Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 373 – 382

ISSN 1896-7094
© 2006 PIPS

RT-UML in modeling of multimedia applications

Paweł Cichoń[1]

[1] Instytut Informatyki Stosowanej, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław
pawel.cichon@pwr.wroc.pl

Abstract. RT-UML enables a structural and behavioral description of multimedia applications including time characteristic, however it does not offer mechanisms of a multimedia objects presentation (arrangement) expression or the time constrains expression between events. This fact illustrates RT-UML lack of expressive capabilities, especially in the context of multimedia applications modeling. The presented in the paper extension of RT-UML illustrates how multimedia objects arrangement and events synchronizations can be presented, which means new type of diagrams addition to UML (presentation diagrams) and a synchronized events set class placing into the time model of RT-UML. Moreover extensions provide an application a graphical presentation of synchronized events to sequence and activity diagrams of UML, which enables time constrains of multimedia objects and their activities expression (synchronization of time events, which occur during presentation of multimedia objects). The originality of this approach relies on the extension of RT-UML syntax and on the presentation of an original method of multimedia applications modeling, which can make the production process more formalized and thus more precise.

1 Introduction

A multimedia application connects various forms of media, of which at least one is dependent on precise timing. In the context of the real time a multimedia application is defined as a set of connected scenes organized into a real time system. A scene is a set of presentations of multimedia elements whose behavior leads to, and impacts, other scenes [4, 5]. The multimedia elements within a given scene are composed of media objects (Fig. 1) and are influenced by various processes, but also could be open for external interactions from an user (Fig. 2). The processes connect media objects to one another and serve as an intermediary that facilitates data communication between them and between their environment. Additionally, the processes reflect the critical time dependencies that exist between elements of the multimedia application.

The state of a given scene is a snapshot of the current set of processes and multimedia elements at that instant. The state of a multimedia application is understood as the state of the active scene. To traverse between successive states of a multimedia application is known as a transformation, or going from one scene to another (Fig.3). The array of all states and transitions between scenes makes up the scenario of a multimedia application [5].

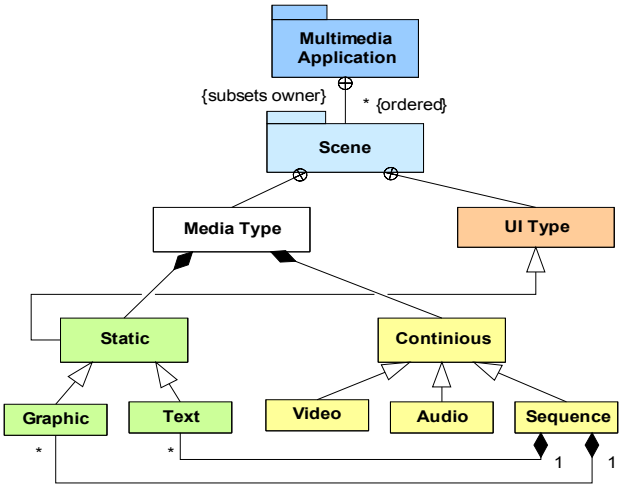

Fig. 1. The structure of a multimedia application's components with a hierarchy of multimedia classes types (Media and User Interface types)

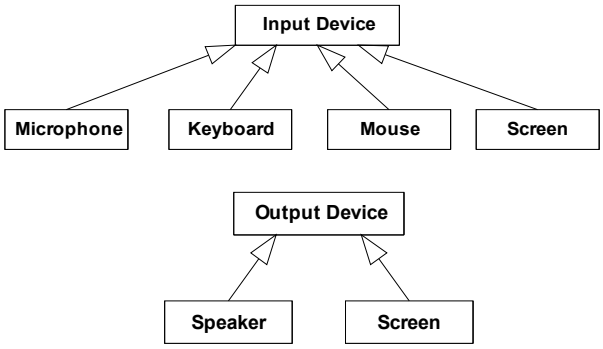

Fig. 2. The general classification of input-output devices

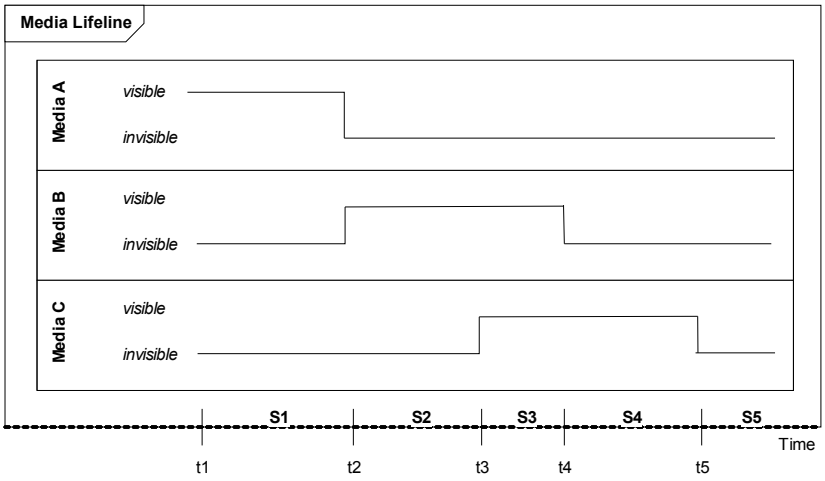

Fig. 3. The transition between scenes of a multimedia application. t1…t5 – Instances, which refer to occurrences of timed events and to the state of a multimedia application at a given point of time. S1…S5 – successive scenes of a multimedia application. Media A…Media C – life cycle (presentation) of multimedia objects

A real time multimedia application's life cycle consists of the object oriented presentations, which occur in defined periods and time limits. The multimedia objects can initiate events, which are handled by a multimedia application. An event is a specific phenomenon, which occurs in the time and space (inside or outside of a multimedia application). The RT-UML profile (*UML Profile for Schedulability, Performance, and Time Specification, Ver.1.1*) [3, 4] defines the time model (Fig. 4, Fig. 5).

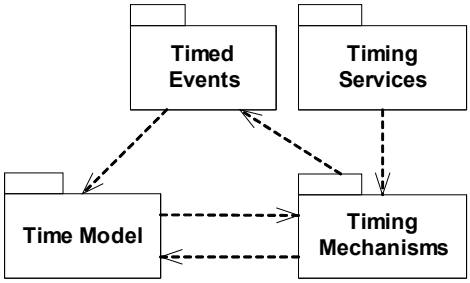

Fig. 4. The RT-UML modules

A time event (TimedEvent class) in RT-UML is an event, whose definition includes a time characteristic, it means the definition of time stamps [3]. A time stamp is a time value (TimeValue class included in TimedEvents package) (Fig.4, Fig.5). A time constraint is expressed by a time interval (TimeInterval class) (Fig.3), in which some events occur. The time constrains are divided into individual and group constraints. An individual constraint refers to a single event, while group constraints refer to a set of events.

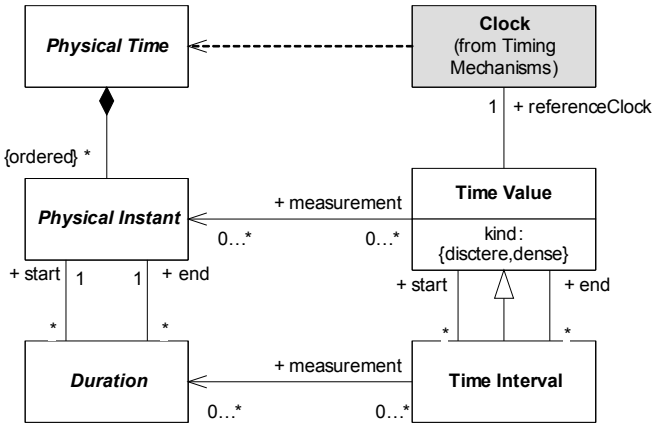

Fig.5. The conception of time modeling in RT-UML

RT-UML enables a structural and behavioral description of multimedia applications including time characteristic [5], however it does not offer mechanisms of a multimedia object's presentation (arrangement) expression or the time constrains expression between events. This fact illustrates RT-UML lack of expressive capabilities, especially in the context of multimedia applications modeling. The presented in the paper extension of RT-UML illustrates how multimedia objects arrangement and events synchronizations can be presented, which means new type of diagrams addition to UML (presentation diagrams) and a synchronized events set class placing into the time model of RT-UML. Moreover extensions provide an application a graphical presentation of synchronized events to sequence and activity diagrams of UML, which enables time constrains of multimedia objects and their activities expression (synchronization of time events, which occur during presentation of multimedia objects). The originality of this approach relies on the extension of RT-UML syntax

and on the presentation of an original method of multimedia applications modeling, which can make the production process more formalized and thus more precise.

2 Synchronization expression of multimedia application's events in RT-UML

A presentation of multimedia objects is based on time conditions, which means that events, which occur during an objects life cycle are dependant on time constraints. A single event z satisfies a time constraint, which is expressed by the time interval $[t_a, t_b]$, if a moment of its occurrence belongs to defined time interval ($z.timestamp \in [t_a, t_b]$) – an individual time constraint is satisfied this way. A set of events $A = \{z_1 ... z_n\}$, $n > 1$ satisfies a time constraint, if $\forall_{1 \le i \le n} z_i.timestamp \in [t_a, t_b]$ - a group time constraint is satisfied this way. A synchronization is a characteristic of an event, which relates to its occurrence in a specified time interval in relation to other events. This is expressed by a particular type of time constraint between events. A synchronization is divided into individual and group types. An individual synchronization relates to a single event and is determined by an individual time constraint. A group synchronization relates to a set of events and is determined by a group time constraint.

A single event z characterized by time constraint $[t_a, t_b]$ is synchronized per Δ_z level if and only if, $t_b - t_a \le \Delta_z$. Δ_z which is called the synchronization interval of a single event.

A set of events $A = \{z_1 ... z_n\}$, $n > 1$ is synchronized per Δ_z level if and only if $\max_{\substack{0 \le i \le n \\ 0 \le j \le n}} |z_i.timestamp - z_j.timestamp| \le \Delta_A$. Δ_A is called the synchronization interval of an events set.

If $startSynch(A) = \min\{z_1.timestamp, ..., z_n.timestamp\}$ - a function that evaluates the start time value of the synchronization interval within an events set

$\wedge$

$endSynch(A) = \max\{z_1.timestamp, ..., z_n.timestamp\}$ - a function that evaluates the end time value of the synchronization interval of an events set

then

$|startSynch(A) - endSynch(A)| \le \Delta_A$.

A notation representing the synchronization interval can be projected in single-argument form $\{duration\}$, when $\Delta = duration$ or in double-argument form, when $\Delta = t_2 - t_1$.

A single-argument form refers to a relative definition, while a double-argument form refers to an absolute definition of the synchronization interval (meaning that both the start and end time values of the synchronization interval are determined).

An absolute synchronization occurs when event, which synchronizes individually or set of events, which synchronize in group have defined timestamp (which refers synchronization's beginning) or absolute time constraint. A relative synchronization occurs if the timestamp is undefined and the time constraint is expressed relatively (we state only, that an event or an events set must synchronize in a defined time interval. There is no precise definition of the synchronization's beginning).

In order to express the synchronization of events, a new class is provided to the time model of RT-UML – a set of synchronized events (TimedEventsSet class) added to the TimedEvents package, which consists of timed events (TimedEvent class), which synchronize (Fig.6). A single event may belong to many sets of synchronized events. A synchronized event (besides the timestamp), has a defined set's name, to which its belongs (setName attribute in the TimedEventsSet class) and synchronization interval Δ (TimeInterval or [TimeValue, TimeValue]) (Fig. 5, 6).

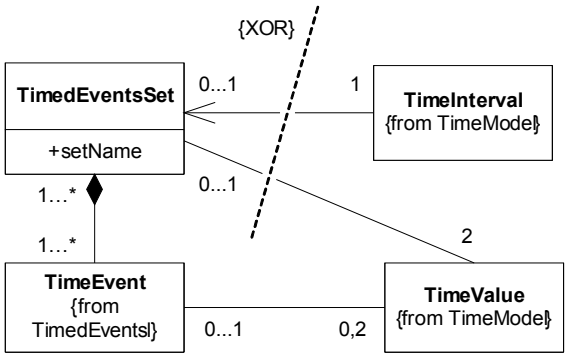

Fig. 6. The synchronized events set (extension of RT-UML)

A synchronized event can be divided to three distinct types based on the method of synchronization (individual or group) (Tab.1). A graphical representation of an individually synchronized event has a synchronization interval defined and an optional timestamp. A graphical representation of a synchronized event within a group has defined set's name, as well as an optional timestamp (TimeValue class) and a synchronization interval (TimeInterval class) (Fig. 5). A set of synchronized events should have at most one occurrence of synchronized event representation with defined synchronization interval, which refers to whole set.

Table 1. The notation basis of synchronized events*

Notation basis	Example	Type of synchronized event
t ○ { Δ }	1s ○ {3s}	An event, which synchronize individual
t ● setName1{ Δ_1 }, setName2, setName3{ Δ_2 }, …	● A {3s},C	An event, which synchronize in group
t ⊙ setName1{ Δ_1 }, setName2, { Δ_2 }, …	⊙ A {3s}, {1s}	An event, which synchronize both individual and in group

*where: t - timestamp, Δ - synchronization interval, setName – name of set of synchronized events

The presentation of these constructions is necessary to accurately express time constrains, which occur during the life cycle of a multimedia application. This is because they enable the indirect expression of both absolute, as well as relative synchronization of multimedia objects in a defined multimedia application's model, which consists of both a static model, which refers to its structure and a dynamic model, which presents its behaviors and refers directly to behavioral aspects of multimedia objects.

3 Static model – expression of structural aspects of multimedia objects

A multimedia object is represented in UML as an instant of a multimedia class or classes, whose attributes and operations describe both static and functional aspects of the object. In order to present the objects in a structured hierarchy, the description of structural aspects of the multimedia application's elements require the use of class diagrams. The introduction of new type of diagrams to UML - it means presentation diagrams is grounded by the need of mechanism, which allows to describe the arrangement, spacing and relative sizes of objects presented in a given scene. The class diagrams are used for the structured modeling of multimedia objects and the presentation diagrams resulting in a visual description of scenes (this is for the presentation layer description of a multimedia application).

Shifting gears, we will focus on a toy-example of a real module within a multimedia application developed for Washington Gas (utilizing a video media player, which is created to run in an Internet-based environment) (Fig. 7). This example was used to present the use of the previously described RT-UML extensions (Fig. 6). The examples presented are simplified parts of the original model of the Washington Gas application, without any specification of multimedia class attributes (formats and parameters of multimedia classes) or operations (methods of multimedia classes), which would require a reference to the implementation environment (object oriented programming language).

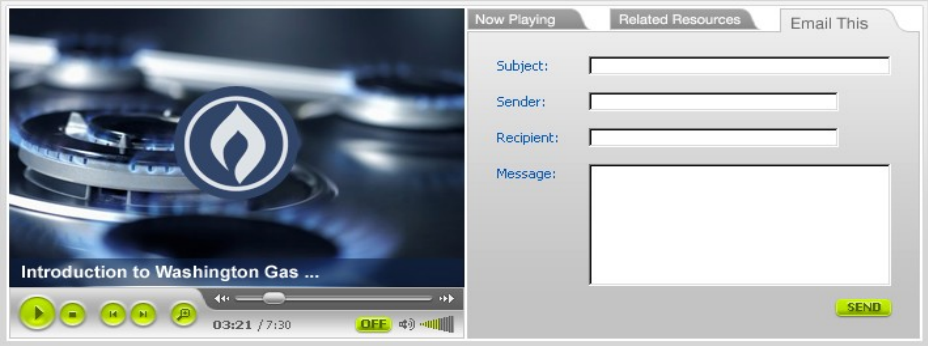

Fig.7. The example of chosen module of multimedia application (Washington Gas Media Player)

3.1 Class diagrams – structure modeling of multimedia application

The class diagrams are used to define the application's structure description. A class diagram consists of a class definition and the bonds set between classes (associations), which describe a structure of multimedia objects and their structural dependencies (relations) one another. The class diagrams are expressive enough to satisfy the demands required for proper architecture used in multimedia application modeling. A static model of a multimedia application should include the definition of media classes types (Fig. 1), a structure and hierarchy of multimedia classes definitions and a description of the interactions between multimedia classes instances, meaning the interactions between multimedia objects and between an environment. The class diagrams are used to:

- define various media types (definitions of classes for all media types) (Fig. 1),
- structured modeling and the description of classes (objects) within a multimedia applications hierarchy (which consists of classes description together with the definition of relationships which occur between them) (Fig. 8, 9, 10),

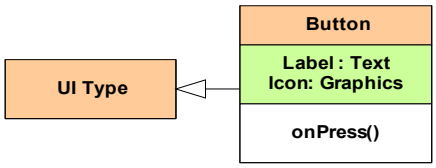

Fig. 8. An example of multimedia class with attributes and operations definition

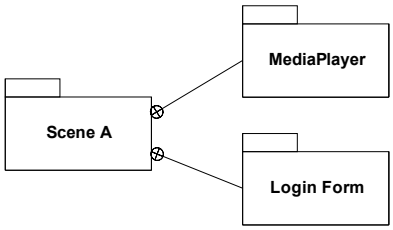

Fig. 9. A structure of a multimedia application's scene

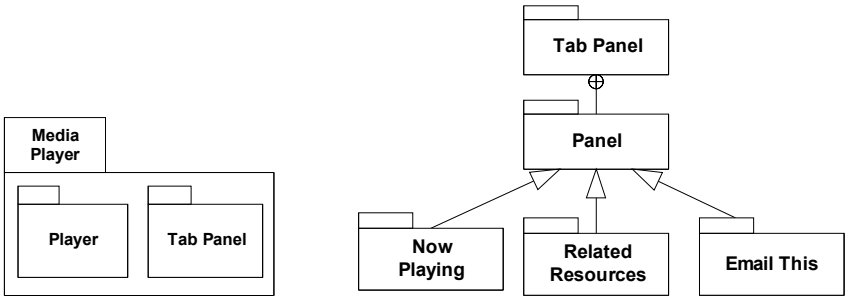

Fig. 10. A structure of chosen module of a multimedia application's scene

3.2 Presentation diagrams—modeling of user interface of multimedia application

To provide both a precise and general method of modeling new type of diagrams using static aspects of the multimedia application within RT-UML, allows the presentation diagram, which is the description of the arrangement of multimedia objects in an application's scene, to model the user interface, including some graphical aspects such as relative sizes and positions. The presentation diagram is used for the layout of the multimedia application modeling utilizing UML packages and objects. Objects are divided into passive and active objects. The passive objects are instances of media classes, which are not open in opposition to active objects for user interaction. The active objects have communication interfaces. Examples of active objects are buttons, inputs, scrollbars, etc.

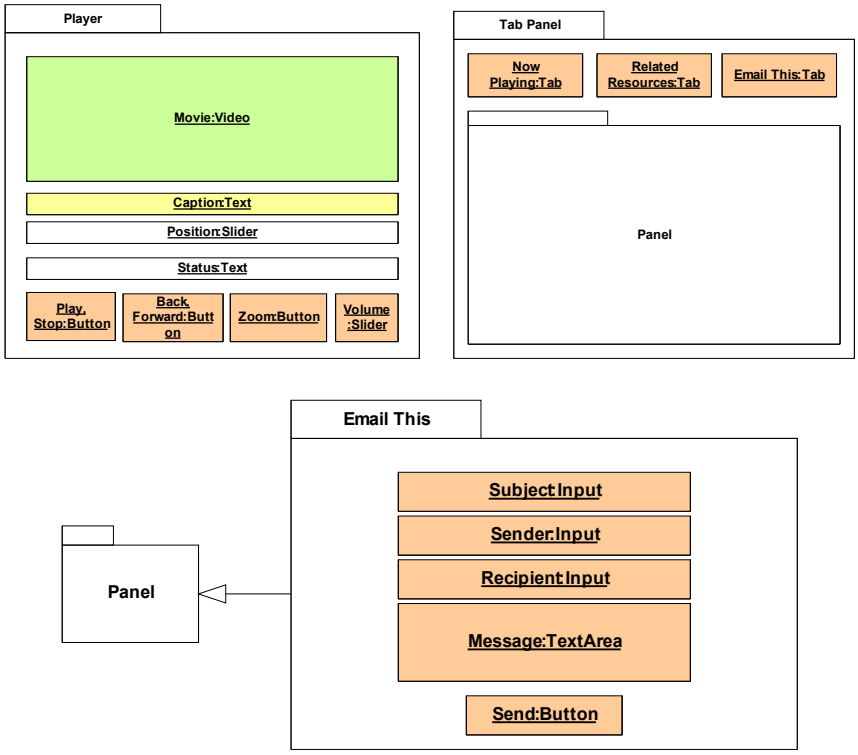

Fig. 11. An example of application of presentation diagrams to modeling of chosen module of multimedia application's scene

4 Dynamic model—expression of behavioural aspects of multimedia objects

A behavior of a multimedia application is composed of both sequential or parallel presentations of multimedia objects, as well as scenes and interactions, which occur between objects and between an environment. Both the presentations and interactions of objects may occur in an asynchronous or synchronous fashion meaning that for behavioral aspects of multimedia applications, modeling the construction of a time constrains expression is necessary, which states the absolute or relative synchronization of timed events. The dynamic model of describing multimedia applications uses the sequence and activity diagrams of UML, extended by mechanisms of synchronization expression.

4.1 Sequence diagrams—intercations modeling between multimedia objects

In the preceding chapter a set of synchronized events was defined, which includes timed events, allowing a graphical representation to be applied to the sequence and activity diagrams thus making the expression of multimedia objects synchronization possible.

The sequence diagrams [4] are used for the description of interactions between multimedia objects [1]. Individual synchronization, in the context of sequence diagrams, refers to a single event on the multimedia object's timeline synchronization. At the same time, group synchronization refers to events sets on multimedia objects' timelines. (Fig. 12). A timed event, which occurs on object's timeline refers to a moment of calling or receiving some action or message. On Fig. 12 there is an example of the use of graphical notation of synchronized events in sequence diagrams. This shows which individually synchronized events a, b, c and set of in groups synchronized events $Fade = \{c, d\}$ are marked.

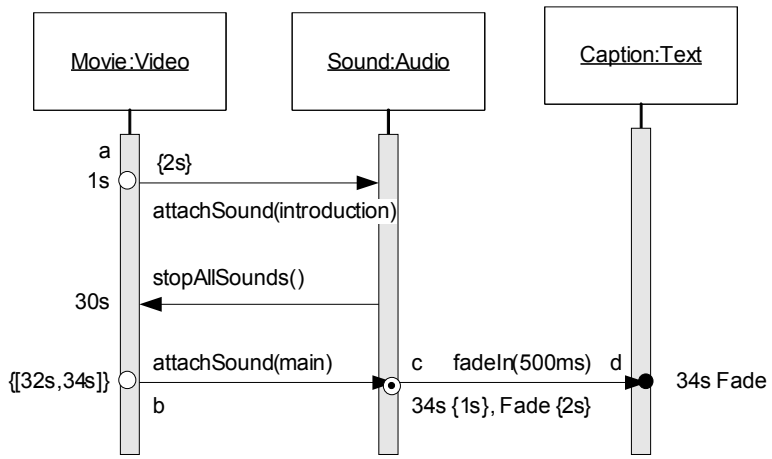

Fig. 12. An example of sequence diagrams with marked synchronized events

The *attachSound(introduction)* operation must be called within 3 s of the *Movie* object's life cycle (absolute and individual synchronization of event a). At the earliest this can occur at the 32 s mark, and at the latest in 34 s of *Movie* object life cycle, the *attachSound(main)* operation must be called (individual synchronization of event *b* with absolutely expressed time constraint {[32s, 34s]}). Once this occurs, by the 35 s mark of *Sound* object life cycle operation *fadeIn(500ms)* must be called (absolute, individual and in group synchronization of event c, which belongs to *Fade* set of in group synchronized events). The events *c* and *d* synchronize in group in 1 s interval in relation to 34 s of *Caption* object life cycle.

4.2 Activity diagrams—control and data flows inside objects and between activities of multimedia objects

The activity diagrams [4] are a graphical representation of the control and data flows between activities of multimedia objects [4, 1]. In the context of activity diagrams individual synchronization means the synchronization of a single event, which refers to some object's activity. At the same time, group synchronization means the synchronization of a set of timed events, which refers to many objects' activities (Fig. 13). A timed event that occurs on a flow determines the moment of an activity's execution or termination. On Fig. 13 the activity diagrams is shown extended by synchronized events notation.

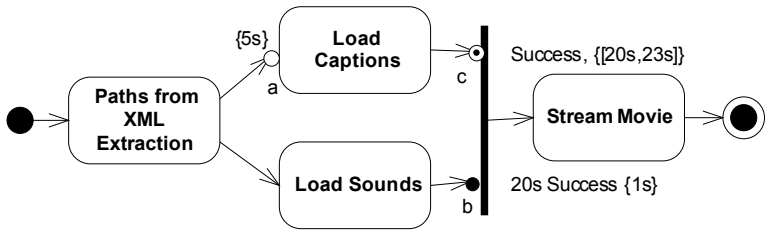

Fig. 13. An example of activity diagrams with marked synchronized events

The completion of the *Paths from XML Extraction* activity causes event *a* synchronizes individually in a 5 s interval. After that control is sent to the *Load Sounds* and *Load Captions* activities. The events $b, c \in Success$ synchronize absolutely in a 1 s interval in relation to the 20 s of activity sequence duration. Additionally event *c* synchronizes individually and absolutely in a 3 s interval.

5 Summary

The availability of precise and useful modeling methods are required in order to enable the production process of a multimedia application, which is reliant upon the ability to describe the necessary mechanisms of the synchronized event expressions. RT-UML allows for time characteristic modeling of time events, which occur during multimedia objects/activities' life, but without synchronization aspects. The ideas expressed in this paper, regarding the extension of RT-UML to include aspects of synchronization illustrates the originality of the concept. Taking advantage of the language's syntax, as well as its application to the original modeling method of multimedia applications, stands to formalize the production process of such applications.

References

[1] Marcinkowski B., Wrycza S., Wyrzykowski K., *UML 2.0 in information systems modeling,* Helion, 2006 (in Polish).

[2] OMG, *UML™ Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanism*s, OMG Press, 2003.

[3] OMG, *UMLTM Profile for Schedulability,* Performance, and Time Specification,Ver.1.1, OMG Press, 2005.

[4] OMG, *UMLTM Superstructure Specification,* OMG Press, 2005.

[5] Stasiak A., Wolski M., *Modeling of behavioral of information systems in UML 2.0,* SCR 2005 Proceedings, WKŁ, 2005 (in Polish).