SBMídia 2001 - VII Simpósio Basileiro de Sistemas Multimídia e Hipermídia, p.37 - 52.
Florianópolis, Brazil. SBC, 2001

Bringing Flexibility to the Specification and Coordination of Temporal Dependencies among Multimedia Components

André L. V. Coelho, Alberto B. Raposo, Ivan L. M. Ricarte

Dept. of Computer Engineering and Industrial Automation (DCA)
School of Electrical and Computer Engineering (FEEC) - UNICAMP
P. O. Box 6101 – 13083-970 – Campinas, SP

{coelho, alberto, ricarte}@dca.fee.unicamp.br

Abstract. *We introduce a methodology for the high-level specification and decentralized coordination of temporal interdependencies among multimedia document objects. Such methodology encompasses a three-step process comprising (i) the design of multimedia presentation scenes by means of a fuzzy descriptive plan; (ii) the parsing of such layout to classify the multimedia entities that compose the scenes and to check the consistency of temporal relationships among them; and (iii) the generation of event-driven time and action managers as distributed mechanisms for the orchestration of the elements presentation. This approach centers around a novel multimedia synchronization model based on fuzzy sets and software components concepts.*

1. Introduction

The representation, specification, and orchestration of temporal interdependencies among multimedia objects comprehend a basal, yet complex, task to be properly addressed in the construction of any multimedia system [Blakowski 1996, Little 1990]. Such activity is referred to as *intermedia temporal synchronization* since it takes into account the various kinds of relationships between time-dependent (continuous) as well as time-independent (discrete) media objects. In such realm, there are primarily two complementary subject issues to be undertaken. The first concerns on how to indicate (represent and specify), step by step, the sequential or concurrent ordering of the various media contents presentation. This is known as a *multimedia synchronization model*. The second issue copes with how to guarantee the correct exhibition (orchestration) of the already specified relations. This is the role of *multimedia synchronization mechanisms.*

To provide more flexibility in the portrayal of multimedia presentation scenarios, we focus here on a new methodology approaching a high-level, designer-oriented, descriptive specification of intermedia temporal associations. In this scheme, both of the above issues are contemplated: The media objects' interdependencies are portrayed in a *fuzzy synchronization model* wherein fuzzy rules and constraints [Pedrycz 1998] behave as basic synchronization mechanisms that are directly derived from the authored scenario characterization. The idea is to capture the interactions between multimedia document (MMDoc) elements in a more amenable way from the designer perspective as well as to give space for the representation and handling of inaccurate relationships and unpredictable situations. Such aspects are very appealing for a range of circumstances, mainly when one wishes to take advantage of the soft character behind most multimedia synchronization requirements–as for pre-elaborated multimedia presentation scenarios–, or when it is desirable to promote support for user interactivity.

Such methodology (named as *MuSCoF*[1]) is oriented towards the pervasive appliance of software components [Hopkins 2000], as they comprehend a conceptual abstraction that seems to be more suitable for the modeling of multimedia document elements. Behind this decision, the purpose is twofold: (*i*) to embed in the same container all structural and logical information related to a particular media object (such as the various presentation plans that may be available for distinct environments); and (*ii*) to decentralize the coordination activities necessary for the proper arrangement of the (possible numerous) elements presentation. *MuSCoF* shall be implemented in a proper multimedia environment, conceived to assist the designer in all authoring phases (scenarios specification, verification, implementation, and deployment).

In the sequence, we acquaint the reader with the fundamental concepts underlying fuzzy systems, software components and the multimedia synchronization problem (Section 2). Further, the proposed methodology is described, following a top-down vision. This is done in Section 3, where we focus on the fuzzy synchronization and multimedia component models. Section 4 is devoted to an illustrative example, whereas in Section 5 we situate and assess our proposal in light of related work. Finally, some concluding and future work remarks are presented.

2. Background

For the sake of clarity, this section concentrates on some basic notions, concepts, and requirements that are handled further in the methodology specification.

2.1. Fuzzy Systems

The basic concept underlying fuzzy systems theory is that of *fuzzy set* [Pedrycz 1998]. Fuzzy sets involve capturing, representing, and working with linguistic notions–objects with unclear boundaries–and are very apropos to be employed in those circumstances where impreciseness, unpredictability, vagueness, approximation, soft gradation and flexibility are in concern. A fuzzy set has been defined as a collection of objects with membership values between 0 (complete exclusion) and 1 (complete agreement), where these values express the degrees to which each object is compatible with the properties or features distinctive to the collection. A fuzzy set is a generalization of the concept of a set and can be defined in either finite or infinite universes. Formally, a fuzzy set *A* is characterized by a membership function (μ_A) mapping the elements of a domain of discourse $\mathbf{X}$ to the unit interval [0,1]. That is, μ_A: $\mathbf{X} \rightarrow [0,1]$. As for crisp sets, fuzzy sets can be compared or aggregated by means of special types of operators. Amongst them, it is worthy to quote the classes of fuzzy intersection and union operators, known as *triangular norms* (*t-norms*) and *co-norms* (*s-norms*). Examples of such operators are the *minimum* (*min*) and *maximum* (*max*) [Pedrycz 1998].

Beside the above definition lies another concept, that of a *linguistic variable*. In contrast to a variable with numbered values, a linguistic variable can be regarded as one that may assume fuzzy values consisting of words or sentences in a language. These fuzzy values which are used to model imprecise quantities are known as *fuzzy numbers*, each one associated with a characteristic membership function mapping the real line $\Re$ to the unit interval. Typically, a linguistic variable is characterized by a name Δ (e.g., *temporal duration*) and a finite set of primary linguistic terms $T(\Delta)$ (e.g., $T(\Delta)$ =

---

[1] *MuSCoF* stands for "Multimedia Synchronization via Software Components and Fuzzy Systems".

{*beginning*, *middle*, *finish*}), where each labeled term relates to a particular fuzzy set (with a proper membership function) over a defined partition of the variable's associated universe of discourse (Figure 1a). As well, a linguistic variable may be associated with a set of *modifiers*, such as hedges *H* (e.g., *very*, *more or less*, *quite*, *almost*, *few*), the connectives *and* and *or*, and the negation *not*. Such modifiers alter the semantics of the variable's linguistic terms and to provide for a means of representing and computing composite terms (e.g., *very high*, *not low and not very high*). For this purpose, modifiers are expressed as configurations of elementary membership function operators, such as those depicted in Figure 1b (*dilation*, *concentration*, and *fuzzification*). Table 1 brings expressions for some modifiers.

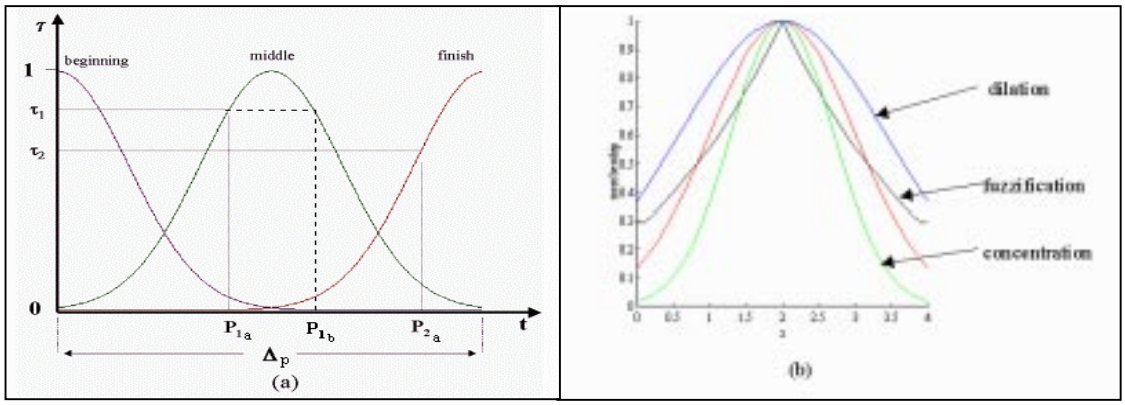

Figure 1. (a) A linguistic variable *Δ* and its partitioned terms. (b) Some elementary operators acting upon a Gaussian membership function.

Modifier/Operator	Expression
concentration (Con_L(x))	$L^p(x)$, $p > 1$
dilation (Dil_L(x))	$L^r(x)$, $r < 1$
fuzzification (Fuzz_L(x))	$[L(x)/2]^{½}$ if $L(x) \leq 0.5$ $1 - [(1 - L(x))/2]^{½}$ otherwise
very L	*Con_L(x), p =2*
plus L	*Con_L(x), p =1.5*
minus L	*Dil_L(x), r =0.75*
highly L	*plus very L or minus very very L*
much L	*Fuzz_L(x)*
more or less L	*Dil_L(x), r = 0.5*
not L and not very M	*[1-L(x)] **t** [1-Con_M(x)]*
very L or M	*Con_L(x) **s** M(x)*

Table 1. Some linguistic modifiers and their calculus. *L* and *M* stand for linguistic terms and *t* and *s* refer to t- and s-norms.

Linguistic variables constitute an expressive way of capturing the meaning of a concept or representing knowledge about real-life facts, such as *the temporal duration of A is on beginning*. Compound propositions (e.g., *the temporal duration of A is on beginning and the temporal duration of B is quite finishing*) may be constructed through conjunctions (t-norms) and/or disjunctions (s-norms) of other propositions. Conditioned fuzzy propositions, known as *fuzzy rules*, generally comprehend two distinct parts, an antecedent and a consequent (e.g., *IF the temporal duration of A is on beginning THEN the temporal duration of B is finishing*) and may be chained or fired together via inference mechanisms. Fuzzy rules are very suitable for grabbing the *causality* between events and actions. By other means, linguistic variables may be logically connected by

means of *fuzzy constraints* if one wishes to declare vague or partial (not stringent) relationships (dependencies or coupling) among the concepts they represent. Sets of fuzzy rules and fuzzy constraints are typically employed for high-level modeling and controlling purposes, as they properly capture the subjective aspects behind the expertise of a domain specialist (such as a multimedia presentation designer).

Albeit this range of functionalities, yet few reported works have tried to explore the applicability of fuzzy systems theory into multimedia context. This is particularly noticeable in the multimedia synchronization arena, where the relations among media elements are typically set through specification models that, most of the times, provide deficient support degree for modeling flexibility and uncertainty issues. In Section 5, we comment on other approaches concerning media synchronization with fuzzy aspects.

2.2. Software Components

Components are the smallest self-managing, independent, and useful parts of a system that works in multiple environments [Lewandowski 1998]. Hopkins (2000) defines a software component as "a physical packaging of executable software with a well-defined and published interface that can be delivered as a unit and that can be composed, unchanged, with others to build large loosely coupled systems." Components may contain multiple distributed or local objects, and they are often used to centralize code related to a specific functionality. The flexibility of *granularity* is also attractive: Components can be as small as required to be specialized in a particular capability, or as large as required to encapsulate all the logic of a particular subsystem without reliance on other components. There are two engineering drivers in the development of a component-based system: (*i*) the ability to *reuse* existing designs and implementations and (*ii*) the easiness of *maintainability.*

One of the most important distinguishing factors in component-based development is the separation of the interface and the implementation. Since components are designed for use in a variety of systems, their interface must identify clearly both the service they subsume and the means of invoking them (this is known as the component's *service contract*). By other means, a component may or may not be implemented using OO tools and languages. Components communicate through the generation of events following the producer-consumer style. In this sense, events can be viewed as synchronization objects that allow one thread of execution to notify others that something has happened. Using this model, an event can announce to a component that it should perform a certain action. Listening for events provides a more robust framework for interaction between objects than a model that forces their coupled interplay. All these aspects justify the employment of software components as an adequate paradigm for the modeling of multimedia presentation elements.

2.3. Multimedia Synchronization Requirements

It is possible to characterize specific interdependencies that occur in a multimedia scenario and identify coordination mechanisms to manage them. Among them, we can highlight the temporal interdependencies [Blakowski 1996, Pazandak 1994, Wahl 1994], which encompass those situations in which it is necessary to establish an execution order for the multimedia objects presentation, somehow resembling a *scheduling* process. Such objects encapsulate single or composite media and are interrelated by means of logical presentation plans. Once a multimedia scenario has been specified, using some formal or informal notation, its temporal synchronization

requirements must be represented in a format that is automatically processable by the subsequent steps. Such format is known as the *synchronization specification* of the multimedia scenario and follows the formalism expressed in a multimedia synchronization model. The fulfillment of the requisites stated by the synchronization specification is achieved by means of synchronization mechanisms, which pursue the correct orchestration of the elements real-time rendering.

With respect to how temporal elementary units and temporal dependencies are expressed, multimedia scenario models can be classified into several categories [Blakowski 1996, Kwon 1999, Wahl 1994]: *sync-point models*, *timeline models*, *hierarchical models*, *causality models*, *interval-based models*, and *constraint-based models*. In sync-point models, synchronization is realized by logically connecting together reference points (defined in a unidimensional time space) [Steinmetz 1990] that have been inserted within the media objects. Timeline models [Gibbs 1991] are also instant-based models, but now all multimedia objects are related only with the same time axis (a global timeline) and are independent to each other. In hierarchical models [Karmouch 1996], a multimedia presentation may be regarded as a tree, whose leaves are media objects and whose edges and internal nodes are composition operators. Although such models have the advantage of being very simple and intuitive, their major drawback is that it is impossible to specify *relative* temporal relationships among the playout of the various objects–this may turn to be very awkward in case of changes in the objects' playout or when not all presentation time spans are known in advance. Causality, interval- and constraint-based models attempt to overcome such deficiency. The first [Pazandak 1994] involves synchronizing begin/end points via specific *signals* (system, application, or user-defined events), following a "cause-effect" semantics (the relative temporal ordering is an implicit result of this process). In interval-based models, each object has an associated interval representing the time required for its presentation, and the synchronization requirements are specified by correlating such intervals through a set of basic (Allen's) relations [Huang 1998, Wahl 1994]. In constraint-based models [Karmouch 1996, Kwon 1999], synchronization aspects are described through sets of constraints of the form $event_1 \{<, =, >\} event_2 + delay$, where $event_i$ refers to instant events (start/end time), and *delay* is a number (zero or beyond) of time steps. As it is discussed in the next sections, the *MuSCoF* synchronization model may be regarded as a hybrid of most of the aforementioned approaches.

3. Methodology

MuSCoF encompasses a three-step process (Figure 2) and is based on two interdependent models (fuzzy synchronization model and multimedia component model). The steps include (*i*) the user input layout of the multimedia presentation scenes by means of a fuzzy script plan (synchronization specification); (*ii*) the parsing of such editing sketch for the classification of the multimedia entities that compose the scenes and for the consistency checking of the temporal relationships among them; and (*iii*) the generation of temporal and action managers (synchronization mechanisms) associated to each multimedia entity. Such models and steps are discussed in the next subsections.

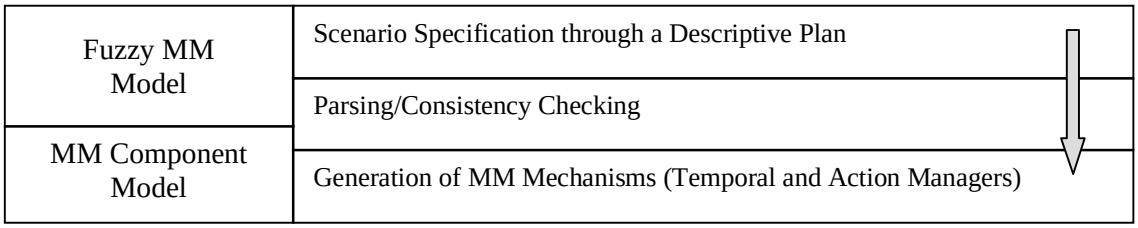

Figure 2. *MuSCoF* methodology.

3.1. Fuzzy Multimedia Synchronization Model

It is many times very difficult to completely describe the activities underlying a MMDoc scenario. This happens because the temporal/spatial relationships among the media elements may sometimes not adjust to an accurate specification of well-defined interdependencies. Such modeling imprecision implies, as a positive side effect, more flexibility to multimedia designers as they can focus on the high-level description of the scenes, without having to deal with peculiarities such as exact presentation times. Employing fuzzy sets theory, interdependencies in multimedia specifications can assume a more manageable perspective, in a manner more closely related to the subjective human reasoning, improving both the understandability and the feasibility of the interacting rules that identify the whole scheduling process.

In active MMDocs, *temporal objects* are the model elements upon which relations are specified. Pazandak and Srivastava (1994) enumerate three classes of temporal objects: the media objects (in *MuSCoF* they appear encapsulated in multimedia components–MMComps), events (referring to the endpoints of media objects as well as user- or application-defined signals), and timepoints (instants with no duration). Each of these elements should have their semantics fuzzified in our model, giving rise to *fuzzy temporal objects*. Some media objects have an implied *duration* (audio/video sequences are rendered at the rate at which they were captured), whereas others (static media such as image and text) are assigned to arbitrary temporal values according to the designer's intent. In a manner or the other, the majority of the temporal models only accommodate media objects with known or predictable duration, giving no treatment to the representation of uncertainty. Conversely, in *MuSCoF* we define two types of media duration associated to a MMComp that may have their values fuzzified: one relative to its presentation time Δ_p (temporal length since a *beginning event* in instant t_b until an *end event* in instant t_f) and the other relative to its time of liveness Δ_l (temporal length it is capable to affect the presentation of other components). Thus, a designer may capture temporal uncertainties through the use of fuzzy numbers, such as "about 5min" or "below 10s", and vague relations between temporal objects may be represented through the employment of both types of fuzzy duration. The membership functions associated to such fuzzy numbers may, as well, be altered via fuzzy modifiers, such as those illustrated in Table 1. Thus, assuming that "about 10s" is characterized by a triangular-shape fuzzy number $Z(m,a,b)$, then "almost about 10s", $V(n,c,d)$, is possible to be grabbed by the following computation, whose result is shown in Figure 3.

$$\text{“}almost\ about\ 10s\text{”} = V(n,c,d) = Dil\_L(Z(m,a,b)) = Z(m,a,b)^{0.5}$$

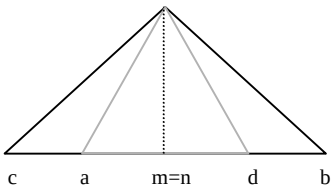

Figure 3. Result of the calculus of a temporal triangular fuzzy number (normalized).

It is also possible to place fuzzy arithmetic operations between fuzzy temporal objects to express inexact or flexible constraint relations between them. Thus, in *MuSCoF*, we define a *fuzzy temporal constraint* as a constraint in which at least one of its elements is quantified by a fuzzy value. This may be described as $event_1$ *{<, =, >}* $event_2$ *{⊕, Θ, ⊗} delay*, where ⊕, Θ, ⊗ refer to fuzzy addition, subtraction, and multiplication [Pedrycz 1998], and $event_i$ and *delay* represent *fuzzy events* and *fuzzy timepoints*. In this context, it is feasible to ascertain that “the media object O_1 should startup roughly two seconds after the end of media object O_2 presentation”: “startup” and “end” are fuzzy events and “roughly two seconds” is a fuzzy timepoint.

To give more flexibility to the specification of temporal interdependencies in a multimedia scenario, the duration Δ_p may, otherwise, be represented as a linguistic variable defined over a temporal scale. In *MuSCoF* model, we have taken such fact into account to represent presentation activities and phases. The designer is the one who stipulates the amount, the meaning, and the format of the membership functions associated to the linguistic terms ranging over each Δ_p partition (so, the Δ_p of each media object can be customized). The linguistic terms allow for the definition of time parcels representing possible synchronization points between two media presentations. In Figure 1, the duration Δ_p is associated with three linguistic terms (*beginning*, *middle*, *finish*), whose membership functions overlap, each representing a possible phase of rendering. In this case, P_1 and P_2 are two *fuzzy synchronization points*, each representing a subset of values delimited by the endpoints P_{ia} and P_{ib} given by

$$P_{1a} = \inf_{t \in \Delta p} \ middle \ \tau_1$$

$$P_{1b} = \sup_{t \in \Delta p} \ middle \ \tau_1$$

$$P_{2a} = \inf_{t \in \Delta p} \ finish \ \tau_2$$

$$P_{2b} = \sup_{t \in \Delta p} \ finish \ \tau_2$$

where *sup* and *min* refer to the *supremum* and *infimum* operations and

$$middle_{\tau_1} = \{ t \mid middle(t) \geq \tau_1 \}$$

$$finish_{\tau_2} = \{ t \mid finish(t) \geq \tau_2 \}$$

are the *α-cuts*[2] [Pedrycz 1998] of *middle* and *finish* in relation to the universe of discourse defined by Δ_p and to the *threshold* levels defined by τ_1 and τ_2, respectively. To directly specify a temporal interdependency between a pair of media objects, say $<O_1, O_2>$, the designer establishes a threshold value explicitly indicating the minimal and maximal values delimiting the fuzzy synchronization point for assuming that “O_1 has

[2] The α-cut of a fuzzy set *A* is another set containing those elements of *A* exceeding a threshold α.

fired a fuzzy synchronization event in relation to O_2". This strategy introduces uncertainty in the occurrence of a synchronization event (which is very apropos to capture the existence of unexpected holdups in the rendering application) and allows for the "relative firing" between different pairs of objects. Therefore, $<<O_1, finishing, almost, \tau_1> \rightarrow <O_2, start, state_2>>$ (e.g., O_2 will start when O_1 is "almost finishing") and $<<O_1, beginning, very, \tau_1> \rightarrow <O_3, start, state_3>>$ (e.g., O_3 will commence after the "very beginning" of O_1) should incur distinct synchronization points/events for the same media object O_1. Thus, it is possible to introduce and represent fuzzy causality between the activities realized by the media objects. Pazandak and Srivastava's temporal model [Pazandak 1994] also supports the notion of causality, albeit they do not envisage situations where the "cause-effect" may be fuzzified, as proposed here. For this purpose, we have made use of fuzzy conditional rules of the type

IF FUZZY TEMPORAL CONDITION THEN ACTION/DECISION,

where the temporal condition may be a fuzzy compound proposition and may be relative to each of the above three temporal objects, for expressing the "cause-effect" relations among media objects. In this sense, the triggering of a fuzzy associative rule may bring about the triggering of several other events (and rules, as consequence). The ordering of events occurrence, thus, is made part explicit (e.g., "startup of O_1 is about 5 min") and part implicit (unpredictable) to the execution of the presentation script (e.g., "IF some event occurs, like an user input or a media object has achieved a certain presentation phase, THEN some actions may be performed"). Therefore, both the whole scenario presentation duration as well as what should be presented along the time may not be once entirely deterministic and depends on the peculiarities of each execution.

According to Figure 2, the first step of the methodology is the user's scenario specification through a descriptive plan, which is typically done through the employment of a high-level script [Blakowski 1996] with some simple "command-like" directives. Elements of such script are activities and subscripts (each one relating to a scene/frame of execution). The idea is to provide for the user a set of basic temporal *primitives* (each relating to one of the above types of fuzzy relationships) he/she can overly employ and combine to depict the scenes and interactions behind a MMDoc execution. The script may be supported by a full programming or declarative language [Rutledge 1999] and should have its execution interpreted by the multimedia presentation tool. In *MuSCoF*, each high-level script is associated to an *abstract* multimedia component, which encapsulates, through a nesting hierarchy, all the scenario elements. In the same manner, a subscript contains media elements that are temporal correlated in a given fragment/scene of the exhibition. Table 2 brings the syntax and meaning of *MuSCoF* temporal primitives, as well as some logical, layout, and additional operators. The **cond** operator (which may be extended to encompass other, not yet contemplated, cases) expresses extra requirements regarding the MMComps' state that must be "on-the-fly" verified before their exhibition. Such operators should come adjacent to the MMComps definition or to the specification of the fuzzy rules/constraints. Hence, it is possible to represent such exceptional cases where (*i*) "two MMComps should have their presentation resumed together if one detects in runtime that the other is also being rendered" [Antonacci 2000]; or (*ii*) "a MMComp should startup after a second one but ought not be rendered together with a third one"; or (*iii*) "a MMComp should not be presented if any element is in an 'out-of-function' state." For case (*i*), the compound directive could be ***fuzzyCausal***$(<A, finish, [H], \tau_1> \rightarrow <B, stop, state_B>)$ ***cond***$(<playing, A> AND <playing, B>)$, whereas for case (*ii*) one could

specify that ***fuzzyCausal***(<*C, finish, [H],* τ_1> $\rightarrow$ <*A, start, state*$_A$> ***cond***(*NOT*<*playing, B*>)*.* In Section 4, some of these primitives are applied for the description of a scenario.

	Primitive Syntax	Primitive Semantics
Logical / layout	**script**(MM)	Implies the beginning of a (sub)script. *MM* is an abstract container
	sensor(sens)	*sens* is a multimedia sensor
	mmcomp(MM)	*MM* is a multimedia component
	contains(MM0, MM1)	*MM1* is encapsulated in *MM0*
	peer(MM0, MM1)	*MM0* and *MM1* are in the same container (are temporal collocated)
	cycle(MM, gen)	Indicates that *MM* should have its presentation iteratively presented for *gen* cycles
Temporal	**crispTimePoint**(CT)	*CT* is a timepoint with crisp (non-fuzzy) value
	fuzzyTimePoint(FT, pert)	*FT* is a fuzzy timepoint with membership function *pert*
	fuzzyEvent(e, t, MM, action)	*e* is an endpoint event related to the action *action* of MM (*t* indicates whether it is a begin or end event)
	fuzzyLingVar(V, u, {p})	*V* is a fuzzy linguistic variable with universe of discourse *u* and has one or more fuzzy timepoints (*p*) as labels for linguistic terms
	crispPresDur (MM, ps, ct)	*MM* has a presentation object *ps* with crisp duration *ct*
	fuzzyPresDur (MM, ps, ft)	*MM* has a presentation object *ps* with fuzzy duration *ft*
	fuzzyPresDur (MM, ps,V)	*MM* has a presentation object *ps* with duration (*V*) fuzzified
	crispLiveDur(MM, t)	*MM* has a crisp liveness duration (*t* is a crisp timepoint)
	fuzzyLiveDur(MM, t)	*MM* has a fuzzy liveness duration (*t* is a fuzzy timepoint)
	fuzzyCausal (<MM0, p, [H], τ> $\rightarrow$ <MM1, action, state>)	This should be read as "IF MM0 reaches the synchronization point delimited by the ling. term *p*, the set of zero or more modifiers *H* and threshold τ, THEN perform action *action* of MM1 and changes its state to *state*"
	fuzzyConstr(e1, e2, d, o)	Indicates a fuzzy temporal constraint where *e1* and *e2* are fuzzy events, *d* is a fuzzy delay and *o* is a fuzzy operator
	sensorSignal(sens, <MM,action,st>)	Denotes that the action manager relative to action *action* of MM should be notified by sensor *sens* whenever it is active
Other	**cond**(<MM1,st1> AND/OR/NOT <MM2,st2>)	This primitive indicates other kinds of presentation restrictions that must be satisfied every time some MMComps are going to be presented (MM1 and MM2 are two objects in states *st1* and *st2*, respectively, and AND/NOT/OR are logical connectives
	at_cycle(MM, gen)	This specifies an additional refinement to a temporal primitive, by indicating the cycle *gen* at which the fuzzy causal rule or constraint ought to be handled

Table 2. *MuSCoF* primitives for MMDoc temporal specification.

Once a scenario has been created, there should be some mechanism to verify the correctness of the specification, checking whether it matches the intended schedule. This is the role of the *parsing* step of the methodology, which should also be supported by the multimedia tool/environment. One possible way is to assist the user by playing back (parts of) the document, although this can be cumbersome or ineffective (scenarios have the potential to become very large and complicated as the degree of concurrency and uncertainty increases). Another choice is to automate the process, which, conversely, typically fails to foresee all the non-deterministic possibilities behind the fuzzy causal rules/constraints (there may be hundreds of them). The solution should be a mixture of both; whatever, it must detect inconsistencies among the objects' rendering. (A multimedia object is said "unrenderable" if two or more rules relating to it are contradictory one another, implying, e.g., that the end of presentation of an object occur before the start of the same one.) Additionally, the parsing process ought to indicate

critical points of entanglement where the uncertainty aspects of the model (e.g., fuzzy temporal duration) should be refined or revisited for eliminating ambiguities during the runtime presentation. Finally, another role of the parsing step is to provide the designer with the means to interactively fine-tune the parameters of the fuzzy script directives, such as the thresholds and membership function shapes/partitions.

3.2. Multimedia Component Model

Typically, the MMDoc creation involves searching for media data in an information repository or capturing them through special equipment, and then organizing them to convey the intended use. Once located/produced, the multimedia contents can be assembled into an abstract structure compliant to some global logical framework and also be assigned to some region of a display device according to a layout structure [Karmouch 1996]. Following the guidelines from Gibbs' work [Gibbs 1991], in *MuSCoF* we assume a logical model wherein every MMDoc entity is modeled after a particular type of component (an extension of the *Scene_Comp* abstract class), and these components should be logically arranged through composite constructs (**contains**/**peer** primitives in Table 2) and communicate one to the other through the interchange of events. In our framework, there are two types of scene elements already modeled, MMComp and MMSensor. The last comprehends special kinds of components situated in a virtual display (that is, with no rendering features), which are created specifically to cope with user interactivity. This class of elements should be extended if one wishes to provide support for novel interaction styles, typically via special input devices. Figure 4 shows the communication infrastructure necessary (e.g., event brokerage) for the elements' runtime location and for the conveyance of their synchronization events.

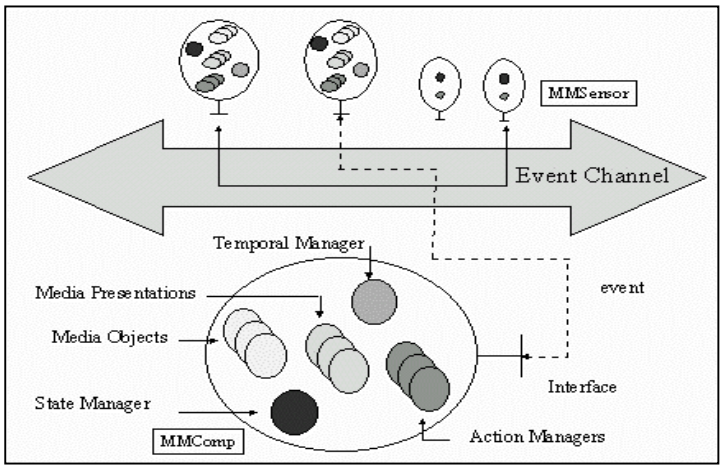

Figure 4: Interactions underlying the MuSCoF multimedia component model.

MMComps can be seen as aggregations of two types of elementary units, those relating to media contents and those concerning management tasks. Media objects represent pieces of raw (binary) information classified into pairs of MIME types/subtypes [Borenstein 1993] and may be associated to one or more media presentation objects, which, in turn, are presentation versions containing rendering and layout information. Hence, it is possible to have associated to the same MMComp one or more media objects (all relating to the same semantics contents but with dissimilar formats, like *gif*, *jpeg*) whose exhibitions are customized by the choice of specific presentation objects. This decision allows that the same MMComp may be, in different

times, associated to distinct (fuzzy) duration intervals (hence, the fuzzy synchronization points assume a relative character), therefore justifying the high-level, event-based synchronization decoupling posed here.

The *temporal manager* is the unit responsible for maintaining, interpreting, and executing all temporal directives associated to the MMComp it represents. For such, it should be configured by the parsing step of *MuSCoF* with a base of event rules containing all the information necessary for the triggering of synchronization events it may produce along its existence. Such information typically indicates when and for who a synchronization message should be forwarded. This manager keeps in charge of running two clocks, for presentation and active influence purposes respectively. The presentation clock is a kind of timeline marked with synchronization points to be fired out for the communication with other components, following a "producer-consumer" scheme. Once a sync point is reached, the temporal manager should broadcast a notice towards those *action managers* (of one or more MMComps) which are listening to a particular channel of the communication broker. Temporal managers export (through the MMComp interface) operations permitting the online configuration of their clocks through invocations coming from their peer action managers. Two instances of temporal managers (from distinct MMComps) may have their clocks adjusted to run in the same temporal pace for fine-grain synchronization purposes [Pazandak 1994]. As well, such strategy allows that conditional restrictions (Table 2) of the form "if two MMComps are somehow being concurrently rendered and at least one of them detects such fact, then harmonize their presentation clocks in such a way that they can finish their exhibition together" [Antonacci 2000] can be properly fulfilled.

Another kind of management mechanism relates to the MMComp state maintenance. In *MuSCoF*, there are two orthogonal means of classifying a MMComp state. In the first categorization, a MMComp may be in one of three states: *passive*, *active*, or *indefinite*. A passive entity has no capability of influencing the behavior/execution of a peer, but may be, otherwise, swayed by the activities produced by other active components. Active components (furthering Gibbs' conception [Gibbs 1991]) may have their presentation duration Δ_p fuzzified and assigned with fuzzy synchronization points. In some cases, it is not important or possible to discriminate whether a MMComp state is passive or active, so it is labeled as indefinite.

The second type of categorization sets the MMComps apart into the following (possibly extensible) situations: *alive*, *expired*, *out-of-function*, or *indefinite*. The last category keeps the same semantics as discussed before. A MMComp in alive state allows for another style of synchronization through the employment and fuzzification of its associated liveness duration Δ_l. Such variable refers to the temporal interval its "physical presence" may influence the presentation of other scene elements. Thus, even inert in the sense of rendering, this kind of component may still affect the state/behavior of other elements (this is the case of any MMSensor). Alive components may be active or passive and have their state classification changed along its execution, which is done through invocations coming from peer action managers (see **fuzzyCausal** primitive in Table 2). For this purpose, the state manager exports operations allowing for runtime state settings. MMComps are said "expired" when their existence/exhibition do not have anymore the power to influence other entities, and are said "out-of-function" when either its existence or rendering have been hampered by some malfunction behavior of the multimedia environment infrastructure (e.g., a loudspeaker fault). The parsing step

of *MuSCoF* interprets the user descriptive plan and then sets the initial states of all MMComps according to the two aforementioned classification types.

The last kind of management element underlying the *MuSCoF* component model constitutes the action managers. They cope with all activities realized by a particular MMComp in respect to the behavior perceived by the other scene elements. Some actions have well-known semantics, such as VCR operations (start, stop, pause, resume), and may be already supported by the multimedia environment, whereas others may be conceived by the designer for specific purposes. The separation of temporal, state, and action managers promotes a range of interesting functionalities to the *MuSCoF* component model: (*i*) it allows two or more managers being assigned to parallel threads; (*ii*) it lessens the bad side-effects caused by runtime faults in the managers; and (*iii*) it introduces another level of non-determinism, by allowing such situations where two or more actions are concurrently trying to change the temporal/state conditions of the same MMComp. Figure 5 shows the OMT static model for the *MuSCoF* component design.

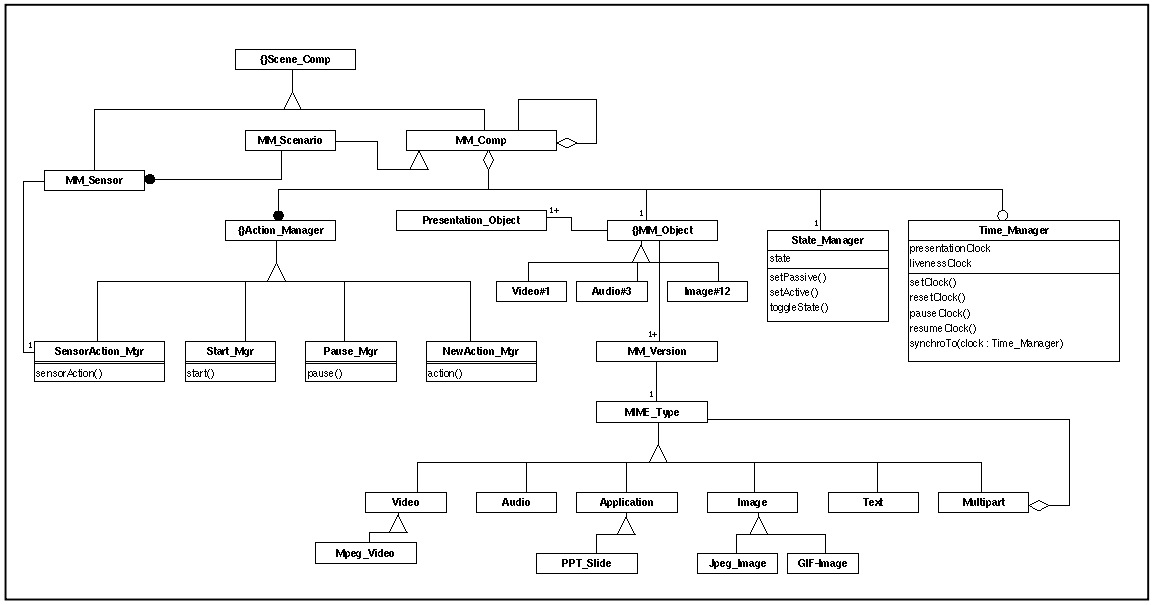

Figure 5. Classes diagram for *MuSCoF* multimedia component model.

4. Example

In this section, we present an example of specification for a simple interactive multimedia presentation. The example focuses on fuzzy relations among some media objects and on the support to user interaction, which may influence the presentation's execution cycle. As illustrated in Figure 6, the presentation is composed of four general phases (states). *State A* indicates that the presentation is turned off. *State B* implies that it is turned on, but with no video and audio being presented. *State C* indicates that the user has inserted a video-card in the virtual monitor and a video is being presented without legend. *State D* denotes that a video is presented with legend. Each phase ought to be described through a descriptive plan provided with a **script** directive. In this scenario, all phases are composed by the same scene elements, that is, they have the same logical/layout design (but different temporal interrelations), which should be somewhat like follows: {**script**(MM0); **sensor**(mouse_sensor); **mmcomp**(video); **mmcomp**(audio); **mmcomp**(legend); **contains**(MM0, mouse_sensor); **contains**(MM0,

video); **contains**(MM0, audio); **contains**(MM0, legend);}. The user may execute six actions capable of performing transitions between the system's states: *1)* turn the presentation on (causing the state change A ⇒ B); *2)* turn the presentation off (causing one of the following state transitions, B ⇒ A, C ⇒ A, or D ⇒ A); *3)* choose a video (and associated audio) to play, by inserting a card in the virtual monitor (state change B ⇒ C); *4)* deactivate the video, by ejecting the card (C ⇒ B, or D ⇒ B); *5)* insert the legend (C ⇒ D); and *6)* remove the legend (D ⇒ C). Additionally, the user may execute regular VCR actions (pause, forward, rewind) on each media object (video, audio, and legend) separately, which should be supported by specific instances of action managers.

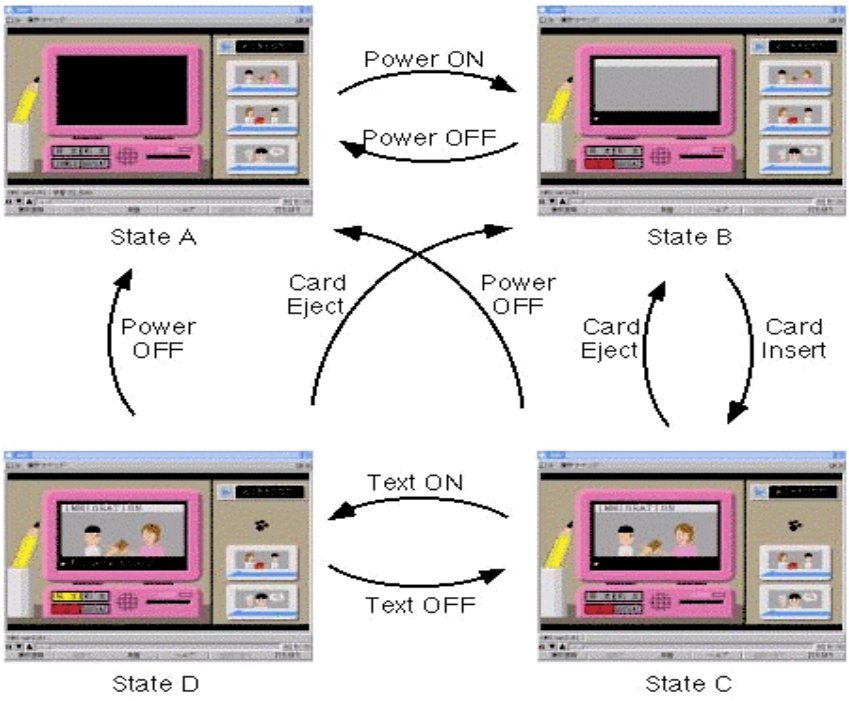

Figure 6. Execution cycle of a multimedia presentation with user interaction.

In the above layout specification, there appears a multimedia sensor (*mouse_sensor*) that keeps in charge of user control input. This sensor has a different *SensorAction_Mgr* instance (see Figure 5) for each presentation frame, thus performing different activities in accordance with the current system phase. It is through this sensor's action manager that the presentation of the MMComps of the scenario is started up. For instance, moving from State B (all elements already created but none of them in active state) to State C (video and audio being concurrently exhibited) implies that the *SensorAction_Mgr* has signaled both the audio and video start managers for commencing their presentation.

When the user inserts a card in the virtual monitor, audio and video should have their exhibition initialized together. Although, in the case of delays (or failures) in the sensor's notifications, this may not be the case, and, thus, the designer should specify another level of causal association between both MMComps. A possible fuzzy causal rule for this situation is that the audio starts "quite in the beginning" of video presentation, giving rise to **fuzzyCausal**(<video, beginning, "quite", τ)> → <audio, start, active>). As well, when the user inserts the legend in the video (C ⇒ D), the text presented may change its color according to the video's playback state (beginning,

middle, finish). This may be captured via three causal rules of the form **fuzzyCausal**(<video, phase, , τ)> $\rightarrow$ <*legend*, change_color, toggle_state>), one for each rendering phase. Finally, when one decides to specify the power-off of the whole presentation by coordinating the sequential termination of all scene elements, the following piece of plan description could be conceived {**fuzzyEvent**(e1, end_event, video, stop); **fuzzyEvent**(e2, end_event, audio, stop); **fuzzyEvent**(e3, end_event, legend, stop); **fuzzyTimePoint**("about 100ms", triang(100,95,105)); **fuzzyConstr**(e2, e1, "about 100ms", $\oplus$); **fuzzyConstr**(e3, e2, "about 100ms", $\oplus$);}.

5. Related Work

Currently, there are some new "driving forces" that ought to be observed for the conception of more effective synchronization models [Antonacci 2000, Karmouch 1996, Pazandak 1994]: *expressiveness* (referring to the types and number of temporal orderings that can be expressed in a model), *simplicity* (can authors readily create MMDocs that are clearly and concisely structured and easily interpreted, understood, and modified?), *flexibility* (e.g., in the definition of the coordination rules and/or in the maintenance of MMDoc temporal consistency) and *support to interactivity and uncertainty*. The combination of the expedients offered by fuzzy and component systems is advocated here as a promising solution towards such accomplishment.

Huang and Wang (1998) have devised a methodology towards the synchronization of multimedia elements regarding the effects of user interaction. The authors employ a finite state machine extension (very akin to Petri nets with delays and priorities [Raposo 2001]) for synchronization control purposes, and investigate on how the user interaction during real-time presentation might alter the temporal relationships among the media objects. However, a drawback of such specification is that it is too much complex for inexperienced designers grasping its peculiarities. It is our sentiment that, by employing fuzzy sets reasoning, the multimedia specification comes to be more tractable as well as flexible enough to incorporate subjective aspects.

Mediadoc [Karmouch 1996], a document architecture for multimedia information, supports the creation of MMDocs through well-delineated (logical and layout) structures and provides proper instruments for the detection and prevention of illegal temporal specifications during the scenarios correctness verification. Yet simple, its provision to synchronization specification does not cogitate any kind of activity coming from external entities nor supports uncertain temporal duration nor allows the decoupling and reuse of media elements into different scenarios.

Ali *et al.* (2000) have considered the problem of multimedia synchronization in a Web environment and proposed a neurofuzzy controller as the scheduling device responsible for the maintenance of the workload in a multimedia Web server. The fuzzy controller has its fuzzy rule base constructed by means of a hybrid learning algorithm that both defines the rules and adjusts their membership functions. Such initiative, although very apropos to distributed multimedia purposes, does not take into account the requirements imposed by application-layer, MMDoc temporal specifications. By other means, Zhou and Murata (1998) have made use of a fuzzy Petri net (FPN) approach for the uncertainty modeling of synchronization events in a distributed environment. Although somewhat similar to the philosophy presented here, their work has not centered upon the multimedia specification requirements imposed by MMDoc elements, being mostly oriented to performance analysis and evaluation.

Our approach has characteristics borrowed from most of the models shown in Section 2.3: It is based on the production-consumption of punctual events and on the definition of fuzzy causality/restriction rules that determine their semantics, provides decentralized events ordering through the encapsulation of separate timelines into each MMComp, and also brings support to the hierarchical assembling of media components into composites, as their decoupling allows for the combination of sequential/parallel events that may be occurring in distinct levels of the same logical structure.

6. Final Remarks

This paper introduces a novel methodology that uses concepts of fuzzy sets and software components for the temporal arrangement of multimedia document elements. As we showed in our multimedia synchronization model, the use of fuzzy causal rules and constraints allows for the creation of high-level descriptive plans for the design of multimedia presentations. Among the advantages of this model, we can highlight the support to the unpredictability of human interaction [Huang 1998] and the enhancement of the understandability of specification scripts (besides dismissing a rigid formal specification, fuzzy rules are more closely related to human reasoning).

Synchronization mechanisms based on software components generate a decentralized coordination apparatus (following an event-based approach) that enhances the scalability of the model. Moreover, the same MMComp may be reused in different MMDoc scenarios, may be composed with others for the creation of composites, and, as well, may aggregate several media objects (e.g., different formats for a video) and their related media presentations (e.g., distinct resolutions) [Antonacci 2000]. The separation between temporal and action managers provides a further degree of flexibility to the methodology, in the sense that (*i*) each MMComp may have a different duration interval associated to each of its media presentations (hence, each MMComp may have customized its synchronization aspects with other entities through different synchronization points); and (*ii*) each action manager should be specialized to a particular task (customized by the application designer), whose effects may dynamically cause changes in the state and temporal rendering of its associated MMComp. Another interesting feature of the decentralized control is that it supports unpredictable events, such as the user interaction or device failures, without affecting the whole presentation.

As future work, a better formalization of the fuzzy synchronization model is required, to fully evaluate its advantages and discover some possible drawbacks. A declarative language based upon this model (e.g., an XML/SMIL extension [Antonacci 2000, Rutledge 1999) would ease both the portability and the creation of the descriptive plans, as it might provide high-level constructs to implement the synchronization primitives addressed here. Moreover, additional managers for dealing with other kinds of interdependencies (e.g., spatial synchronization [Kwon 1999]) could be devised. As well, it seems viable to employ FPNs for the modeling of the behavior of both temporal and action managers [Raposo 2001, Zhou 1998] as well as for extending some other Petri net-based approaches for temporal specification (e.g., OCPN [Little 1990]).

Acknowledgements. The first and second authors are sponsored by CNPq (140719/1999-7) and FAPESP (00/10247-3), respectively.

References

Ali, Z., Ghafoor, A. and George Lee, C. S. (2000) “Media Synchronization in Multimedia Web using a Neuro-fuzzy Framework”, *IEEE JSAC*, 18(2), p.168-183.

Antonacci, M., Saade, D., Rodrigues, R. and Soares, L. F. G. (2000) “Improving the Expressi-veness of XML-Based Hypermedia Authoring Languages”, *Procs. of MMM*, p. 71-88.

Blakowski G. and Steinmetz R. (1996) “A Media Synchronization Survey: Reference Model, Specification, and Case Studies”, *IEEE JSAC*, 14(1), p. 5-35.

Borenstein, N. (1993) “MIME: A Portable and Robust Multimedia Format for Internet Mail”, *ACM Multimedia Systems* Journal, 1(1), p. 29-36.

Gibbs, G. (1991) “Composite Multimedia and Active Objects”, *Procs. of OOPSLA*, p. 97-112.

Hopkins, J. (2000) “Component Primer”, *CACM*, 43(10), p.27-38.

Huang, C-M. and Wang, C. (1998) “Synchronization for Interactive Multimedia Presentations”, *IEEE Multimedia*, 5(4), p. 44-62.

Lewandowski, S. (1998) “Frameworks for Component-based Client/Server Computing”, *ACM Computing Surveys*, 30(1), p. 3-27.

Little, T. and Ghafoor, A. (1990) “Synchronization and Storage Models for Multimedia Objects”, *IEEE JSAC*, 8(3), p. 413-427.

Karmouch, A. and Emery, J. (1996) “A Playback Schedule Model for Multimedia Documents”, *IEEE Multimedia*, 3(1), p. 50-61.

Kwon, Y-M., Ferrari, E. and Bertino, E. (1999) “Modeling Spatio-temporal Constraints for Multimedia Objects”, *Data & Knowledge Engineering*, 30, p.217-238.

Pazandak, P. and Srivastava, J. (1994) “A Multimedia Temporal Specification Model and Language”, *Tech. Report 94-33*, Dept. Comp. Science, Univ. Minnesota.

Pedrycz, W. and Gomide, F. (1998) *An Introduction to Fuzzy Sets: Analysis and Design*, MIT Press.

Raposo, A. B., Coelho A. L. V., Magalhães, L. P. and Ricarte, I. L. M. (2001) “Using Fuzzy Petri Nets to Coordinate Collaborative Activities”, *Procs. of IFSA/NAFIPS*, p. 1494-1499.

Rutledge, L., Hardman, L. and van Ossenbruggen, J. (1999) “Evaluating SMIL: Three User Case Studies”, *Procs. of ACM Multimedia*, p. 171-174.

Steinmetz, R. (1990) “Synchronization Properties in Multimedia Systems”, *IEEE JSAC*, 8(3), p. 401-412.

Wahl, T. and Rothermel, K. (1994) “Representing Time in Multimedia Systems”, *Procs. of ICMCS*, p. 538-543.

Zhou, Y. and Murata, T. (1998) “Fuzzy-Timing Petri Net Model for Distributed Multimedia Synchronization”, *Procs. of IEEE SMC*, p. 244-249.