\equiv \Diamond_{\mathrm{i}}\, a\ \wedge\ (\mathbf{len}(n)\ \mathbf{;}\ b)\ \wedge\ (n < \hat{a})
\end{aligned}
$$

As in many temporal logics, first order state conditions in *Mexitl*, such as $n < \hat{a}$, are lifted (coerced) into interval conditions; these conditions are evaluated in the first state of the interval.

It is in point of fact possible to specify the other four relations in this fashion, that is, making explicit the implicit interval lengths. In these cases, however, the length conditions would be assertions, used perhaps for consistency checking, rather than constituents of the specification of the relation. King [Kin96] refers to these specifications as display forms.

By way of concluding this section, we note that the Allen taxonomy is not the only one of use. Wahl and Rothermel [WR94] develop a set of twenty-nine interval relations, derived from an initial set of relationships between start and end points of media items. They then demonstrate how the twenty-nine may be reduced to ten generic relationships similar to the four just given. Keramane and Duda [KD96] present a set of relations similar to Allen's, but which also includes the notion of causality between media items. The temporal (non-causal) aspects of their relations can all be represented in *Mexitl*. Their work differs from ours in that they restrict composition of relations to functional composition whereas *Mexitl* uses a powerful set of logical connectives. Finally, the criteria stipulated by Buchanan and Zellweger [BZ93] for such temporal relations are satisfied by *Mexitl*; the details are beyond the scope of this paper.

3 More Complex Synchronization

The previous section dealt with synchronization conditions between the entire intervals over which the multimedia artefacts are displayed. We now turn our attention to cases of mutual synchronization of media items dependent on events or conditions occurring *within* the media items themselves. The occurrence of such events or conditions is, in general, assumed to be dynamic, depending upon the "run time" display of the media items. In particular, the synchronization events may arise from external stimuli, such as reader intervention, though this distinction is irrelevant to the taxonomy we present here.

3.1 The Model

We restrict ourselves to synchronization between pairs of media items. As is the case with the Allen set, relations between $n > 2$ media items are obtainable by composition. Rather than taking a symmetric view of synchronization, it is convenient to distinguish between the two media items in our taxonomy:

- media item a is being displayed
- media item b is to be displayed in synchrony with a.

We believe that this distinction corresponds more closely to an author's viewpoint; it certainly corresponds to the situation occurring in multi-authorship. Media items a and b are referred to as the "base item" and "synchronized item", respectively.

Our model is based on two considerations. The first issue concerns the classes of events in the base item a that should be available for synchronization, which we term *synchronization events*, refered to as *granularity* in [BZ93]. The second issue concerns the classes of items in the synchronized item b which should be so synchronized, which we term the *synchronization items*. Our model of synchronization possibilities will then be given by the Cartesian product

$$\textit{synchronization events} \times \textit{synchronization items}.$$

These events and items will be illustrated in the next subsection.

3.2 Synchronization Events and Items

We distinguish two broad classes of synchronization events which may occur in the base media item a. These are termed *temporal events* and *conditional events*. In [BZ93], the terms *predictable* and *unpredictable* are used. Temporal events refer to points or subintervals on the time line comprising the interval over which a is displayed. Such events may occur once or more than once. In the latter case, they may occur at regular or irregular time intervals. Examples include:

- five seconds after the movie starts — occurs once;
- every five seconds after the movie starts — occurs at regular time intervals;
- the first two seconds of the third movement — occurs once;
- at times generated by a random number generator — occurs at irregular time intervals.

Conditional events refer to conditions which may occur dynamically within the display of a. Again, they may occur once or many times, and may be single points or (sub)intervals.

- the first time the shark appears — single point occurring once;
- during the first appearance of the shark — an interval occurring once;
- the staccato notes — single points occurring many times;
- each individual crescendo passage in the third movement — intervals occurring many times.

Turning to the synchronization items which may occur within the item b, we distinguish between point items and interval items. The nomenclature should be clear, but we provide some illustrative examples:

- flash the screen red — a point item;
- display a picture for three seconds — an interval item;

Synchronization situations which might arise in specific multimedia documents are combinations of synchronization events and synchronization items, that is, instances of the Cartesian product referred to earlier. Examples, including some based on the events and items just listed, would be

– flash the screen red the first time the shark appears;
– keep a running count of the staccato notes;
– display a picture for three seconds for each individual crescendo passage in the third movement;
– five seconds after the movie starts, display a title for three seconds;

4 Synchronization Taxonomy

In this section, we consider in greater detail the classes of synchronization introduced in the previous section. We will give further examples of each. We will also indicate how each is represented in the *Mexitl* notation. We first consider in detail the various forms of events and items.

4.1 Synchronization Events

As our discussion in the previous section suggests, events have three attributes, each being one of a possible pair:

sort: *temporal* or *conditional* (T,C);
kind: *point* or *interval* (P,I);
number: *one* or *several*.

For convenience, we further divide the number attribute *several* into *regular* or *irregular*, corresponding to occurrences at regular or irregular times.

number: *one* or *regular* or *irregular* (O,R,I).

Moreover, an author may wish to concatenate a sequence of synchronization events to form a single synchronization event, which would occupy a non-contiguous subinterval of the original base item. An example would be "during the entire time the shark is on the screen", which is a single synchronization event constructed from several non-contiguous subintervals of the original. We therefore add the fourth attribute:

result: *separate* or *concatenated* (S,C).

We find it convenient to treat *separate* as a default, since it is the more usually occurring case of this attribute.

This taxonomy yields twenty-four distinct classes of synchronization events. We refer to each class using a mnemonic 4-tuple (or triple if we adopt the default just mentioned).

Examples:

[TPO]: three seconds after the movie starts
[CIIC]: during the time the shark is on screen
[CIO]: during the first viola solo
[TPR]: every three seconds
[CPI]: each staccato note in the first movement

Some of these twenty-four classes are less useful than others; in particular, the specification of the number attribute *regular* or *irregular* in the case of a conditional event may be useless over-specification. Further, the distinction between *separate* and *concatenated* in the case of an event with number attribute *one* is also questionable. Also, in the case of events with sort attribute *temporal*, the function of the item a is merely to establish a time line for the item b; beyond that, there is no synchronization, as such, between a and b. Further, in the case of events with kind attribute *point*, the point(s) must be one(s) which the authoring tool (in our case the formalism) can recognize, and which are of importance to the author in the application at hand. Usually, therefore, such points are intimately related to the substructure of the item a; for example, the start of each new level in a video game, three seconds before the end of each movement in a musical presentation, each new problem in a CAI application.

4.2 Synchronization Items

Our taxonomy of synchronization items, that is, the items b which are to be synchronized with the events just described, is rather simpler. Items have one of three possible sort attributes:

sort: *action*, *item*, *sub-item* [A,I,S]

Some remarks on each of these are appropriate.

action: In Section 2.1 and in [BCKT97a], we elaborate on actions in multimedia and in the *Mexitl* notation. The attribute *action* refers to any operation which can be accomplished in a single clock tick. We do not distinguish between an action which corresponds to a multimedia item, such as "flash the screen red", and one which performs some housekeeping, such as "add one to the running total".

item: The attribute *item* signifies the display of an independent media item which occupies an interval of length at least one, that is, at least two clock ticks.

sub-item: The attribute *sub-item* allows for a set of media items $\{b_i\}$, say, to be considered as sub-items (points or subintervals) of a composite media item $b = \{b_1, b_2, \ldots\}$. To see how the attribute *sub-item* differs from *action* and *item*, consider a synchronization event that has number attribute *regular* or *irregular*, meaning that the synchronization item is to be displayed several times. If the synchronization item has attribute *action* or *item*, the entire synchronization item is repeated multiple times. If the synchronization item b has attribute *sub-item*, then sub-items b_i are displayed in succession instead of repeating the entire synchronization item b each time. A synchronization item with attribute *sub-item* might be, for instance, the successive frames of a video clip, or successive portions of a looped video display.

4.3 Synchronization Taxonomy

The Cartesian product of the three classes of synchronization items with the twenty-four classes of synchronization events gives rise to seventy-two synchronization possibilities. While there are indeed seventy-two theoretical possibilities, two remarks are to be made. First, as we have pointed out, a number of these possibilities are unlikely to be of interest to an author. Second, we do not intend that an author need be aware of the details of this taxonomy, nor of precisely which possibility is being used in the particular situation at hand. Rather, our intention is to present the taxonomy as a reference model for authoring systems and for formal models. We terminate this section with a number of further examples, chosen to illustrate both our taxonomy and our formal notation. In the next section, we present one further more complex example.

First, by way of illustration, we classify each of the examples appearing at the end of Sect. 3, and represent each in *Mexitl*. We represent the classification of each example as a five-tuple (a four-tuple if the default result attribute *separate* is used). The last attribute is the sort attribute of the synchronization item.

- **Requirements:** flash the screen red the first time the shark appears
  Classification: [CPO;A]
  Specification: We introduce the operator **halt**, where $\mathbf{halt}(p)$ is true over an interval if the point property p is true in the final state (and not in any earlier state). Hence, we may use $\mathbf{halt}(\text{shark})$ to "swallow" all time before the first appearance of the shark and then flash the screen red.

$$\text{movie} \;\wedge\; \diamondsuit(\mathbf{halt}(\text{shark})\ \mathbf{;}\ \text{red})$$

- **Requirements:** keep a running count of the staccato notes
  Classification: [CPI;A]
  Specification: We introduce the operator **when**, where p **when** q is true if p is true over the interval of states in which the point property q is true. Hence, we may use the operator **when** to pick out all the staccato notes, which are treated as a single contiguous interval over which variable count counts up from 1.

$$\text{music} \;\wedge\; (((\text{count} = 1) \;\wedge\; (\text{count}\ \mathbf{gets}\ \text{count} + 1))\ \mathbf{when}\ \text{staccato})$$

  The classification and specification of "count the length of the violin solos" are similar.
- **Requirements:** display a picture for three seconds for each individual crescendo passage in the third movement
  Classification: [CII;I]
  Specification: First, we define condition cres, which determines whether a point is part of a crescendo by checking if the volume of the current point $(\text{vol} = x)$ is larger than the volume in the previous point $(\circleddash(\text{vol} < x))$ or less than the volume in the next point $(\bigcirc(\text{vol} > x))$.

$$\text{cres} \equiv (\exists x \leq \text{maxVol})\ ((\text{vol} = x) \;\wedge\; (\circleddash(\text{vol} < x) \;\vee\; \bigcirc(\text{vol} > x)))$$

Next we move forward to the first point of a crescendo using **halt**(cres), then move forward to the end of the crescendo while displaying a picture for the first three seconds. By applying chop star (*) to these steps, we loop through any following crescendo passages. The final **;** $\Box\neg$cres ensures that the repetition ends after the last crescendo passage by specifying that all points in the remainder of the movement must not be part of a crescendo.

$$\mathrm{mov}_3 \wedge ((\mathbf{halt}(\mathrm{cres}) \mathbin{\mathbf{;}} (\mathbf{halt}(\neg\mathrm{cres}) \wedge \diamondsuit(\mathbf{len}(3'') \wedge \mathrm{picture}^*)))^* \mathbin{\mathbf{;}} \Box\neg\mathrm{cres})$$

We assume for simplicity that the crescendo passages are longer than three seconds.

The classification and specification of "play the scary music each time the shark appears" are similar to these.

- **Requirements:** five seconds after the movie starts, display a title for three seconds

  Classification: [TPO;I]

  Specification: Some initial subinterval of the movie (given by $\diamondsuit$) is partitioned by chop into two pieces. The first piece, **len**($5''$), corresponds to the first five seconds of the movie. The second piece plays the title for three seconds.

$$\mathrm{movie} \ \wedge \ \diamondsuit(\mathbf{len}(5'') \mathbin{\mathbf{;}} (\mathrm{title}^* \ \wedge \ \mathbf{len}(3'')))$$

Second, we present some examples to illustrate more carefully the sub-item attribute within synchronization items:

- **Requirements:** Suppose we want to play a video of a tap-dancer over the staccato portions of a piece of music. The video is not restarted at each new staccato portion but continues where it left off.

  Classification: The result attribute is C (*concatenated*) because we wish to concatenate all the staccato notes into a single interval over which the video is played. The synchronization item has sort attribute *sub-item* in this case because the next frame of the video is played at successive staccato notes, that is, the video is not repeated in entirety at each staccato note. Overall, the classification is [CPIC;S].

  Specification: music $\wedge$ tap-dancer video **when** staccato
- **Requirements:** In a CAI application, display successive hints every three minutes until the student solves the problem.

  Classification: Again, because we are not repeating the same display, but are displaying a different hint every three minutes, the synchronization item has sort attribute *sub-item*. The entire classification is [TPRS;S].

  Specification: We introduce the *Mexitl* **while** structure.

$$(i = 0) \ \wedge \ \mathbf{while} \ (\neg\mathrm{solved}) \ \mathbf{do} \ (\mathbf{len}(3') \ \wedge \ \mathrm{hint}[i]^* \ \wedge \ (i \leftarrow i + 1))$$

5 A More Complex Example: Towards Lip-Synching

In this section we illustrate the ideas of the previous section by presenting a more complex example of media item synchronization. The example in question is the lip-synchronization problem. The specification of lip-synching, that is, synchronization of a video stream with an audio stream in an appropriate fashion, has been studied by a number of authors for various purposes. This "canonical" problem appears frequently as an illustration of real-time languages, such as ESTEREL [SHH92] and temporal LOTOS [Reg93]. Blair *et al* [BBBC95] use lip-synching to illustrate the notion of quality of service in a distributed system. Courtiat and De Oliveira [CD96] use lip-synching to illustrate a synchronization model based on RT-LOTOS. Here we are interested in using it to illustrate our taxonomy of synchronization in multimedia. In this example, we assume we have an audio stream a with which we wish to synchronize a video stream v. We consider an increasingly complex sequence of levels of the problem, where later versions are increasingly realistic representations of the lip-synching problem.

5.1 Level 1

In the simplest case, we ignore consideration of synchronization events in a. For instance, a might be a recorded sound track and v the video of a dubbed film (movie). In this case, the relations of Sect. 2 are sufficient. If it is known that a and v have equal length, we might use a equals v. If a is longer than v, we might use any one of a starts v, a finishes v, v during a or we could make use of one of the relations of Sect. 2.3 for precise timing.

If the relative lengths are unknown, we might wish to specify simply that a and v start together, the implication being that the longer continues after the shorter stops. In *Mexitl* this is written as

$$a \mid\mid v \;\equiv\; (\diamondplus a \;\wedge\; v) \;\vee\; (a \;\wedge\; \diamondplus v)$$

Alternatively, if the length of a is a multiple of that of v, we could scale the playback rate of v to conform to that of a. To do this, we use the **proj** operator: P **proj** Q is true over an interval which can be partitioned into subintervals such that P holds over each of the subintervals and Q holds over the interval constructed from the endpoints of the subintervals. Thus $\mathbf{len}(\hat{a}/\hat{v})$ **proj** v has the effect of inserting $\hat{a}/\hat{v}$ clock ticks between each frame of the video, and the synchronization in this case is

$$a \;\wedge\; \mathbf{len}(\hat{a}/\hat{v}) \;\; \mathbf{proj} \;\; v$$

5.2 Level 2

A common paradigm is one where a and v are composite items and are to be synchronized on a component by component basis. Hardman and Bulterman [HB95] describe a city tour of Amsterdam in which a sequence of audio segments describe a sequence of video pictures. In [BCKT97a] we describe a complex multimedia presentation of Beethoven's Fifth Symphony which includes the specification

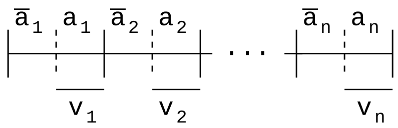
At the start of each movement, display an appropriate title for five seconds.

Both these may be regarded as a synchronization of class [TPI;S].

Suppose that a and v are composite items defined as follows:

$$a = (\bar{a}_1, a_1, \bar{a}_2, a_2 \ldots, \bar{a}_n, a_n)$$
$$v = (v_1, v_2, \ldots, v_n)$$

where the $\bar{a}_i$, a_i and v_i are media items, and the intention is that each v_i is to be synchronized with the corresponding a_i. Th following diagram illustrates the situation:

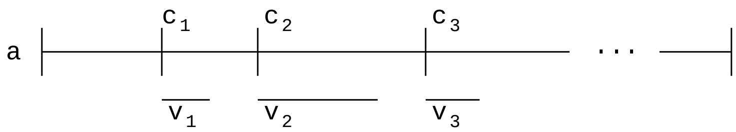


We give the *Mexitl* form of two possible synchronizations:

- Assuming that all the corresponding components a_i and v_i are of equal length:
$$\textbf{for } i := 1 \textbf{ to } n \textbf{ do } (\bar{a}_i \textbf{ ; } (a_i \wedge v_i))$$
- Assuming the corresponding components are of unequal length, we wish to display the components of a in sequence and display v_i as the corresponding component a_i starts.
If the relative lengths of the corresponding components of a and v are unknown, we can simply write:
$$\textbf{for } i := 1 \textbf{ to } n \textbf{ do } (\bar{a}_i \textbf{ ; } (a_i \parallel v_i))$$
This form of playback may cause a delay in the audio until a video segment is complete. If, on the other hand, we know in advance that all the a_i are at least as long as the corresponding v_i, we could adjust the play-rate of each component of v:
$$\textbf{for } i := 1 \textbf{ to } n \textbf{ do } (\bar{a}_i \textbf{ ; } (a_i \wedge \textbf{len}(\hat{a}_i/\hat{v}_i) \textbf{ proj } v_i))$$

5.3 Level 3

We now proceed to a level closer to a real representation of lip-synching by using a synchronization of class [CPI;S]. v is a composite item, say $v = (v_1, \ldots, v_n)$, and a offers a sequence of conditions $\{c_i\}$, where c_i triggers the display of the corresponding video segment v_i. We assume that the points in a in which the conditions c_i are offered are disjoint. This situation is a simplified model of lip-synching, where the display of a particular video segment might be synchronized to a particular sound segment in the audio. The following diagram illustrates the situation:



We consider the representation of this in *Mexitl*. The following formula specifies a(n) (sub)interval where c_i is true in the first state and v_i is displayed in an initial subinterval:

$$\textbf{beg}\ c_i\ \wedge\ \Diamond_{\mathrm{i}}\, v_i$$

The complete solution is therefore:

$$a\ \wedge\ \Diamond\textbf{for}\ i\ :=\ 1\ \textbf{to}\ n\ \textbf{do}\ (\textbf{beg}\ c_i\ \wedge\ \Diamond_{\mathrm{i}}\, v_i)$$

This solution would fail in the case in which v_i is longer than the interval between c_i and c_{i+1}. In such a case, we may wish to specify that as much as possible of v_i is to be displayed. Thus, we might write

$$\begin{gathered} a\ \wedge\ \Diamond\textbf{for}\ i\ :=\ 1\ \textbf{to}\ n\ \textbf{do}\ (\textbf{beg}\ c_i\ \wedge\ (x = \textbf{mylen}) \\ \wedge\ \Diamond_{\mathrm{i}}\,\textbf{for}\ j\ :=\ 1\ \textbf{to}\ \min(\hat{v}_i, x)\ \textbf{do}\ v_i[j]) \end{gathered}$$

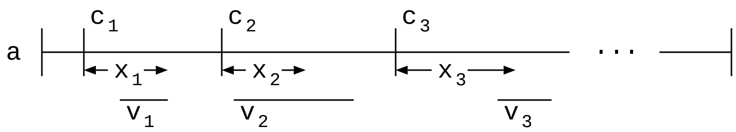
5.4 Level 4

Our final level gives a looser specification of lip-synching, which is related to the specifications of real-time synchronization and quality of service [BBBC94] that arise in distributed multimedia systems. Blair *et al* [BBBC95] present a distributed version of the problem, with one stream carrying video frames, the other voice packets. As the streams are synchronized, certain real-time constraints relating the two streams must be maintained so as to achieve acceptable lip-synching performance. We achieve this in our model by applying the revised temporal relations introduced in Sect. 2.3 to occurrences of the segments v_i in Level 3. We use *Mexitl* to illustrate what we mean, and make use of the construct $\textbf{lesseq}(E)$ which is true over any interval of length $\leq E$.

Consider $\textbf{beg}\ c_i\ \wedge\ (\textbf{lesseq}(x_i)\ \textbf{;}\ \Diamond_{\mathrm{i}}\, v_i)$. This specifies that the component v_i must begin at most x_i units after the condition c_i. Thus, the complete picture would be

$$a\ \wedge\ \Diamond\textbf{for}\ i\ :=\ 1\ \textbf{to}\ n\ \textbf{do}\ (\textbf{beg}\ c_i\ \wedge\ (\textbf{lesseq}(x_i)\ \textbf{;}\ \Diamond_{\mathrm{i}}\, v_i))$$

The following diagram illustrates the situation:



We have assumed that the video segment v_i is not too long for the interval $[c_i, c_{i+1}]$, even allowing for the possible delay of up to x_i units, and have used $\diamond$ to "pad" the remainder of this interval should v_i be shorter. We could also add scaling to the display of v_i, making use of a multiplication projection, as was done in Level 2. It is also possible for a more symmetric version of lip-synching to be represented, in which audio segments are also triggered by video segments, but this lies outside the scope of this paper.

6 Conclusions

We have presented what we believe to be a complete taxonomy of the possible internal synchronization situations between pairs of media items in multimedia documents. Our taxonomy is based largely on pragmatic considerations, and results in seventy-two such possibilities. However, as we have already suggested, some of the seventy-two possibilities are less likely to arise and to be of practical use in actual documents. We do intend that the taxonomy be used as a reference point for authoring systems and for formal models. We have established the representation of each possible relation in our *Mexitl* temporal logic, though the details of this are beyond the scope of this paper. Further, there is a close correspondence between points in the Cartesian product which yields the taxonomy and constructs in *Mexitl*. Thus, the taxonomy is useful knowledge for an author using a formal system such as *Mexitl* to specify synchronization relationships.

This work is a component of the authors' on-going project in the area of formal specifications of multimedia documents and on the study of authoring tools based on such formalisms. Our particular interest lies in documents with very rich sets of temporal relationships, where questions of consistency, prototyping, and modeling assume great importance. We expect that this work will be central in our ongoing design of a high level language for such authoring.

References

[All83] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11), November 1983.

[BBBC94] Howard Bowman, Lynne Blair, Gordon S. Blair, and Amanda G. Chetwynd. A formal description technique supporting expression of quality of service and media synchronization. In *Multimedia Transport and Teleservices International COST 237 Workshop, Vienna, Austria*, pages 145–167, November 1994. Appears as Volume 882, Lecture Notes in Computer Science, Springer-Verlag.

[BBBC95] Lynne Blair, Gordon Blair, Howard Bowman, and Amanda Chetwynd. Formal specification and verification of multimedia systems in open distributed processing. *Computer Standards and Interfaces*, 17:413–436, 1995.

[BCKT97a] Howard Bowman, Helen Cameron, Peter King, and Simon Thompson. *Mexitl*: Multimedia in Executable Interval Temporal Logic. Technical Report 3-97 (Kent), Computing Laboratory, University of Kent, 1997.

[BCKT97b] Howard Bowman, Helen Cameron, Peter King, and Simon Thompson. Specification and prototyping of structured multimedia documents using interval temporal logic. In *International Conference on Temporal Logic 1997*, 1997.

[BZ93] M. Cecelia Buchanan and Polle T. Zellweger. Automatic temporal layout mechanisms. In *ACM Multimedia 93*, pages 341–350, 1993.

[CD96] J.-P. Courtiat and R.C. De Oliveira. Proving temporal consistency in a new multimedia synchronization model. In *ACM Multimedia 96*, pages 141–152, 1996.

[Erf93] Robert Erfle. Specification of temporal constraints in multimedia documents using HyTime. *Electronic Publishing*, 6(4), pages 397–411, December 1993.

[HB95] Lynda Hardman and Dick C.A. Bulterman. Authoring support for durable interactive multimedia presentations. In *State of The Art Report in Eurographics '95, Maastricht, The Netherlands, 28 August - 1 September*, pages 119–143, 1995.

[JLSI97] Muriel Jourdan and Nabil Layaïda and Loay Sabry-Ismail. Time representation and management in MADEUS: an authoring environment for multimedia documents. In *Proceedings of Multimedia Computing and Networking 1997 SPIE 3020, San-Jose*, February, 1997.

[KD96] Chérif Keramane and Andrzej Duda. Interval expressions – a functional model for interactive dynamic multimedia presentations. In *IEEE International Conference on Multimedia Computing Systems (Multimedia '96)*, June 1996.

[Kin96] P.R. King. A logic based formalism for temporal constraints in multimedia documents. In *PODP 96*, September 1996. Revised version to appear in LNCS, Springer Verlag.

[KS95] Michelle Y. Kim and Junehwa Song. Multimedia documents with elastic time. In *ACM Multimedia, San Francisco, CA USA*, pages 143–154, 1995. Revised version to appear in LNCS, Springer Verlag.

[MHM96] Elina Megalou, Thanasis Hadzilacos, and Nikos Mamoulis. Conceptual title abstractions: Modeling and querying very large interactive multimedia repositories. In J.P. Courtiat, M. Diaz, and P. Sénac, editors, *Multimedia Modeling: Towards the Information Superhighway*, pages 323–338. World Scientific, 1996.

[Reg93] T. Regan. Multimedia in temporal LOTOS, a lip synchronisation algorithm. In *Proceedings of the 13th International Symposium on Protocol Specification, Testing and Verification (PSTV XIII)*. Elsevier Science, 1993.

[SHH92] J.-B. Stefani, L. Hazard, and F. Horn. Computational model for distributed multimedia applications based on a synchronous programming language. *Computer Communications (Special Issue on FDTs)*, 15(2), March 1992.

[SHT96] Timothy K. Shih, Lian-Jinn Hwang, and Jich-Yan Tsai. Formal model of temporal properties underlying multimedia presentations. In J.P. Courtiat, M. Diaz, and P. Sénac, editors, *Multimedia Modeling: Towards the Information Superhighway*, pages 135–150. World Scientific, 1996.

[WR94] Thomas Wahl and Kurt Rothermel. Representing time in multimedia-systems. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, pages 538–543, 1994.