Multimedia Object Models for Synchronization and Databases

Thomas D.C. Little  Arif Ghafoor
Department of Electrical and Computer Engineering Syracuse University Syracuse, New York 13244-1240

ABSTRACT

In this paper we propose a technique for formally specifying and modeling the temporal composition of multimedia data. The proposed model is based on Timed Petri Nets and the logic of temporal intervals. A strategy is presented for constructing a database schema to facilitate data storage and retrieval of data elements based on the inter-media timing relationships established by the proposed modeling tool. An algorithm is proposed which allows the retrieval of media elements from the database in a manner which preserves the temporal requirements of the initial specification. The proposed model accomplishes the specification of the synchronization requirements for complex structures of temporally related objects.

1 Introduction

A multimedia information service is characterized by multifarious service requirements. These systems must store, retrieve, and communicate complex representations of information under conditions of profuse heterogeneity. There are many potential multimedia applications. A typical example of a multimedia application is of an electronic catalog in which a "reader" can browse (or query) through a set of listed catalog items listening to audio descriptions, reading prices, and viewing video demonstrations. This simple example requires data of three types: audio, video, and numeric. Many other examples demonstrating presentation, manipulation, and communication of multimedia documents include [1], [2]. Characteristic of these systems is the need to compose, or integrate data of various types and origin into singular presentations.

Two forms of composition are apparent. Spatial composition involves assembling data based on positional parameters. For example, the composition of textual and image information. For this kind of composition the order of composition has no significance since there are no temporal relationships between the data elements and there are no time dependent data, such as video. For temporal composition there is an ordering assigned to the presentation of elements of the multimedia object which must be maintained via synchronization in time to provide functionality for a multimedia information service.

Composite representations of data to be integrated in a multimedia application are based upon three types of data: static, dynamic, and mixed. Static types can be electronic documents consisting of text and images arranged in various spatial orientations. Dynamic types include voice annotations, video images, or general animations. Mixed types are formed by uniting both static and dynamic types in a single composite presentation. The retrieval of data to compose a static object primarily involves spatial organizations of the components comprising the object, as in the case of still images in a textual document. Dynamic object composition requires additional consideration for temporal time constraints. Temporal relationships between the media may be implied, as in the simultaneous acquisition of voice and video, or may be explicitly formulated, as in the case of a multimedia document which possesses voice annotated text. In either situation, to properly schedule the synchronization of media with vastly different presentation requirements, the characteristics of, and relationships between media must be established.

The Office Document Architecture (ODA) [3] has been an effort to standardize the aspects of data integration with respect to spatial types, which we do not address in this paper. In [4] the types of temporal presentation control are summarized as sequential, concurrent or simultaneous (synchronous), and independent (asynchronous). Three levels of multimedia integration have been proposed [5]. These are the physical level, the service level, and the human interface level. At the physical level, data from different media are multiplexed over a single physical connection. The service level is concerned with the interactions between the multimedia application and the various media, and between the elements of the application. The human interface level is concerned with the presentation of the different media to the user.

The problem of multimedia synchronization has been addressed by various application environments [1], [2]; however, no formalism has been proposed for general description of multimedia synchronization. Our work in this paper differs from earlier works in two major aspects. We provide a formal specification for multimedia synchronization and for the retrieval and presentation of multimedia objects. We do not assume a particular application environment and generate a specific synchronization technique, rather, we propose an explicit representation which can be applied to various applications.

We indicate three contributions to the multimedia data integration problem. First, we propose a technique for the formal specification and modeling of multimedia composition with respect to inter-media timing. Next, we indicate a strategy for constructing a database schema to facilitate data storage and retrieval of media elements based on the temporal relationships specified through the proposed model. Third, we present an algorithm which indicates the retrieval of media elements from the database in a manner which preserves the temporal requirements of the initial specification. This is achieved through the constructed database schema.
2 Temporal Models

The presentation problem requires simultaneous, sequential, and independent display of heterogeneous data in a manner such that a user is not overwhelmed by the speed of delivery. For our model we seek an abstract synchronization scheme that describes the temporal relationships between data elements and facilitates their subsequent storage and retrieval. In the general case, a technique to model the real-world in a succinct way is sought. We choose Petri nets as our modeling tool, and indicate proper justification for our selection. Description of the temporal relationships is facilitated based on the concept of temporal intervals, which will be discussed in Section 4 for the purpose of building a database schema.

2.1 Petri Nets

For the presentation of data from multiple, independent sources we require a method to model both parallel and sequential composition activities. The Petri net [6] is chosen for a representation of the synchronization of multimedia entities. More specifically, we consider the timed [7], and augmented [8] Petri net models. These models are chosen for their desirable attributes of representation of concurrent and asynchronous events.

For simple Petri nets, the time from enabling a transition to firing is unspecified and indeterminate. Firing of a transition is assumed to be an instantaneous event. To represent the concept of nonzero time expenditure in the Petri net, extensions of the original model are required. A class of enhanced Petri net models have been developed which assign a firing duration to each transition [7]. These models are generally called timed Petri net (TPN) models. Another TPN model [8] represents processes by places instead of transitions. For this type of model, nonnegative execution times are assigned to each place in the net. The notion of instantaneous firing of transitions is preserved, and the state of the system is always clearly represented during process execution (tokens are at all times in places, not transitions). We choose this augmented model for its more compact representation of process interaction; however, both schemes are sufficient for our purposes. The augmented model is supplemented with resource information associated with TPN models for the purpose of illustrating the use of the multiple data sources.

In synopsis, we suggest a modification of earlier Petri net models, and call the new model Object Composition Petri Net (OCPN) for distinction from other Petri net models. The OCPN model augments the conventional Petri net model with values of time as durations and resource utilization on the places in the net. Formally, the OCPN is defined as a bipartite, directed graph $G = (T, P, A, D, R)$ where,

- $T = \{ t_1, t_2, ..., t_n \}$; set of transitions
- $P = \{ p_1, p_2, ..., p_m \}$; set of places
- $A: T \times P \cup \{ P \times T \} \rightarrow I, I = \{ 1, 2, ..., n \}$; set of directed arcs
- $D: P \rightarrow \mathbb{R}$; set of durations
- $R: P \rightarrow \mathbb{R}$; set of resources

A marked OCPN $C = (T, P, A, D, M)$ includes a marking $M$ which assigns tokens (dots) to each place in the net. Associated with the definition of the Petri net are a set of firing rules governing the semantics of the model. Since we define a transition to occur instantaneously, places rather than transitions have states. The firing rules are summarized as follows:

1. A transition $t_i$ fires immediately when each of its input places contain an unlocked token.
2. Upon firing, the transition $t_i$ removes a token from each of its input places and adds a token to each of its output places.
3. After receiving a token, a place $p_j$ remains in the active state for the interval specified by the duration $d_j$. During this interval, the token is locked. When the place becomes inactive, or upon expiration of the duration $d_j$, the token becomes unlocked.

Consider the following example to illustrate the use of the OCPN. A multimedia slide presentation is to be represented. The presentation consists of a sequence of synchronized audio and visual elements of varying duration. Assume that there are $n$ slides to present, and corresponding image and audio pairs to each slide. Time is indicated on the horizontal axis. Resources are described by a dimension on the vertical axis indicating multiple threads of presentation. We call this axis space. Using our Petri net representation, these activities are indicated by the Petri net indicated in Figure 2.

![Figure 1. Slide Presentation Timeline](image)

For each place we have assigned the required presentation resource, or device, and the time required to output the presentation data. The transitions in the net indicate points of synchronization, and the places processing. For example, upon completion of the first verbal annotation, place $t_1$, represented by talk1, unlocks its token. Because elements of the image (slide) and audio data streams have the same stage in a verbal annotation, the place $s_1$, indicated by slide1, unlocks its token synchronously with place $t_1$. The common transition fires immediately, allowing the next image/audio pair to be presented.

It is possible to represent arbitrarily complex synchronization with various degrees of granularity by using this technique. Frames of full motion video, delivered at 30 per second, can be isolated and synchronized to associated audio segments in an OCPN structure as in the previous example. This is achievable with, for example, videotex technology, which permits access of individual frames of full-motion video.

We have indicated a methodology for specifying related presentation processes associated with different media, considering sequential and parallel relations; two elements of a larger class of relations on temporal intervals, which are defined in the next section. For a general process model that represents all potential multimedia scenarios we draw from the field of time modeling.

2.2 Temporal Intervals

Temporal information has been modeled in information systems at the conceptual level in many studies. [9] provides a survey of this field. Two principle applications of this research are historical database maintenance and inference generation in temporal systems. Query languages have been developed for reference to temporal information, for example TQel [10] and Time-by-Example [11].
Hamblin [12] presents a logic of intervals which is very useful in the development of a synchronization scheme. Given any two intervals, there are thirteen distinct ways which they can be related. These relations indicate how the two intervals relate in time; whether they overlap, abut, precede, etc. Using the representation of [13], these relations are indicated graphically by a timeline representation of Figure 3(a). We only show seven of the thirteen relations since the remainder are inverse relations (for example, after is the inverse relation of before). For inverse relations, given any two intervals, it is possible to represent their relation by using the noninverse relations only by exchanging the interval labels (the equality relation has no inverse).

In Figure 3(a) we show intervals as processes $P_a$ and $P_b$ with durations $z_a$ and $z_b$, respectively. One axis indicates the time dimension, and the other space, or resource utilization, as previously discussed. We define an atomic process to be one which is cannot be decomposed into subprocesses, as in the case of the presentation of a single frame of a motion picture.

We present the following theorem relating temporal intervals to Object Composition Petri nets.

**Theorem 1:** Given any two atomic processes specified by temporal intervals, there exists an OCPN representation for their relationship in time [14].

Let $P_a$ and $P_b$ be atomic processes with finite, nontrivial (nonzero) temporal intervals $z_a$ and $z_b$, respectively. Let $\tau_R$ be the finite delay duration specific to any temporal relationship $T_R$. We construct a Petri net for each relation, indicated in Figure 3(b). A delay process and place is introduced depending on the type of temporal relationship specified. For the meets, starts, and equals relation, $\tau_R = 0$. For the overlaps, during, and starts relations, the model does not explicitly indicate the delay of the process which completes first. Since both places (processes) lead to the same transition, the Petri net requires that tokens from each be unlocked prior to the execution of the succeeding process. Thus the Petri net assures synchronization for these cases.
The atomic presentation processes are directly mapped to Petri net places, and additional delay processes are associated to facilitate the correct timing relationships. Using the hierarchical modeling property of the Petri net we can replace facilitation of a Petri net by equivalent abstract places determined by the durations of subprocess transitions, and additional delay processes are associated to two processes and their temporal relationship we can build a Petri net. Since each process is represented by a subnet in the net, we can say that subnet \( SP_N \) can be replaced by a place \( P_a \). Thus we allow a subnet to replace an equivalent abstract place, and vice versa. The following lemma characterizes the replacement of subnets by equivalent places:

**Lemma 1:** The duration, \( \tau_a + 1 \), of process \( P_{a+1} \), can be uniquely determined by the durations of subprocesses \( P_a, P_{a+1} \), the duration of the temporal relation between them, \( \tau_{p,y} \), and the temporal relation between them, \( \tau_{p,y} \).

For the sequential cases of \( T_{p,y} = 1 \), before and meets, duration \( \tau_{a,y} = \tau_{a} + \tau_{p,y} + \tau_{b,y} \). For overlaps, \( \tau_{a,y} = \tau_{a} + \tau_{p,y} + \tau_{b,y} \).

For the remaining parallel cases, \( T_{p,y} = 1 \), during, starts, finishes, and equals, \( \tau_{a,y} = \max(\tau_{a}, \tau_{b,y}) \).

Using subnet replacement, and Lemma 1, we can state the following theorem for the construction of process models.

**Theorem 2:** An arbitrarily complex process model composed of temporal relations can be constructed with Petri nets by choosing pairwise, temporal relationships between process entities [14].

### 3 Database Architecture

The objective for which we present Petri nets is the specification of multimedia presentation process hierarchies indicated by temporal relationships of any type. For a multimedia information storage and retrieval service, we must maintain the information necessary to synchronize any pair of temporally related multimedia objects. This information clearly consists of the temporal relation, \( T_{p,y} \), and the temporal intervals, \( \tau_{a} \) and \( \tau_{b,y} \), between process entities, as we have demonstrated. Given this information, we can completely construct the synchronized process interaction, modeled by a Petri net. Recalling the slide presentation example, process interaction is completely specified except for the temporal relations. For this example, between corresponding audio and image components the temporal relation \( \tau_{a} \) is valid, with equivalent values for \( \tau_{b} \) and \( \tau_{b,y} \). Between sequential image/audio pairs the relation \( \tau_{a,y} \) holds with time values \( \tau_{a} \) and \( \tau_{b,y} \) corresponding to the durations of successive pairs of elements.

#### 3.1 Database Schema

The Petri net model described in Section 2 allows us to build or describe complex presentation processes. For this reason, a hierarchical database schema is selected. A hierarchical representation is natural for a logical database structure indicating pairs of entities with associated temporal intervals and relations. For this structure, we allow three tree node types which facilitate the activities of multimedia objects. These tree types are called terminal, nonterminal, and meta.

A terminal node type, indicated in Figure 4(a), has attributes which indicate node type (terminal, nonterminal, or meta type), media type (text, image, video, etc.), and an additional unspecified attribute, and a pointer which indicates the location of the data for presentation. This node type indicates the elements at the finest grain for synchronization. Nonterminal nodes have a different structure as indicated in Figure 4(b). Attributes include type, an unspecified attribute, left and right child pointers, and temporal data (\( \tau_{a}, \tau_{b}, \tau_{b,y} \), and the temporal relation, \( T_{p,y} \), between the children).

![Figure 4](image)

Two node types allow complete specification of the synchronization requirements, as indicated by the OCPN model. However, there is a benefit to describing a third structure for the description of a database schema. There are applications in which it is not desirable to retrieve the entire object hierarchy upon access of the root element. Additionally, there may be a large amount of redundant information in a database with implied, pairwise, binary tree representation. To allow any number of child nodes with a single temporal attribute, we use a meta node. These nodes, indicated in Figure 4(c), possess attributes like the other node types. Meta nodes associate many temporal intervals with a single temporal relation. They allow a compact representation of a common relation among many similar process intervals. In Figure 5 we indicate the association of many slide-talk pairs with a single meta node, sharing the temporal relation \( \tau_{a} \). In this manner the number of nodes can be reduced from a potential \( 2n - 1 \) to \( n - 1 \).

A database schema is described by composing a tree structure based of the temporal interrelationships of the various components to be synchronized, and the intended functionality of the retrieval operation. The OCPN is used to relate synchronization constraints between data elements at the desired level of synchronization. Once the model is created, temporal relations can be established with associated timing values. A database is built by assigning data to tree structure of the database schema. Time durations at the terminal or presentation element level must be identified and combined based on temporal relationships specified by the real-world and subsequent
Given a database based on our schema construction technique containing only consistent markings with respect to timing, we propose an algorithm which presents multimedia data elements based upon the OCPN used for schema creation. The algorithm builds a process network which indicates the original Petri net model used to specify the multimedia presentation. The algorithm is recursive and can be applied to a schema of arbitrary complexity. The basic algorithm is:

```
built-process-tree(Obj: object)
begin
  fetch_attributes(Obj);
  if Obj.node_type = nonterminal then
    if Obj.temporal_relation in (before, meets) then
      do begin /* sequential */
        build_process_tree(Obj.left_child);
        delay(t);
        build_process_tree(Obj.right_child);
      end
    else /* overlaps, during */
      create_process(build_process_tree(Obj.left_child);
      create_process(build_process_tree(Obj.right_child);
      delay(t);
    end
  else /* parallel */
    create_process(build_process-tree(Obj.left_child));
    create_process(build_process-tree(Obj.right_child);
    end
end
```

Since the formulated database schema contains temporal relations, the algorithm can identify the nature of the processing concurrency. The relations belong to the sequential or parallel categories. The algorithm generates a processing structure which possesses a thread for each presentation element whether sequential or parallel. We use a procedure, `create_process`, which generates a new thread of execution, while the parent continues to execute (we assume that parent processes continue to execute until all of their children have terminated). With the algorithm we build a Petri net with the desired sequential and parallel structure, and with observance of the time requirements for terminal data elements. For uniformity in the parallel relations, the inverse relations for overlaps and during are used. This allows for a simpler algorithm without loss of functionality. As per the Petri net model, we let the right child represent process $P_r$, and the left child represent process $P_l$.

Pairs of processes are synchronized within a call to `build_process_tree`. Since the parent cannot terminate until all of its children have terminated, the wait and subsequent termination of a call to `build_process-tree` represents the waiting and firing of a transition in the Petri net model in which two arcs join. Process waiting is illustrated in Figure 6.
The presented algorithm only considers terminal and non-termin
tinal nodes. Meta nodes allow groups of related temporal hierarchies to be presented with a common temporal attribute. This node type facilitates repetitive retrieval of temporal structures. Out-degree of this type of node can be greater than two, permitting one-to-many relationships necessary for network or hierarchical database schemata. Components of a meta node are all related by the same temporal attribute, for example, meetings or equals. We permit meta nodes to eliminate the construction of very large binary trees, as discussed in Section 3.1.

A minor modification to the algorithm allows for the correct spawning or sequencing of the necessary child processes indicated by a meta node, not presented for lack of space.

The algorithm presented facilitates the retrieval and subsequent presentation of stored multimedia entities. Specifically, the algorithm builds a process network which replicates the original Petri net model used to specify the multimedia presentation, assuming a consistent database with respect to timing. In Section 5, we develop a database schema for which the algorithm is applied.

5 OCPN Demonstration

In this section, we present a detailed example which elaborates the ideas and models presented in the preceding sections. The example contains both static and dynamic data in the form of audio, image and textual data which require synchronization at the presentation level. The Anatomy & Physiology Instructor is a simple multimedia application example based on the hypermedia paradigm and temporal relation specification. Figure 7 indicates the coarse grain hypermedia network for this example. We assume that arcs in this figure represent links between informational units, represented by nodes, which contain lessons. Beginning at the Start node, users may browse via link selection or formulate explicit queries against the overall database.

When browsed or selected, information at a node is activated, causing its subsequent retrieval and presentation. For example, browsing Alveolar Ventilation assuming an initial starting point of Respiration, causes the information associated with that node to be presented. Fine grain synchronization is performed within the informational units.

A database structure is developed at two levels. At the coarse grain we desire the ability to select information units such as Heart or Kidneys. At the fine grain the synchronization constraints must be specified for concurrent presentation of various media elements. Each information unit can have different presentation requirements requiring distinct synchronization specification. Ideally, one specification suffices for all information units, simplifying the task of presentation design. For the general case, all nodes have different characteristics. We describe the synchronization requirements for one information unit, Heart Muscle.

To specify the synchronization requirements within an information unit a model for the presentation of information is required. For the example, the model used is based on the sample workstation screen indicated in Figure 9. In this figure the informational unit Heart Muscle is presented. Note the different regions associated with the various media types including audio, video, text, image, and animated image ("Views"). Let us assume some characteristics and relationships between elements for this example. Figure 8, indicates possible relationships in time of the various elements of the example. This timeline representation shows that the textual component, the icon, and the "Location in body" images should persist for the duration of the informational unit presentation. The views are specified to change with time to provide a gradual rotation, or animation, of the organ selected, while the video and associated audio components begin presentation after a constant delay of \( \tau_{12} \). On the spatial axis are indicated the resources \( r_1 \) through \( r_6 \), associated with the workstation screen regions and the audio output.

Figure 6. Process Waiting

Figure 7. Anatomy & Physiology Instructor Information Network

Figure 8. Anatomy & Physiology Instructor Timeline

Synchronization of the media elements is specified by the OCPN model. By combining elements of the presentation in temporally related pairs, we create a OCPN for the information unit shown in Figure 10. This OCPN indicates the processing required for the presentation of a single Anatomy & Physiology Instructor information unit. Between transition \( t_1 \) and \( t_2 \) we see a maximum of eight threads of presentation corresponding to the timeline of Figure 8. At the outset, seven threads are created as indicated by the initial transition, \( t_1 \). Places \( P_{16} \)
through $p_{19}$ represent static data which is active for the duration of the presentation. The two interacting threads, $p_1, p_2, p_5, p_7, p_9, p_{11}$, and $p_2, p_4, p_6, p_8, p_{10}, p_{12}$, represent the animation of two views of the organ, synchronized at points indicated by $t_1, t_2, t_2, t_3, t_4$, and $t_5$ of the OCPN. The places $p_{14}$ and $p_{15}$ represent dynamic audio and video data, delayed by $\tau_{13}$.

Having specified the synchronization requirements of an informational unit, we create a network database schema to store the medical data and temporal relationships between data items. This structure is built directly from the pairs chosen in OCPN development. Choosing View1 and View2 pairs based on the temporal relation $equals$ forms a nonterminal node of the database schema, which we call Organ-Horizontal-Vertical-View. Text elements, still images, and video/audio are similarly combined using the $equals$ relation generating nonterminals node Organ-Text-Info, Organ-Icon-Location, and Organ-Video-Image, respectively. These nonterminal nodes are related by further pairing and temporal relation assignment. Organ-Horizontal-Vertical-View is appended to a meta node; Organ-Views, with temporal attribute $meets$, allowing representation of multiple image pairs. Relating the nonterminal nodes results in additional nonterminal nodes. Organ-Image relates Organ-Views and Organ-Video-Image with the $finishes$ relation, accounting for time delay $\tau_{14}$. Similarly, Organ-Text-Icon is related to Organ-Image by Organ-Info-Group. Finally, Organ, a meta node, represents the set of organs with a similar schema for which Heart Muscle is a member. Terminal nodes, not specified here, indicate the exact location of stored data instances within the medical database. The resultant schema, presented in Figure 11, is composed...
with a meta node at its root to facilitate selection via query or hypermedia browsing. In this case we assume that all organs have a similar temporal synchronization requirement.

In assigning data to the database schema we must maintain temporal consistency. According to our OCPN model, based on the pairings we have chosen and the temporal relations used, we have a set of consistency requirements: duration of unit = \( t_1 + t_2 + t_3 + t_4 + t_5 + t_6 + t_7 + t_8 + t_9 = t_{15} + t_{16} + t_{17} = t_{18} + t_{19} + t_{20} + t_{21} + t_{22} + t_{23} + t_{24} + t_{25} = t_{26} + t_{27} + t_{28} + t_{29} = t_{30} \). By storing data which satisfy these constraints, the system performs as specified by the OCPN, without out delays caused by inappropriate values.

6 Concluding Remarks

Presented in this paper is an approach to object storage and retrieval for multimedia synchronization. The proposed OCPN scheme needs to be examined from the viewpoint of functional, object, and access-oriented paradigms. These methods are expected to reveal an elegant approach to interfacing general multimedia services to databases.

Recapitulating, we have proposed the tools and strategy for the formal specification of multimedia object composition with respect to synchronization. Using a form of timed Petri net, called a OCPN, multimedia objects can be choreographed in time and stored in a database indicated by a schema developed using the OCPN. This database is utilized as a storage element supporting multiple access techniques including conventional query systems and hypermedia. We have presented examples to support our models, indicating the usefulness in building multimedia information services.

7 References


Acknowledgement

This research was supported in part by the NYNEX Corporation through the New York State Center for Advanced Technology in Computer Applications and Software Engineering (CASE) at Syracuse University.