Supporting Weak Synchronization over World Wide Web*

TAN Kun SHI Yuan-chun XU Guang-you

(Department of Computer Science and Technology Tsinghua University Beijing 100084)

E-mail: tank@sist.tsinghua.edu.cn

Abstract The World Wide Web has become a new and powerful medium. However, applications on the Web are currently not fully multimedia supported. This is mainly because that the synchronization in the Web environment can be extremely complex, as the network infrastructure on which the Web is based provides no guarantee of real time data transfer, especially for large media, such as audio and video. In this article, weak synchronization is introduced as an efficient approach to specify temporal relationships among objects in a multimedia document to gain the best presentation performance. This approach provices a new modeling mechanism that can specify the precise as well as imprecise temporal relationships among involved objects. Specifications on inter-media synchronization, such as strict synchronization method and loose synchronization method, are also included in the approach.

Key words Distributed multimedia, Internet, Web, synchronization, temporal relationship.

With its glorious growth, the World Wide Web has proved its power as a new medium. Many Web-based applications have emerged and advanced rapidly. Nowadays, besides surfing from page to page, people can do shopping, receive education or even do business over the Web. However, with the rapid development of the Web, users also demand a more 'natural' way to communicate and exchange information; that is, to integrate multimedia information into these applications. Unfortunately, there still lacks support for multimedia on the Web because of its distributed nature. Therefore, the multimedia synchronization remains an open problem in a distributed environment such as the Web. Synchronization between media objects can be considered at several levels. Meyer, Effelsberg and Steinmetz[1] proposed a four-layer synchronization reference model, as shown in Fig. 1. The four layers are: the media layer for intra-stream synchronization of media stream, the stream layer for the inter-stream synchronization between media streams, the object layer for the presentation of abstract multimedia objects, and the specification layer for user interface. There are two major missions to synchronize objects in a distributed environment: 1) determining and specifying temporal relationships that are required, and 2) implementing scheduling schemes for objects, including inter-media synchronization and intra-media

* This research is supported by the National 'Ninth-Five' Sci-Tech Key Project of China ("九五"国家重点科技攻关项目, No. 96-743-01-06-04). **TAN Kun** was born in 1973. He is a doctoral candidate in the Department of Computer Science and Technology, Tsinghua University. He received his BS and MS in computer science from Tsinghua University in 1997 and 1999, respectively. His research interests include multimedia systems, distributed environments, and multimedia networking. **SHI Yuan-chun** was born in 1967. She received the Ph. D. degree in computer science from Tsinghua University in 1999. She is an associate professor in the Department of Computer Science and Technology, Tsinghua University. Her research interests include distributed multimedia information modeling and CSCW. **XU Guang-you** was born in 1940. He is a professor in the Department of Computer Science and Technology, Tsinghua University. He is a senior member of the IEEE and the chairman of the Department of Multimedia Technology, China Graphics and Image Association. His research interests include HCI and computer vision.

Manuscript received 1999-03-24, accepted 1999-06-15.

synchronization to guarantee the temporal specification of the involved objects.

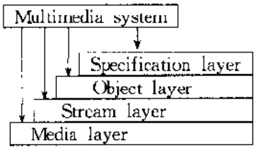

Fig. 1 The synchronization reference model

Nowadays, researchers have developed many models and methods for temporal relationships in the multimedia presentation[2~4]. However, most of these approaches focus on only precise temporal relations. Very few approaches support imprecise temporal relations. Because of the distributed nature of the Web, some timing information might not be determined by the creators of the multimedia presentations in advance, i.e., the duration of a video stream, so that the temporal relationships among objects need, or can, not be defined precisely. Therefore, the creator may need a more powerful way to specify imprecise temporal relationships among media objects lacking the interrelated temporal information.

In this article, we present a new approach for the synchronized multimedia documents on the Web. This approach includes a new modeling mechanism that can specify the imprecise as well as precise temporal relationships among involved objects. Our method also provides specifications on low-level synchronization issues, such as inter-media synchronization.

1 Temporal Modeling of Objects

Time in a multimedia document could be considered as a point or as an interval. A temporal point is zero-length time duration, while a temporal interval is a nonzero duration of time. Because the multimedia objects always have durations in their playout, it is convenient to use temporal intervals in discussing the temporal relationships among these objects.

J. F. Allen[5] introduced a temporal model based on intervals in 1983, which includes 13 simple interval relationships. Among these types, some are invertible, i.e. before and after, etc. Thus, only seven are original. They are before, meet, equal, overlap, during, start, and end, as shown in Fig. 2. These temporal relationships are considered as precise because the relation between the start times and end times of intervals is explicitly defined. However, in some cases, the temporal relationship can be imprecise. For example, in the situation that two objects A and B are to start simultaneously, the only constraint specified here is $Sa=Sb$, where Sa is the start time of object A and Sb is the start time of object B, respectively. Since we give no constraint on the end times of these objects, the relationship between them is imprecise. When the presentation is played back, the relationship of the objects' playouts can be one of the following: A equals B, A starts B, or B starts A. We use imprecise temporal relationship costart to specify this situation.

Wahl and Rothermel[6] introduced an enhanced interval-based model in 1994. In the enhanced model, 29 temporal relationships are presented. Besides Allen's 13 precise relationships, 16 imprecise relationships are included, and 10 of them are original——startin, endin, delays, cross, startendof, beforestartof, beforeendof, costarts, coends and all. Wahl used ten temporal operators to specify the interval relationships in this model, while Li Minglu[7] reduced these operators down to two in 1996. In Ref. [7], Li states that with little enhancement, every relationship described above can be expressed by before and costart operators. This theorem has become the basis of our specification approach.

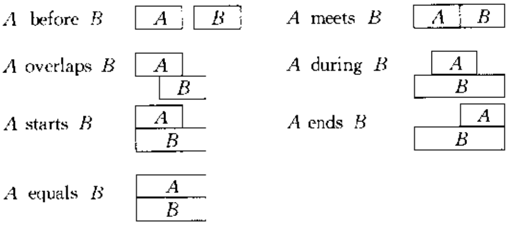

Fig. 2 Types of the precise relationships

Imprecise temporal relationships are helpful in specifying multimedia scenarios that contain uncertain timing aspects. In distributed environments, the undetermined network delay may influence the temporal relations in many ways. For example, a user may specify a multimedia scenario where some text and image objects are displayed while background music is playing. However, for viewers, the background music provides no important information. Thus, one might get annoyed if he must wait for the music object to be prepared before he can see the presentation, especially when the preparation takes a long time due to the network latency. Therefore, the author should use imprecise timing operator costart (with a delay defined as a subset of +0) to specify this scenario, that means 'display the text and image promptly and synchronously, and a flexible played back of the background music is acceptable'. In addition to this, even some precise relationships can be in fuzzy styles under the influence of network delay. For instance, a video clip needs to be played back with a picture as its background. However, as the video data are streamed down from server, the actual playback time of the clip can not be decided in advance, therefore the duration of the picture object is also undetermined. Thus, the user must define this scenario in a fuzzy style specification, e.g., the video must be played back during the playout of the background picture.

2 Strict and Loose Synchronization Mechanism

The synchronization mechanism of media streams can be classified into two types: intra-medium synchronization and inter-media synchronization. Intra-medium synchronization deals with internal behaviors of a medium stream, i.e., to eliminate jitter, while the inter-media synchronization maintains temporal relationships of several media streams, i.e., to limit the skew of two media streams. In the model discussed above, aspects of these lower layers of synchronization are not included. However, when we compose multimedia document in a distributed environment, we must take these aspects into consideration, especially the inter-media synchronization, which should be included in the synchronization specification.

Let's consider the following two slide applications. In the first one, slides are shown one by one with an audio narration. The slides and the audio must be synchronized, because a certain segment of audio should correspond to a specific slide. However, as the audio is streamed from the server, the synchronization might be broken when the audio stream is blocked due to network faults. To tame such a problem, the presentation client must have the ability to detect this blockage of media streams. When one synchronized object is blocked, the client should be able to block other related objects. The presentation should be allowed to resume when all blocked objects have enough data for display. Thus, the skew of the audio stream and presentation of other objects is limited. However, things might differ in the other application, where instead of an explanatory audio, a background music object might play back throughout the sequence of the images. This time, the music and the image sequence need not be strictly synchronized. Thus, if the network blocks the music stream, we do not need to stop the presentation of the image sequence. These two media objects can be rendered according to two different presentation clocks with little harm to the whole presentation, while improving the efficiency of the system.

We define these two inter-media synchronization methods as strict synchronization mechanism, for the former, and as loose synchronization mechanism, for the latter. In strict synchronization, involved objects are played back with the same clock. When one object is blocked, the whole presentation is blocked as well. In loose synchronization, each object has its own presentation clock and the blockage of a single object will not affect other objects.

3 Weak Synchronization Model

We use terminology 'weak synchronization' to title our approach that we use to specify the temporal relationships in a multimedia document. In our approach, imprecise temporal relations and precise temporal relations (including those fuzzy styles) can be defined. Moreover, inter-media synchronization specifications are also included in our approach. The reason why we take the model as a 'weakened' synchronization is based on the following considerations:

- Because of the distributed nature of the Web, many scenarios can not be specified with traditional methods, such as time-line, hierarchical specification, etc., which only deal with precise temporal relationships. Thus, we have to include the imprecise temporal specifications in our model. Compared with precise relationships, imprecise relationships are weakened as they are described only by certain references without several explicit timing parameters, such as start time, end time or duration.
- Loose synchronization mechanism between media streams is considered as an important aspect of weak synchronization. As we have discussed above, loose synchronization provides a flexible way to ignore the blocks of unimportant media streams and present the important part of the presentation to viewers efficiently. Regarding to strict mechanism, this mechanism is much weaker.

Before we proceed with the detail specification of weak synchronization, let us first take a quick glance at objects in the model.

3.1 Objects in weak synchronization

Every medium is abstracted as an object in weak synchronization. An object is an atom in multimedia presentation, such as a piece of text, a picture or a video clip. Each object has registered several states. The presentation of a medium object is always related to a series of transitions of object states. There are six states of a medium object:

- Defined. In defined state, the object's name and type are registered and the memory has been allocated for new object, but the content in the memory is invalid.
- Created. In created state, the object's necessary properties have been specified and the system is preparing the data needed for the object.
- Ready. If all data needed are prepared, the object becomes ready.
- Running. In running state, the object is presented to the viewer.
- Retired. After presentation, the running object retires and disappears from the output device, but in retired state, all properties of the object keep valid.
- Dead. In dead state, all resources relevant to the object are released and the registrations are removed from system database.

Objects also have several properties. Some properties are considered as presentation parameters of the object, for example, the layout information of a picture object. Others are synchronization properties. Two synchronization properties are most important. They are duration, which explicitly specifies the presentation duration of the object, and initial-delay, which explicitly specifies the delay from the time the object is ready to the time the object is really displayed to the viewer.

3.2 Specifying weak synchronization

The playback of a multimedia document can be considered as a process. This process, called sync-process, contains a sequence of finite synchronization primitives. To specify weak synchronization, we introduce three primitives: sync, wait and follow.

Sync primitive is a block operation. A sync operation is considered to be finished after all involved objects are in ready state. (A time independent media object will be ready when the data are fully downloaded, whereas a time-dependent media object is ready when it has collected enough data in the buffer pool). The syntax of sync is

SYNC (object1, object2, ..., object$n$): sync_type

Sync_type can be strict (which indicates a strict synchronization method should be applied) or loose (a loose synchronization method should be applied).

Wait primitive is also a block operation. The syntax of wait primitive is

WAIT (object1, object2, ..., object$n$): wait_type

The parameter wait_type indicates how wait primitive is operated. If the wait_type is last, the wait operation is finished when all involved objects are dead. If the wait_type is first, the wait operation is finished when any single object is dead, and the result of the operation will force to kill all relevant objects.

Follow primitive specifies the objects that follow a certain object. When an object is finished, all of the subsequent objects should be started. Moreover, the ensuing objects should inherit the synchronization attributes of the object being followed. An object can have several successors. The syntax of follow is

FOLLOW Object BY (object1, object2, ..., object$n$).

With high abstraction and expression ability, these three synchronization primitives can specify various temporal relationships in any kind of multimedia document. The modeling of all 29 interval temporal relationships is shown in Appendix A.

Example 1 (A before B). This relationship can be specified as the following sync-process.

Sync-process 1:

Objects $A(a,b)$, $B(c,d)$
FOLLOW A BY B
WAIT (B): first

The follow operation specifies that object B is displayed after object A. The delay between these two objects is determined by parameter c.

Example 2 (A costarts B). This relationship can be specified as in sync-process 2.

Sync-process 2:

Objects $A(0,b)$, $B(0,d)$
SYNC (A,B): loose

The sync operation synchronizes objects A and B at the beginning. With the loose synchronization method, we can not tell which object will end first no matter what value we define for parameters b and d.

Example 3 (A during B). This relationship means the object A is played back during the playout of the object B. However, this scenario can be fuzzy if the presentation duration of any object, e.g., object A, can not be determined in advance. The only thing we can infer from this scenario is the object B has a longer lifetime than object A. In order to specify this scenario, a dump object is introduced. A dump object is not a presentable object, but it obeys the same synchronization rules as normal medium objects. The scenario can be specified as in sync-process 3, where the short dash lines represent the timing parameters that are not defined.

Sync-process 3:

Objects $A(a,-)$, $B(0,-)$, $D(0,d)$
SYNC (A,B): loose
FOLLOW A BY D
WAIT (B,D): first

Object D here is a dump object. The duration of dump object is d. The sync operation synchronizes objects A and B at the beginning and the objects A starts a seconds later than the object B. While object A is finished, the dump object D is triggered to start. Then after d seconds, object D is finished and object B is forced to die according to the wait operation.

Example 4 (a more complex scenario). The scenario includes several distributed media objects (Fig. 3). The scenario start with a picture, $P1$, and a background music object, $A1$. Then, a text object $T1$ overlaps $P1$. When $T1$ finishes, after a short delay, image sequence $P2$ and a video $V1$ start simultaneously, and these two objects should be ended at exactly the same time. The background music, $A1$, is playing back during the whole presentation. The sync-process is outlined below.

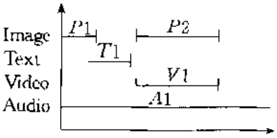

Fig. 3　A scenario of multimedia presentation

Sync-process 4:

Objects $P1(0,b)$, $T1(a,c)$, $A1(0,-)$, $P2(d,e)$, $V1(d,e)$

SYNC ($P1$,$A1$,$T1$): loose

FOLLOW $T1$ BY ($P2$,$V1$)

WAIT ($T1$): last

SYNC ($P2$,$V1$): strict

WAIT ($V1$,$A1$): first

In Examples 1 to 4, we show that our specification method is rich enough to deal with imprecise as well as precise temporal relationships among media objects. The sync-process has several advantages, such as simple and flexible. It can also be implemented easily with scripting languages.

4　Architecture and Implementation Issues

Weak synchronization has been implemented in the NOAH multimedia system[6] at Tsinghua University. NOAH (network-oriented object architecture for hypermedia) contains a hypermedia information model, a scripting language (NOAHScript) and a browser software which can browse traditional HTML documents as well as those in the NOAH document format.

The NOAH multimedia system is wholly based on the current Internet infrastructure. It uses a Web server and HTTP to transfer data from server to client. The architecture of the NOAH system is shown in Fig. 4.

- The asynchronous reading sub-system provides network services for time-independent media, such as documents, texts and images. Priority can be specified for each object. Higher priority objects should be served earlier. There is an implicit priority for each type of media: the shorter data media usually have higher priority. However, the user can specify explicitly the priority of the objects by himself.

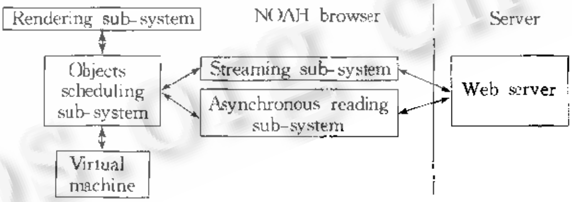

Fig. 4　Architecture of NOAH system

- The streaming sub-system provides streaming services for time-dependent media, such as audio and video.

- The objects scheduling sub-system is the core of the browser. It schedules objects based on the specification in a multimedia presentation.

- The virtual machine interprets and executes the virtual instructions. These instruction sequences are the compiled results of NOAHScript actions, which are used to control the objects and do client-side information processing.

- The rendering sub-system presents the active objects to the user by audio-visual means.

4.1　The streaming sub-system

The streaming sub-system is extremely important in a distributed multimedium system as it provides control on intra-medium synchronization of a single medium stream. The architecture of the streaming sub-system in the NOAH system is shown in Fig. 5.

- The HTTP I/O module is the interface to the network. It is implemented as an HTTP client, sending HTTP requirements and retrieving data from the server.
- The buffer pool provides a management to buffers. The buffer should be large enough to contain at least k LDUs (k is a configurable parameter) in the medium stream. The buffer pool module maintains buffers for each stream, and when the stream runs out of data, the module should set an event, which can be caught by the object scheduling sub-system.

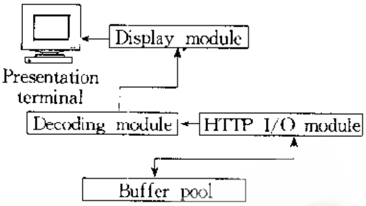

Fig. 5 Architecture of the streaming sub-system

- The decoding module decodes all compressed media data to raw data, which are ready for display.
- The display module, related to the rendering sub-system, displays cooked media data to the user.

4.2 Maintaining strict synchronization

When strict mechanism is specified to maintain the inter-media synchronization of two media streams, the system needs to limit the skew between them. Thus, if data of one stream fail to arrive in time, the system needs to detect this starvation, stop the relevant presentation, and after enough data are gathered, resume the blocked presentation.

When a strict sync operation is performed, the objects scheduling sub-system will create a semaphore to guard all involved objects. The semaphore works as a counter with two operations: get and release. A get operation increases the count by one whereas a release operation decreases the count by one.

At the beginning, the semaphore has its maximum value. As each synchronized object gets ready, the object will perform a release operation until the semaphore has a zero value which means the presentation is ready for display. The rendering sub-system will then render the document to the user.

If any object is blocked, a starvation signal should be set by the streaming sub-system and received by the objects scheduling sub-system. It will cause a get operation to be performed, which gives the semaphore a value greater than zero. Consequently, the presentation is paused.

If the blocked object has enough data, a recover signal will be sent. Accordingly, a release operation is performed. The presentation will be resumed if all blocked objects have sent a release operation.

The maximum value of the semaphore could be changed at runtime. When a new object is about to be displayed, the maximum value of the semaphore should be increased by one. Accordingly, a get operation should be also performed (a release operation should be performed automatically when the object has finished its preparation). However, when an object has died off, the maximum value of the semaphore should be decreased by one. At this time, no additional release operation should be performed unless the object is in blocking state.

5 Related Work

The HTML (HyperText Markup Language) standard was developed to specify the hypertext document on the Web. However, based on static document structure, HTML document does not have the notion of time, so that no synchronization is supported in the traditional HTML document. Recently, many methods have been developed to extend HTML to support multimedia presentations.

Macromedia Company has developed a Plug-in called Shockwave for Netscape Navigator, which can display multimedia presentations created by Director. Director is a multimedia authoring and playback tool developed by Macromedia Company. Synchronization in Director is achieved by means of the timetable in which the user could define frame ordering in a movie and control the timing of special effects such as transitions, sound, and palettes. Director also provides a script called Lingo to control the navigation of the multimedia documents. However, Shockwave is a simple extension to support multimedia on the Web. Although it streams presentations over the Web, the synchronization in the documents is still strict and timetable based, which will lead to severe inefficiency in some cases.

Microsoft Corporation has recently extended traditional HTML to DHTML[9](Dynamic HyperText Markup Language). In DHTML, many new features are added, such as layout information, timing, dynamic styles and partial updating. With these newly provided features, authors can build more wonderful multimedia presentations on the Web. Nonetheless, although authors can perform some synchronization tasks using DHTML, there is still no systematic way to specify temporal relationship among objects.

The SMIL[10] (Synchronized Multimedia Integration Language) is a proposed recommendation by W3C. SMIL focuses on the specification of synchronized multimedia presentations in the World Wide Web. While still only being a proposal, SMIL has received support from several companies. Some of these companies have delivered trial versions of browsers that can view SMIL documents. SMIL specifies the temporal relationships among objects as in parallel or in sequence. SMIL also supports loose synchronization, or soft synchronization, between objects that are displayed in parallel. Although SMIL is quite powerful for creating multimedia presentation and it does support a few specifications on imprecise temporal relationships, such as, costart, coend, etc., it can not specify all imprecise temporal relationships that authors may encounter, for example, cross. Moreover, SMIL doesn't include some fuzzy styles of temporal relations either, for example, the "during" relation provided in Example 3. Another limitation in SMIL is the lack of ability in controlling the objects and navigation at the runtime, which prevents it from becoming a good framework for a multimedia application.

6 Application

NOAH multimedia system is designed as a powerful framework that is suitable for many multimedia applications on the Web, such as information publishing, tele-education and other audio-visual interactive applications. A demonstration site in Tsinghua has been built for current researching achievements of the university. In this site, large videos and audios all together with slides are integrated in synchronized multimedia documents. Since all documents are specified with the weak synchronization, it is possible for server to adapt documents to different clients all around the network. For instance, for high-speed connections, full documents are transferred, while for low-speed connections, server will block the large video streams to reserve bandwidth, but the clients can present the other parts of the document to viewers according to the weak synchronization specification.

7 Conclusions

In this paper, weak synchronization is introduced to meet the requirement of the current Web. The weak synchronization model supports the imprecise temporal relationships among multimedia objects, and the strict and loose synchronization methods for inter-media synchronization. Sync-process is used for specifying weak synchronization scenarios. Each sync-process contains a series of synchronization primitives, and runs in parallel in a preemptive multitasking system. Moreover, sync-processes can also be considered as objects, therefore they can be operated by synchronization primitives too. With this hierarchical organization, authors can solve synchronization issues in a divide-and-conquer way, which will be much useful for authors, especially in composing large multimedia scenarios.

Acknowledgments

This project is supported by CERNet (China Education and Research Network) Developing Plan.

References

1 Steinmetz R, Nahrstedt K. Multimedia: Computing, Communications and Applications. New Jersey: Prentice-Hall International Inc., 1997. 599～608

2 Karmouch A, Emery J. University of Ottawa, a playback schedule model for multimedia documents. Spring: IEEE Multimedia, 1996. 50～61

3 Vazirgiannis M, Mourlas C. An object-oriented model for interactive multimedia presentations. The Computer Journal, 1993,36(1):79～86

4 Roure D D, Hall W. Distributed Multimedia Information Systems. IEEE Multimedia, 1997. 68～73

5 Allen J F. Maintaining knowledge about temporal intervals. Communications of ACM, 1983,26(11):832～843

6 Wahl T, Rothermel K. Representing Time in Multimedia Systems. In: Proceedings of the International Conference on Multimedia Computing and System. MA, Boston: IEEE Computer Society Press, May 1994. 538～543

7 Li Ming-lu. On multimedia synchronization: generalized model, scripting language and implementation[Ph. D. Thesis]. Shanghai Jiaotong University, 1996

8 Tan Kun, Shi Yuan-chun, Xu Guang-you. A distributed hypermedia information model: NOAH. In: Proceedings of the 6th National Conference on Multimedia Technology. Xi'an, China, 1996. 232～238

9 Dynamic HyperText Markup Language Specification. Microsoft Corporation, MSDN 1998. 1

10 Synchronized Multimedia Integration Language (SMIL) 1.0 Specification. W3C Working Group on Synchronized Multimedia. http://www.w3.org/TR/1998/PR-smil-19980409/

Appendix A

Temporal relations of two intervals	Temporal points relations	Specification with weak synchronization model
A all B A before B	$Ea<Sb$	Objects $A(a,b)$, $B(c,d)$ Follow A By B Wait (B): first
A meets B	$Ea=Sb$	Objects $A(a,b)$, $B(0,d)$ Follow A By B Wait (B): first
A overlaps B	$Sa<Sb, Ea>Sb, Ea<Eb$	Objects $A(a,b)$, $B(c,d)$, where $a<c$, $c<a+b$ Sync (A,B): strict Wait (B): last
A ends B	$Sa<Sb$, $Ea=Eb$	Objects $A(a,b)$, $B(0,d)$, where $a+b=d$ Sync (A,B): strict Wait (A,B): first
A during B	$Sa>Sb$, $Ea<Eb$	Object $A(a,b)$, $B(c,d)$, where $a>c$, $a+b<c+d$ Sync (A,B): strict Wait (A): last
A starts B	$Sa=Sb$, $Ea<Eb$	Object $A(0,b)$, $B(0,d)$, where $b<d$ Sync (A,B): strict
A equals B	$Sa=Sb$, $Ea=Eb$	Object $A(0,d)$, $B(0,d)$ Sync (A,B): strict
A startin B	$Sa<Sb$, $Sb<Ea$	Object $A(0,d)$, $B(b,-)$, where $d>b$

		Sync (A,B): strict
A endin B	$Sa<Eb$, $Ea>Eb$	Object $A(a,-)$, $B(0,d)$, where $a<d$ Sync (A,B): strict Wait (A,B): last
A cross B	$Sa<Eb$, $Ea>Sb$	Object $A(a,-)$, $B(b,-)$, $D1(0,b)$, $D2(0,a)$ Sync (A,B): loose If $(a<b)$ {Sync $(A,D1)$ Wait $(A,D1)$;last} Else {Sync $(B,D2)$ Wait $(B,D2)$: last}
A delays B	$Sa<Sb$, $Ea<Eb$	Object $A(0,-)$, $B(b,-)$, $D(0,d)$ Sync $(A(0,-), B(b,-))$: strict Follow A By D Wait (D,B): last
A startendof B	$Sa<Eb$	Object $A(a,-)$, $B(-,-)$, $D(0,a)$ Sync (A,B,D): strict Wait (B,D): last
A beforestart of B	$Sa<Sb$	Object $A(0,-)$, $B(b,-)$ Sync (A,B): loose
A beforeend of B	$Ea<Eb$	Object $A(-,-)$, $B(-,-)$, $D(0,d)$ Follow A By D Wait (D,B): first
A costarts B	$Sa=Sb$	Object $A(-,-)$, $B(-,-)$ Sync $(A(0,-), B(0,-))$: loose
A coends B	$Ea=Eb$	Object $A(-,-)$, $B(-,-)$ Wait (A,B): first

Here, "−" means no definition or any non-negative real, $Sa(Sb)$ stands for the start time of object $A(B)$, $Ea(Eb)$ stands for the end time of object $A(B)$, and a,b,c,d are positive real numbers. $D,D1,D2$ are dump objects.

弱同步及其在WWW上的应用

谭焜　史元春　徐光祐

(清华大学计算机科学与技术系　北京　100084)

摘要　在当今世界,WWW已经成为新颖而强大的媒体,但是,现在运行于Web上的应用仍然不是完全多媒体化的.这主要是因为Internet网络是不可靠、尽力型的,无法提供数据实时传输的保证,尤其对那些大数据量的连续媒体(如视频与音频数据).因此,在Web环境中多媒体对象间的同步就变得格外复杂.该文提出了一种新颖的同步模型:弱同步.它使多媒体文档即使在多变的环境下也能获得最佳效果.在弱同步中,不仅可以定义对象间确定的时间关系,也可以定义非确定的时间关系.此外,媒体间的同步关系,如严格同步和松散同步,也可以在文档的同步描述中定义.

关键词　分布式多媒体,互联网,万维网,同步描述,时间关系.

中图法分类号　TP393