PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON INTERNET COMPUTING

IC'04

Volume I

Editors:
Hamid R. Arabnia
Olaf Droegehorn

Associate Editors:
Zhixiang Chen, Ping-Tsai Chung,
Hoda El-Sayed, Mao Lin Huang, Peter Langendoerfer,
Joan Lu, Xiannong Meng, Youngsong Mun, Anthony Scime

Las Vegas, Nevada, USA
June 21-24, 2004

A Model for Integrating Deterministic and Asynchronous Events in Reactive Multimedia Internet Based Languages

Angela Guercio
Department of Computer Science
Kent State University, Kent, OH 44242

Arvind K. Bansal
Department of Computer Science
Kent State University, Kent, OH 44242

Abstract

Current Internet based multimedia modeling systems and languages lack visualization based upon integration of asynchronous events and deterministic synchronization and triggering based upon such integration. Modeling interactive and real world systems needs such integration. The paper introduces a modeling technique for reactive distributed multimedia system based on triggers, active events and transformers. A scalable system of multimedia clocks (time bases) associated with streams is defined to control the synchronization of streams. Context-related reactions to context-related situations are detected with the use of an active persistent repository to take care of asynchronous events, as well as reactions to the external world.

Keywords: Asynchronous events, Internet based languages, multimedia, synchronization, reactive systems modeling.

1. Introduction

In the last few years, Internet has become pervasive, and the demand for more realism in multimedia based visualization is continuously increasing. In order to achieve this realism, multiple efforts in multimedia transmission [6], multimedia modeling standards [7] and multimedia rendering languages [3] are taking place.

When using distributed reactive multimedia systems, a number of multimedia streams and aperiodic signals are produced in one local or remote location, and are consumed in another location. Multiple streams are synchronized with each other, may be transformed, and the system may react to asynchronous events. An aperiodic signal can interact with other streams or signals, transform a stream (or group of streams), and trigger another chain of events. Media types can react to external stimuli (user intervention) or their own content, or to some other media type content. The repetition of the same actions does not guarantee the same reaction due to the change in context, past events, or the order of events.

Current modeling techniques are limited to such model as ER diagrams and UML, and current synchronization techniques are limited to languages such as SMIL 2.0 [16]. These techniques have very limited integration of multimedia reactivity, computability, event based triggering, dynamic altering of multimedia attributes, and synchronization of multiple streams. These models do not support asynchronous events, loose ordering of events, and automated dilation of time-scale in a group of streams to preserve synchronization. In order to model a real world phenomenon, this integration is essential.

This paper introduces a new formalism integrating asynchronous and synchronous events, deterministic events and triggers and stream transformation for modeling reactive distributed multimedia systems. A multimedia stream is considered to be a particular type of periodical signal. Both periodic and aperiodic signals (along with the associated media content) can cause a trigger to generate an event, one of whose outcomes could be the activation of a synchronization or transformation operation on multimedia streams. Clocks associated with streams are defined to control their synchronization. Context-related reactions to context-related situations are detected by an active repository, as well as reactions to the external world. The use of a persistent repository takes care of partial conditions and the change in the order of nondeterministic conditions since the truthfulness of partial and nondeterministic conditions can be archived in the repository.

The major contributions of this paper are:

(1) The model allows the integration of asynchronous events with deterministic multimedia events, and allows multimedia event based triggering;

(2) The model uses active repository to relax the strict order of occurrence of asynchronous events and allows for the accumulation of partial conditions;

(3) The history in the active repository allows the model to be used in complex reactive multimedia systems where the reaction can be triggered using past and present signals;

(4) The model allows dynamic scaling of the multimedia stream which automatically allows synchronization.

The paper is organized as follow. Section 2 provides the basic definitions for modeling distributed reactive multimedia systems. Section 3 describes the overall model, the notion of triggers, the persistent active repository, and the transformers. In Section 4 we discuss the synchronization issues. In Section 5 we provide an overview of related work. We conclude with a discussion of our results and future research.

2. Multimedia streams and media time.

A multimedia stream is a sequence of tuples. The elements of the tuples can be either tuples or values. A multimedia stream can be:

(1) a *continuous stream* as produced by sensors

(2) a *periodic stream* where data is associated with a periodic signal, or

(3) an *aperiodic stream* where the data is associated with an aperiodic signal generated by an event or external interaction.

Definition 1: A *periodic stream* $\mathcal{S}^p$ is a sequence of elements associated with periodic signals. $\mathcal{S}^p_i$ is the i-th element in the sequence such that the period *p* does not vary, that is the time between $\mathcal{S}^p_{(i+1)}$ and $\mathcal{S}^p_i$ is the same as the time between $\mathcal{S}^p_{(j+1)}$ and $\mathcal{S}^p_j$ for any i,j where $i \neq j$.

Definition 2: A *continuous stream* is data produced continuously by a sensor. A continuous stream can be modeled as a periodic stream with periodicity $0<p<\varepsilon$ where ε is a small value.

Continuous streams play a major role in real world systems. For example, one could transmit a stream of temperatures or a sequence of object movements taken from a surveillance camera.

Definition 3: An aperiodic signal can be generated at any time either by an external stimulus or by an event generated after a computation. An *aperiodic stream* $\mathcal{S}^a$ is a sequence (possibly of length one) of aperiodic signals. In an aperiodic stream $\mathcal{S}^a_i$ represents the i^{th} aperiodic signal.

Unless a distinction between periodic and aperiodic is necessary we will use the term multimedia stream denoted by the symbol S.

Definition 4: A *multimedia stream S* has two components: *attribute-set* and *data.* Three attributes *periodic* or *aperiodic*, number of data elements per unit time, and type of data (such as *audio* or *video* or *music or audiovisual* etc.) are essential. Other attributes are specific to the streams, and vary with different types of multimedia streams.

Example 1: The data for the audio stream is a sequence of sampled packets with the following attributes:

(a_0 = periodic, a_1 = audio, a_2 = 44100 samples per second, a_3 = no. of channels = 4, a_4 = 16 bits per sample, a_5 = media length, ...)

A *common clock* provides a common *time-base*, to which all the associated streams relate for implicit synchronization [8]. A point in the time-base increases monotonically. The *time-base-start-time* is a point in *time-base* where a media stream starts. The *media-start-time* is the point in the stream where playback begins. The *media-time* is the current point in stream-rendering.

The *playback rate* R is a scalar value which affects the rendering speed of media. If $R>1$ the media plays faster than normal; if $0<R<1$ the media plays slower than normal; if $R < 0$ the media plays in reverse.

A clock *c* is associated with one or more multimedia streams. A stream S_i, has a time-base-start-time *tbt-start*(c_i) a media start time *m-start*(c_i) and a playback rate $c(R)_i$.

Definition 5: A *group* $\mathcal{G}$ is defined recursively as:

(i) A single multimedia stream such that the time-base of $\mathcal{G}$ is same as the time-base of the stream.

(ii) Two or more multimedia streams sharing a common time base or with their time-bases related through an equation for proper synchronization.

The condition to relate time-bases for synchronization is necessary but not sufficient since it is not necessarily true that all the streams that share a common time-base are grouped. Logically, a group can be seen as a tree in which the interior nodes of the tree are groups, and the leaf nodes are multimedia streams. Groups can be used to perform operations on sets of related multimedia streams. Operations on multimedia streams (e.g. scaling of time-base) can be applied either to a multimedia stream in isolation or to all of the streams in a group of which the stream is a member. We will call the first type of operation *isolated* and the second type *synchronized.*

Multiple media stream rendering can be synchronized either by sharing a common clock, sharing a common event, or through the use of an intermediate media point which acts as a rendezvous point. The use of common clocks gives very strict synchronization. However, its use is limited when handling synchronization of multiple events.

3. Modeling distributed reactive systems

Distributed reactive multimedia systems must be able to react when certain conditions are met. The reaction of the system consists of the generation of one or more events that "respond" appropriately to the presence of some previous phenomena.

Distributed multimedia systems have been modeled as reaction graphs where the media generation point is modeled as a source, the media rendering point is modeled as a sink, transformers which change the attributes of media, triggers that selectively control the media streams and transformers, and an active repository that samples the media stream, analyzes the media content, matches the media stream against the required conditions, and transmits the outcome to the triggers. A multimedia stream *S* with an associated clock $c \in C$ forms a *multimedia source* $\mathcal{S}rc(S, c)$. Sources can be either local or remote. Each multimedia signal is generated by a source. A *player* is associated with zero or one periodic streams. The state of a player is 'Play' or null. A player with a null state is called a *sink.*

If the reaction requires a computation which involves the input streams, the streams are passed to one or more transformer modules.

A *phenomenon p* is a set of Boolean conditions connected through Boolean operators which have to be true for an event to occur. An *event* E caused by a phenomenon *p* has a destination point $E_p(\mathcal{D})$. A *generic event* has no specific destination. A destination is a module (such as a procedure or a transformer) where a reaction is performed.

The conceptual execution model is shown in Figure 1. In the figure a source is represented by a circle. The output of the trigger is depicted by dashed lines, and the input streams are depicted by solid lines. The arrow between trigger and active repository describe the communication between the two modules.

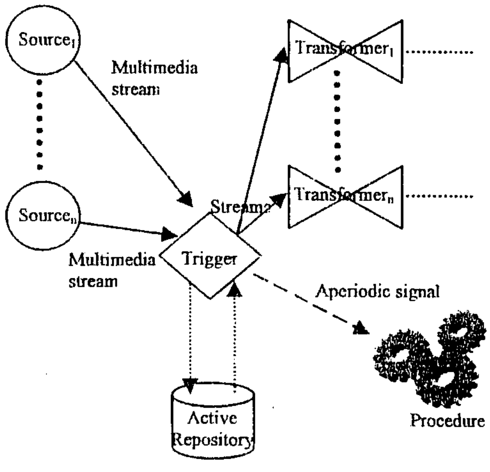

Figure 1 – The conceptual execution model

3.1 Triggers

Triggers provide a general mechanism for actions after a set of conditions are met. A trigger is associated with sets of multimedia stream-groups and a set of conditions. A trigger is modeled as a triple of the form (*trigger-name*, MS, D), where MS is the set of associated multimedia streams, and D is the set of destinations for the events associated with the trigger.

Triggers can activate events in response to the satisfaction of a set of conditions. A trigger reaction may be monitoring of external sensors, transforming streams, redirecting a stream to a different destination, starting a new thread of computation which may modify the attributes of multimedia stream, or starting another event resulting in a cascade of triggers and events.

If the reaction requires a computation which does not involve the input streams, the trigger generates an aperiodic signal which causes a procedure-call associated with the activated event, shown by gears.

In a real-time distributed reactive multimedia environment, we can distinguish two types of triggers: *event-based triggers* and *periodic triggers*. Event-based triggers fire when some constraints (or conditions) are satisfied. For example, the alarm activation of a surveillance camera fires a trigger when a human figure is detected. Periodic triggers fire periodically. For example, a biologist capturing the blooming of a flower will capture and transmit pictures periodically. Triggers can also be classified as *continuous* or *discrete*. *Continuous* triggers do not require resetting, while *discrete* triggers need to be reset every time they go off.

Triggers are associated with zero or more streams and zero or more aperiodic signals. Each stream may meet a partial condition (or a phenomenon). In order to store partial conditions and allow nondeterministic order of conditions in multiple streams, the trigger is in communication with a persistent active repository.

3.2 Active repository

The *active repository* is a persistent active database contains disjoint sets $\mathcal{C}_1 \oplus \mathcal{C}_2 \oplus \ldots \oplus \mathcal{C}_n$ of Boolean conditions (patterns). Each set of Boolean conditions can activate the associated trigger. The word "active" indicates that the database continuously processes the incoming media streams and aperiodical signals, and archives the information used by the associated trigger to verify the required conditions. The processing performed in the active repository includes content analysis of media, pattern detection, and resetting the partial conditions if time constraints are violated. The processing in the active repository does not change the original stream or original data that have been sent to the repository but rather analyzes those data to extract other information. The active repository sends an internal asynchronous signal to the trigger every time a partial condition becomes true. Based upon the signal from the active repository the trigger checks for the whole condition to be true.

The history of the analysis of incoming data as well as the content based analysis of the media data is kept in the repository. This history can be used either by the repository or the trigger if the historical information is part of the condition. Due to the presence of persistent history, the nondeterministic order of partial conditions and aperiodic signals can be handled. The repository also contains the associated temporal constraints to reset the analysis in case all the conditions are not met in a constrained time. The active repository can be interrogated and reset by the trigger.

3.3 Transformers

Transformers change the values of the attributes of the stream and the data stream by applying the appropriate transformation functions. Transformers can explicitly change the rate of the periodic stream or multiplex streams or reduce the number of channels of a video for rendering purpose, or implicitly change the rendering rate in a group of streams to maintain synchronization.

Definition 8: Given a periodic stream $\mathcal{S}^p$ with attributes $(a_0,\ldots,a_n)$ and data d and an event E, a *transformer* $T(\mathcal{S}^p, f, E)$ produces a periodic stream $\mathcal{S}^{p\prime}$ with attributes $(a'_0,\ldots, a'_n)$ where $a'_i = f_i(a_0,\ldots,a_n)$ with $0 \leq i \leq n$ and data d' where $d' = f_{n+1}(d)$. f is the *transformation function* of T connected to the event E, with $f = f_0 \lozenge f_1 \lozenge \ldots \lozenge f_n$

$\Diamond f_{n+1}$ where $\Diamond$ is the concatenation of functions. When $E = \mathcal{E}$ the default transformation is an identity transformation.

A stream can undergo many transformations in sequence before sending the stream into a sink or into a sink/source. The resulting stream can be redirected to one or more destinations (see Figure 2).

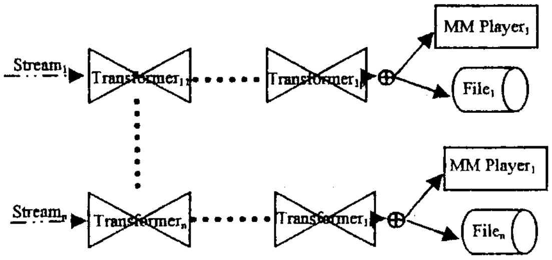

Figure 2 – Application of transformers

4. Synchronization

The synchronization process consists of a set of spatio-temporal functionalities that enables rendering of animated multimedia objects in multiple streams to realize the same perception as if it happened in real time.

The *inter-media synchronization* involving multiple media streams requires the ability to relate the clock of each stream either through a common shared time-base or through an equation relating the two time-bases. If two multimedia streams are grouped together then changing the time-base of one stream affects similarly the time-base of the other stream to maintain synchronization.

Given streams $S_1, S_2, \ldots, S_n$ in a multimedia group $\mathcal{G}$ and a *synchronization* function f, the application $f(S_i)$, $1 \le i \le n$, applies the function f on all the streams $S_1, S_2, \ldots, S_n$ such that the synchronization constraint is maintained.

For example, during lip synching audio and video streams are played in a lock step manner to give a realistic perception. The two streams are grouped so that time-scaling on either stream (e.g. time stretching or compressing the video) causes the other stream to be time-scaled to maintain synchronization.

On the other hand, at times it may not be desirable to affect all of the streams in a group. Imagine a lecture consisting of a sequence of slides synchronized with an audio stream. A viewer of this lecture might want to speed through the slides in order to get an idea of the contents of the lecture. In this case, isolated scaling on the slide sequence is performed and the synchronization of the two streams is broken.

As explained above, the process of grouping changes dynamically and is event driven which means that for certain events a set of multimedia streams may be grouped together while for another event the same set of streams need not be grouped. This dynamic grouping can be handled either by:

(1) declaring multiple orthogonal groups (groups not affecting each other) statically and associating different groups with different events, or

(2) temporarily restricting the group property to isolate a subgroup from being affected.

4.1. Synchronization and triggers

In order to activate triggers, pre-defined conditions should be true within a constrained time (possibly indefinite). The trigger tests the presence of the aperiodic signal together with the presence of the periodic signals to select an appropriate reaction. Since aperiodic signals can come at any time, triggers synchronize aperiodic signals with other signals for the next cycle using a common clock. The arrival of an aperiodic signal in the repository might cause some conditions to be true which causes a signal to be sent to trigger. Conditions in the active repository are time-constrained. While checking for overall condition if the trigger finds that a condition is past its temporal constraint, the condition is reset to the original state.

4.2 Synchronizing distributed streams

In a single system a common physical clock can be used as a common time-base for synchronization. However, this is not possible in distributed system with multiple input multimedia streams coming from different distributed nodes due to physical clock differentials and clock drifts. Clock drifts causes *skew* (defined as the difference in time values) between the clocks. When synchronizing multimedia streams such as in lip synching clock skew is a real concern.

For this reason, each multimedia data source inserts periodic synchronization points in the data stream. The synchronization points can be media points or an event counter which preserves the partial ordering of events [12]. The length of the period for insertion of sync points can be adjusted to give the desired quality of synchronization, with the tradeoff of increased overhead for more sync points.

The trigger can use notion of the partial ordering of events using logical clocks [12] or skew minimization [13] by exchanging clock information at the media points to perform synchronization of the distributed streams. A condition may be specified for a maximum clock skew which the trigger must enforce. Then based on the algorithm being used, the trigger sets the parameters for the technique (e.g. the number of times per second that an NTP server must be queried by each multimedia data source) to satisfy that condition.

5. Related Work

Various models have been used to describe synchronization [1, 14, 15], and a rich literature exists on modeling temporal synchronization in multimedia presentations [2, 16, 11].

Synchronization based models cover issues related to synchronizations of streams [16], the integration of user interaction [9, 10] and the navigation of multimedia documents [5, 17]. These systems do not perform content based analysis, cannot accept events in unordered fashion, and grouping that alters the clock to preserve synchronization are not considered.

Our system can perform synchronization of streams and user interaction in a generic manner. In addition, our system can accepts events in any order (including asynchronous events) using the active repository. Our active repository also performs content based analysis to identify the conditions which above systems do not perform. Our notion of grouping the streams for changing the clock is absent in above systems.

Only recently has there been some effort to identify events based upon content analysis of multimedia [4]. Their model distinguishes between continuous behavior and sequences of discrete events. Even though the model considers discrete events (i.e. user interaction) and continuous events (i.e. author-specified-level conditions) there is no consideration of past events that in our model are handled by the active repository. This model can not perform any grouping of streams, and synchronization based on grouping is missing. They do not discuss handling of unordered events.

6. Conclusions

We have described an enhanced model of synchronization for Internet based multimedia languages, which supports computation, reactivity, visualization, and event based programming based upon multiple multimedia streams and their synchronization. The model is capable of handling both synchronous as well asynchronous events. The model also supports dynamic scaling of groups of time bases to maintain the synchronization. The order and delay in the asynchronous events has been handled by the use of an active persistent repository which sets partial conditions true which can be used by the trigger to identify an event and start a new thread of multimedia computation. The history in the active repository allows the model to be used in complex reactive multimedia systems where the reaction can be triggered both in the presence of past and present signals.

We have developed an XML based language based upon this formalism which will be presented elsewhere.

References

[1] A.F. Ates, M. Bilgic, B. Sarikaya. Using Timed CSP for Specification, Verification and Simulation of Multimedia Synchronization. IEEE Journal on Selected Areas in Communications, 14(1): 126-137, 1996.

[2] E. Bertino, E. Ferrari. Temporal Synchronization Models for Multimedia Data. IEEE Transactions on Knowledge and Data Engineering, 10(4): 612-631, 1998.

[3] S.C. Buraga, M. Brut. Using Multimedia Presentations on Web. Procs. of 14th Interantional Conferernce on Control Systems and Computer Science – vol.II, I. Dumitrache and C. Buiu (eds.), Politehnica Press, Bucharest, 2003.

[4] H. Cameron, P. King, S. Thompson. Modeling Reactive Multimedia: Events and Behaviors. Multimedia Tools and Applications, 19: 53-77, 2003.

[5] A. Celentano, O. Gaggi. Synchronization Model for Hypermedia Document Navigation. Proceedings of the 2000 ACM Symposium on Applied Computing: 585-591, Como, 2000.

[6] S.-C. Chen, M.-L. Shyu, I. Gray, H. Luo. An Adaptive Multimedia Transmission Protocol for Distributed Multimedia Applications. Proceedings of the 23rd International Conference on Distributed Computing Systems Workshops: 537-542, 2003.

[7] Y.-P. Chen (editor). Proceedings of the 10th International Multimedia Modeling Conference, Jan.05-07, 2004, Brisbane, Australia.

[8] R. Gordon, S. Talley. Essential JMF – Java Media Framework. Prentice Hall, 1999.

[9] C. Haung, C. Wang. Synchronization for Interactive Multimedia Presentation. IEEE Multimedia, 5(4): 44-62, 1998.

[10] I. Herman, et al.. A Standard Model for Multimedia Synchronization: PREMO Synchronization Objects. Multimedia Systems, 6(2): 88-101, 1998.

[11] M. Jourdan, N. Layaïda, C. Roisin. A Survey on Authoring Techniques for Temporal Scenarios of Multimedia Documents. Handbook of Internet and Multimedia Systems and Applications, Part I. CRC Press: Boca Raton, FL, 1998.

[12] L. Lamport. Time, Clocks and the Ordering of Events in a Distributed System. Communications of the ACM, 21(7): 558-564, 1978.

[13] D.L. Mills. Internet Time Synchronization: the Network Time Protocol. IEEE Transactions on Communications, 39(10): 1482-1493, 1991.

[14] P.N.M. Sampaio, J.P. Courtiat. A Formal Approach for the Presentation of Interactive Multimedia Documents. ACM Multimedia, 2000.

[15] P. Senac, M. Diaz, A. Leger, P. De Saqui-Sannes Modeling Logical and Temporal Synchronizations in Hypermedia Systems. IEEE Journal on Selected Areas in Communications, 14(1): 84-103, 1996.

[16] Synchronized Multimedia Integration Language (2.0) http://www.w3.org/TR/smil20/, Aug. 2001.

[17] M. Vazirgiannis, S. Boll. Events in Interactive Multimedia Applications: Modeling and Implementation Design. Proceedings of the IEEE International Conference on Multimedia Computing and Systems (ICMCS'97), June 1997, Ottawa, Canada.