Designing and Building Multimedia Cultural Stories Using Concepts of Film Theories and Logic Programming

Francesco Mele, Antonio Sorgente, Giuseppe Vettigli

Institute of Cybernetics, National Research Council
Via Campi Flegrei, 34 Pozzuoli(Naples) Italy
{f.mele, a.sorgente, g.vettigli}@cib.na.cnr.it

Abstract

In this paper we propose a middleware to reuse multimedia resources in order to produce new types of multimedia artifacts. In this work we adopt some basic concepts of film theory, such as the notions of plot, fabula and, in particular, diegetic time. The techniques we use are located within the area of artificial intelligence, using an explicit representation of time. The middleware consists of several modules, some devoted to the semantic annotation of multimedia components, and others to their visualization. Some modules regard the analysis of temporal connectivity and consistency of events. From a methodological point of view, an important module of the middleware contains the representation of a story (time of the narration and time of the story) and the temporal reasoning services, which are both implemented using a logic programming language (Flora2). Finally, there is a module in the middleware that translates the logical representation (in Flora2 language) into SMIL language, which allows the use of the final composition by a standard player.

Introduction[1]

A story, regardless of its cultural context, appears to be a pleasant and efficacious way of transmitting knowledge. This is because it captures the attention of the interlocutor and maintains his interest until the end of the story. In this work, our interest lies in the domain of historical narrations.

In that context, a story not only generates attention, expectations and surprises, but also contains some relations that can be studied in order to find implications among the events that compose it. The application of reasoning to the implications of historical events improve one's, especially a student's, cognitive abilities.

In this work (Fig.1): we propose a new methodology to represent cultural stories, using existing multimedia resources in the forms of photo, video, and text; we present useful services to build stories, such as some automated tools to check the connectivity and consistency of events; we illustrate a methodology for a "Semantic Editing" of cultural video; and finally, we present a program that automatically generates the code in SMIL Language (SMIL 2005) for the visualization of the story.

Cultural stories on Internet, especially on Wikipedia, are some examples of stories which inspired us to define the formalism of the representation of events. In this context, the material available is very extensive and almost all the examples found on Wikipedia contain the HISTORY field.

In the research of computational linguistics, some formalisms for the study of tenses have been developed, particularly the TimeML language (Pustejovsky et al. 2003). Such formalisms have their roots in the works of Reichenbach (Reichenbach 1947) and Allen (Allen 1987) and are useful tools for the representation of relations between the events of a story.

The reuse of multimedia materials in order to construct stories is a new Internet trend where there are some interesting initiatives (see some recent services of Google). However, little attention has yet been given to issues concerning the formal aspects of story representation and their *well-formed* composition (the connection of events and analysis of inconsistencies). In this work these aspects have been developed, where we have proposed a logical formalism and methodology for annotation of the multimedia, using notions of film theories such as diegetic events, plot, fabula, flashbacks and flashforwards (Bordwell 1985).

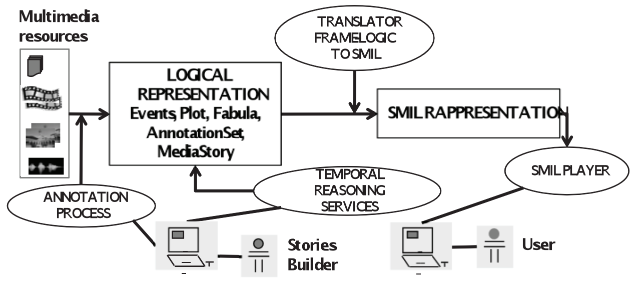

Multimedia resources
TRANSLATOR FRAMELOGIC TO SMIL
LOGICAL REPRESENTATION Events, Plot, Fabula, AnnotationSet, MediaStory
SMIL RAPPRESENTATION
SMIL PLAYER
TEMPORAL REASONING SERVICES
ANNOTATION PROCESS
Stories Builder
User

Fig. 1: The middleware components.

Representation of story events

We will show the basic elements of our representation of the cultural stories. We used the Flora2[2] (Yang et al. 2007) formalism to represent the key concepts of our methodology.

Temporal Point and Temporal Interval

The representation of the time we adopted in this work is mixed, it is based on points and time intervals. Time points are represented by explicit time labels in the form of dates:

```
[03,mar,1953]:date.   [07,sept,2009]:date.
[2009]:date.          [mar,2007]:date.
```

Instead time intervals of the form:

```
I_Time[t1=>date, t2=>date].
```

Some examples of `I_Time` are:

```
int1:I_Time[ t1->[03,sett,1943],
             t2->[17,nov,1943]].
int2:I_Time[t1->[mar,1951], t2->[gen,1951]].
int3:I_Time[t1->[1943], t2->[1947]].
```

(In this work we denote the variables of diegetic time always with the notion tdx).

Events of stories

For events of cultural stories we considered two types of entities: properties and actions that happen in temporal modality. Examples of properties that relate to cultural entities are: the colour or status of a monument (destroyed, renovated, degraded church) and the style of a building (Gothic, Baroque, Renaissance). Examples of actions are bombing, seismic events, and the act of rebuilding or renovating a monument. In this work all rules hold for properties and actions.

Our representation of a diegetic event of a story is as follows:

```
Event[tM=>TemporalMod, pA=>Property_Action].
Property_Action[PA=>Action].
    Property::Property_Action.
    Action::Property_Action.
```

`TemporalMod` is the temporal modality in which an action happens or a property is true. We distinguish two types of temporal modality: `I_TemporalMod`, e `P_TemporalMod`:

```
I_TemporalMod[int=> I_Time].
P_TemporalMod[td=>date].
```

The first class represents all modalities of occurrences of an action or property compared to time intervals, while the second class is compared to time points.

The subclasses of `I_TemporalMod` are: `EqualI`, `DuringI`, `AfterI`, `BeforeI`, `OverlapsI`, `MeetsI`, `StartsI`, `FinishesI` - while the subclasses of `P_TemporalMod` are: `AT`, `DuringP`, `AfterP`, `BeforeP`, `MeetsP`, `StartsP`, `FinishesP`.

The basic relations of our representations are similar to temporal relations proposed in the formalism TimeML (Pustejovsky et al. 2003).

We provide an example of formalism:
'Between 1943 and 1953 the church of Santa Chiara was restructured'

```
e1:Event[tM->tM1, pA->pA1].
tM1:DuringI[int->int1].
int1:I_Time[t1->[1943], t2->[1953]].
pA4:Property_Action[
    PA->ristrutturare(chiesaSantaChiara)].
```

For each type of event (During, After, etc.), we defined a Prolog clause abbreviation of type:

```
ev(?E,DuringI(?td1,?td2]),?az):-
    ?mtd1:DuringI[dt->?int],
    ?int:I_Time[t1->?td1,t2->?td2],
    ?E:Event[tM->?mtd1,pA->?pa1],
    ?pa1:Property_Action[PA->?az].
```

The representation of diegetic events is used for annotating the multimedia elements of the stories, for this reason they are the primitives of our formalism. We provide an informal description for them:
(actions that happen in relation to time intervals)

`ev(?E,EqualI([?td1,?td2]),?az).`
az happens from td1 to td2

`ev(?E,DuringI([?td1,?td2]),?az).`
az happens between td2 and td1

`ev(?E,AfterI([?td1,?td2]),?az).`
az happens after [td1,td2]

`ev(?E,Overlaps([?td1,?td2]),?az).`
az happens in part on [td1,td2]

`ev(?E,MeetsI([?td1,?td2]),?az).`
az ends before of [td1,td2]

`ev(?E,StartsI([?td1,?td2]),?az).`
az starts from the start of [td1,td2]

`ev(?E,FinishesI(?[td1,?td2]),?az).`
az ends to the end of [td1,td2]

(actions that happen in relation to time points)

`ev(?E,AT([?td]),?az).`
az happens at time (day) td

`ev(?E,AfterP([?td]),?az).`
az happens after td

`ev(?E,MeetsP([?td]),?az).`
az ends before of [td]

`ev(?E,StartsP([?td]),?az).`
az starts from the start of [td]

`ev(?E,FinishesP([?td]),?az).`
az ends to the end of [td]

[2]To make reading simpler, here we informally report some key constructs of this language: X::Y (class X is a subclass of Y), X:Y (X is an instance of Y), X =>Y (X is an attribute of type Y), X->Y (Y is the value of the attribute X), X *=>Y (same as X =>Y but it is also hereditable from its subclasses). The literals preceded by the symbol ? are the variables.

The annotation process

The multimedia objects are the basic components of our reuse methodology. In this work, we are not interested in a complete representation of these entities. For the semantic annotation process, we provide the following definitions:

```
VideoElement::MediaElement[ begin=>time,
                            end=>time].
AudioElement::MediaElement[ begin=>time,
                            end=>time].
PhotoElement::MediaElement.
```

(MediaElement is partially defined with respect to MPG7. Definition is not reported.).
The annotation process consists of associating a MediaElement with a diegetic event of the story. Formally, this relationship is:

```
MediaStoryElement[ me=>MediaElement,
                   e=>Event].
```

An example of an instance of a MediaStoryElement (an example of annotation) is as follows:

```
mse:MediaStoryElement[me->m1,e->eB].
eB:Event[tM->tM2, pA->Px]].
tM2:DuringI[int->int2].
int2:I_Time[ t1->[4,dec,1942],
             t2->[8,aug,1943].
Px:Property_Action[pA->bombing(Naples)].
```

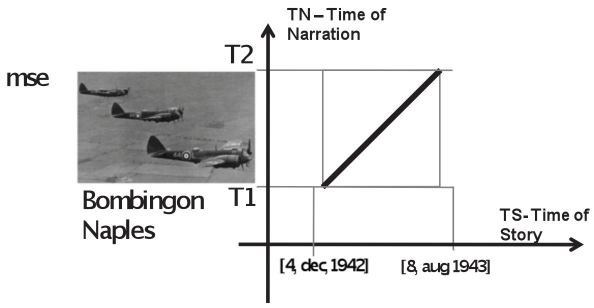


Fig. 2: Multimedial story element (mse) such as scene

We wish to emphasize that this instance corresponds with the notion of a movie scene, due the fact that the diegetic event is bound by temporal indexes, i.e. an instance of MediaStoryElement is a unit (the smallest) having complete sense (for TN-TS diagrams see the next sections). For a given multimedia story, we called the instances of all mse annotations: *Annotation Set*.

Plot and Fabula representation

The definition of fabula that has inspired our formalism is those given in (Eco 1979):

"the fabula is the fundamental structure of the narration, the logic of the actions of the syntax of characters, the temporally ordered course of events. It may also not be a sequence of human actions and may concern a series of events regarding inanimate objects, or also ideas"

```
Fabula[e=>Event, eRel=>EventRelations].
p(e1,e2):EventRelations.
o(e3,e4):EventRelations.
```

In our formalism the fabula is represented by a set of events and by a set of temporal relations among events. In this paper we use p(e1,e2) and o(e1,e2) to denote the qualitative temporal relations "e1 precedes e2", and "e1 overlaps e2".

Of course, the fabula involves more aspects of the representation provided herein. Regarding the objectives we have set ourselves in this work, we formalized the fabula especially under the aspect of time: we represented the basic logic of the actions and the temporally ordered course of diegetic events.

The definition of plot in (Eco 1979) is following:

"story as it is actually told, as it appears in surface, with its temporal dislocations, forward and backward jumps (i.e. anticipations and flash-backs) descriptions, digressions, ..".

```
Plot[int=>Interval, m=>MediaStoryElement].
Interval[ti=>time, tf=>time].
[00,04,53]:time. [00,08,58]:time.
```

The plot is defined as an ordered set of intervals, corresponding to which are associated the semantically annotated multimedia objects.

Multimedia Story and TN –TS diagrams

We provide a definition of Multimedia Story:

```
MultimediaStory[i=>Plot, f=>Fabula].
```

This definition captures the possibility of constructing different stories using the same (diegetic) events and changing the order in which they are presented. We consider the following fabula:

```
fabx:Fabula[ e->{eventx,eventy,eventz},
             eRel->{p(ex,ey),p(ey,ez)}].
ey:Event[tM->tM1, pA->pA1].
tM1:EqualI[int->int1].
int1:I_Time[t1->td1, t2->td2].
pA1: Property_Action.
ex:Event[tM->tM2, pA->pA2].
tM1:DuringI[int->int2].
int2:I_Time[t1->td3, t2->td4].
pA2:Property_Action.
ez:Event[tM->tM2, pA->pA3].
tM1:AfterP[int->int2].
int3:I_Time[td->td4]. pA3:Property_Action.
```

For this fabula, it is possible to define two different stories.

```
story1:MultimediaStory[ i->{p1, p2, p3},
                        f->fabx].
story2:MultimediaStory[ i->{p4, p5, p3},
                        f->fabx].
```

Plot of the story1

```
p1:Plot[int->I1,m->mse1].
I1:Interval[ti->t0,tf->t1].
p2:Plot[int->I2,m->mse2].
I2:Interval[ti->t1,tf->t2].
p3:Plot[int->[t2,t3],m->mse3].
I3:Interval[ti->t2,tf->t3].
```

Plot of the story2

```
p4:Plot[int->I4,m->mse2].
I4: Interval[ti->t0,tf->t11].
p5:Plot[int->I5,m->mse3].
I5:Interval[ti->t11,tf->t12].
p6:Plot[int->[t12,t3],m->mse1].
I6:Interval[ti->t12,tf->t3].
```

To represent relationships between elements of the plot and those of the fabula, we used the diagrams TN-TS introduced in (Mele et al. 2007).

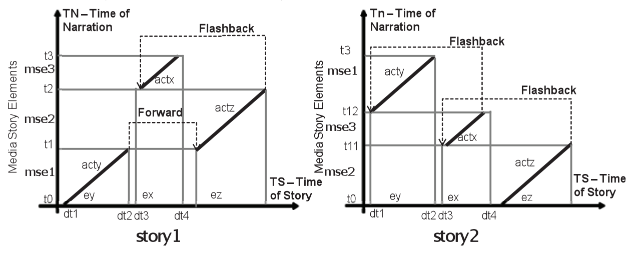


Fig. 3: Examples of TN-TS diagram.

In Fig. 3 the events represented in the previous example are shown. Diagrams TN-TS are part of the middleware services implemented. For the reading of these diagrams, we note that the axis of the stories is not to scale, whereas the axis of narration is.

Semantic Editing and automatic plot generation

Media representation and its semantic have suggested a new methodological approach for the construction of a story's plot, which we have called *Semantic Editing*. The scheme shown in Fig. 4 describes the stages of this approach.

Qualitative Sequential Editing

Film editing is the phase in which the available material is viewed, analyzed and reconstructed on the basis of narrative purposes. We present the approach, called *Qualitative Semantic Editing* (QSeEd), where the designer assembles the sequence of scenes (sequence of annotated media) by temporal qualitative relations among scenes. The language for these descriptions consists of a relation `overlapsdt(mse1,mse2,dT)`, and an index `timeRateMse`.

The `overlapsdt(mse1,mse2,dT)` relation indicates that the media `mse2` begins after a `dT` time from `mse1` (`dT` must be between zero and duration video mse1).

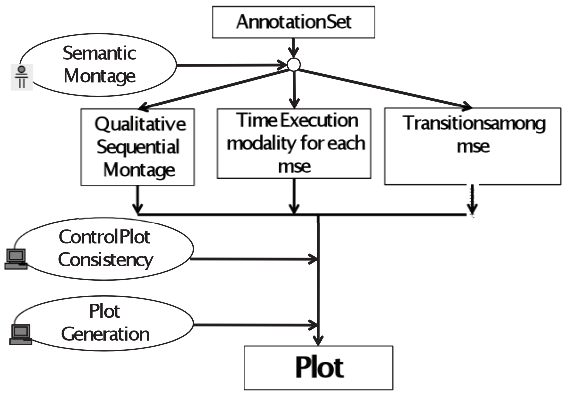


Fig. 4: Services defined for the Semantic Editing

Time Execution modality

Each `MediaStoryElement` has an intrinsic duration. The intrinsic duration of a video or audio is determined by the indexes defined in the `MediaElement` definition and indicates the actual duration of a video or an audio. The photos have an intrinsic time equal to 1.

In the semantic editing, for each `mse`, is useful to establish a ratio (`TimeRate=DEmse/DImse`) between the actual playing time (`DEmse`) and the intrinsic duration of the media (`DImse`).

The ratio is represented by the following predicate: `timeRateMse(mse1,TimeRate)`. A video with `TimeRate` equal to 1 indicates that the video is executed with the same intrinsic duration of the annotated media. `TimeRate` greater than 1 indicates *slowmotion*.

Transitions among multimedia story elements

In the Semantic Editing the user can select, in a symbolic manner, the transactions between two scenes (the effect created when there is a change from one scene to another). The effects are represented by binary predicates:

```
fade(mse7,mseN). cross_fading(mseN,mseM).
```

The name of the predicate describes the type of effect. These elements have a natural correspondence with the transaction in Smil.

Control Plot Consistency

For consistency checks of Fabula and Plot, we used the axiomatic of Russell-Kamp (Lambalgen and Hamm 2004). Using the stable model semantics (Gelfond and Lifschitz 1988), we implemented a new version of such axiomatic (see Table 1).

Tab. 1: An implementaon of Russell-Kamp Theory in ASP

```
p(X,Y) :-  pD(X,Y), not pN(X,Y).
pN(X,Y):-  event(X),event(Y),p(Y,X).
p(X,Z):-   p(X,Y), p(Y,Z), not pN(X,Z),
           event(X),event(Y),event(Z).
pN(X,Y):-  event(X),event(Y),o(X,Y).
o(X,Y) :-  oD(X,Y), not oN(X,Y).
o(X,X) :-  event(X).
o(Y,X) :- o(X,Y),event(X),event(Y).
oN(X,Y):-  event(X),event(Y),p(X,Y).
oN(X,Y):- event(X),event(Y),p(Y,X).
p(X,V):-   p(X,Y),o(Y,Z), p(Z,V),
           not pN(X,V),event(X),
           event(Y),event(V),event(Z).
```

This particular implementation was necessary as the original Russell-Kamp axioms contain rules with negation as failure. `p(x,y)` and `o(x,y)` are the temporal qualitative relations that precede and overlap those previously introduced.

Plot Generation

During semantic editing, a story builder selects the `MediaStoryElement` and defines the order of narration. The order of narration is specified by the `meets` and `overlaps` relations, while the duration of each media segment is calculated by index `timeRate`. We implemented the algorithm QSeMo2Plot for the creation of the plot. QSeMo2Plot receives the inputs Qualitative Sequential Editing and `timeRate` and returns the media story element ordered in time.

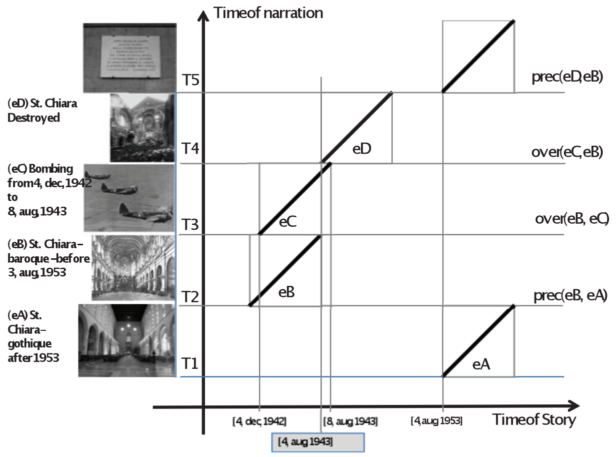
The inputs of the program are the set of relations `oD(x,y)` and `pD(x,y)` of which the program controls the consistency.

SMIL Code Generation

In this work we used SMIL Language (SMIL 2005), a language designed for the integration and synchronization of multimedia sources. We implemented the algorithm Plot2Smil to translate the Plot defined logical representation into SMIL language. The algorithm uses the translation rules described in Tab. 3 and creates a SMIL file that can be run by a standard video player.

The rule T1 defines the translation from an instance of Plot to an element of SMIL. Rules T2 and T3, instead, describe how to synchronize the media according to their *Qualitative Sequential Editing* relations. The synchronization is done using the SMIL's tag `<par>`, which allows parallel executions of multiple media.

`meets(mse1,mse2)` relations have a natural correspondence, in SMIL, with the tag `<seq>`, but in order to simplify the algorithm, we defined the translation rule (T3) with the tag `<par>` and the constraint "for any `mse1` and `mse2`,`mse2` starts after the end of `mse1`".



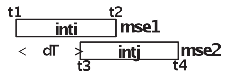
Timeof narration
T5
T4
T3
T2
T1
(eD) St. Chiara Destroyed
(eC) Bombing from4, dec,1942 to 8, aug,1943
(eB) St. Chiara– baroque –before 3, aug,1953
(eA) St. Chiara– gothique after1953
eD
eC
eB
eA
prec(eD,eB)
over(eC,eB)
over(eB, eC)
prec(eB, eA)
[4, dec, 1942]
[8, aug 1943]
[4,aug 1953]
[4, aug 1943]
Timeof Story

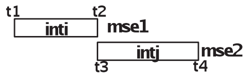
Fig. 5: TN-TS diagram of S.Chiara church story.

Tab.3: Translation rules from Plot to SMIL

	Plot	Smil

T1	`mse1:MediaStoryElement`
	`.`
	`mse1[me=>me, e->e1].`
	`me:videoElement[`
	`media->md,`
	`begin->10.5s,`
	`end->25.7s].`
	`p1:Plot[int->i[ti->t1,`
	`tf-`
	`>t2],`
	`m->mse1]`
		`<video id=mse1`
		`src=url1`
		`...`
		`clipBegin="10.5s"`
		`clipEnd="25.7s"`
		`dur=DEmse(mse1)/>`

T2	`overlaps(`
	`mse1,`
	`mse2,`
	`dT)`
	`Con`
	`0 ≤ dT <DEmse(mse1)`
	 t1
	t2
	inti
	mse1
	< dT >
	intj
	mse2
	t3
	t4
		`<par>`
		`<video id=mse1`
		`src=url1`
		`clipBegin="10.5s"`
		`clipEnd="25.7s"`
		`dur=DEmse(mse1)/>`
		`<video id=mse2`
		`src=url2`
		`clipBegin="13.5s"`
		`clipEnd="42.7s"`
		`dur=DEmse(mse2)`
		`begin=mse1.begin+dT`
		`<par/>`

T3	`meets(`
	`mse1,`
	`mse2)`
	
		`<par>`
		`<video id=mse1`
		`src=url1`
		`clipBegin="10.5s"`
		`clipEnd="25.7s"`
		`dur=DEmse(mse1)/>`
		`<video id=mse2`
		`src=url2`
		`clipBegin="13.5s"`
		`clipEnd="42.7s"`
		`dur=DEmse(mse2)`
		`begin=mse1.end/>`
		`<par/>`

Partial Order Construction, Connectivity Evaluation and Consistency Control

In this chapter we will show some programs written in Flora2/Prolog which constitute a formalization of key concepts of the stories and, at the same time, the basic functions that can be used when the story builder defines a story reusing multimedia resources.

Partial Order Construction

The annotated events of the stories represent how the actions happen with respect to a given temporal modality, for example:

```
  ev(e1,DuringI(Date,Date),Az1).
  ev(e2, BeforeP(Date), Az2).
```

These relations do not represent the temporal order of events in a story. We implemented a program that compares each event of a story with all the other events of the *Annotation Set*. Each comparison attempts to discover whether an event *precedes* or *overlaps* another event. The program consists of about 70 Prolog-Flora2 rules, and we list some of them as examples:

```
findRel(?e1,?e2):-
      ev(?e1,DuringI(?Date1,?Date2),_),
      ev(?e2,AfterI(?Date3,?Date4),_),
      ?Date4 <=D ?Date1,
      inserta{p(?e1,?e2):EventRelations}.
findRel(?e1,?e2):-
      ev(?e1,EqualI(?Date1,?Date2),_),
      ev(?e2,EqualI(?Date3,?Date4),_),
      ?Date1 <=D ?Date3, ?Date3 <=D ?Date2,
      inserta{o(?e1,?e2):EventRelations}.
```

(The predicate `?D1 <= D? D2` - infix form - returns True if the date `?D1` is less than or equal to date `?D2`).

p(?e1,?e2) and (?e1,?e2) are the relations of precedence and overlap previously introduced. They define the partial order of events of the story.

Connectivity Evaluation

The partial temporal order of events does not ensure that a story is connected. This may be because the story builder does not insert the necessary annotations.

We implemented a service that concerns the connection of the story:

```
    connectedSetEx(?Ex, ?RelEv,?EvCList).
```

This program for a given event (`?Ex`) and a set of temporal relations (`?RelEve`) among events, returns the list (`?EvCList`) of all events connected to `?Ex`. The algorithm of `connectedSetEx` (not shown here) was implemented, representing each event in a story as a node of a graph, and relations between events as labeled edges of the graph.

The algorithm was implemented in XSB Prolog (Swift, Warren and Sagonas 2009) and implements a variant of a standard search algorithm in a graph (see Shoham 1994).

`ConnectedSetEx/3` also lets us know if a fabula `F` is connected, checking whether the list of events connected (`?EvCList`) to a given event (`?E`) is equal to the list of all events of the fabula `F`.

If the fabula is not connected, there are other services that can help a story builder to connect the events. These programs suggest some relations in order to connect the fabula.

Consistency Event Control

Cultural stories, for obvious reasons, must be consistent; for example, p(e1,e2) and p(e2,e1) cannot happen. To implement this program we used the algorithm presented in previous chapters (Control Plot Consistency).

Conclusions

We formalized some key concepts for the representation of stories that we believe have a general validity, not only for cultural stories. An approach based on the Event Calculus (Miller and Shanahan 1999), as suggested in (Mueller 2003), is not adapted to this end (the language is not expressive enough to represent cultural events, and not suitable to check the connectivity and inconsistencies). However, the rigorous approach of the theories of Event Calculus may be useful if we want to build a reasoning program on the connectivity of events. In this case, however, it is necessary to implement a complex software architecture that embeds many elements related to the global aspects of events, such as connectivity and inconsistency.

References

Allen, J.F. 1987. *Natural Language Understanding*. Menlo Park, Ca, Benjamin/Cummings.

Bordwell, D. ed. 1985. *Narration and the Fiction Film*. London, Routledge.

Eco, U. 1979. *Lector in fabula*. Milano, Bompiani.

Gelfond, M.; Lifschitz, V. 1988. The Stable Model Semantics For Logic Programming . *In Proceedings of the Fifth International Conference on Logic Programming*, 070-1080. Seattle USA.

Lambalgen, M.V.; and Hamm, F. 2004. *The Proper Treatment of Events*. Blackwell, Oxford and Boston, 2004.

Mele F.; Calabrese A.; Marseglia R. 2007. Interactive Analysis of Time in Film Stories. *In Proceedings of the tenth AI*IA*. 756-772. Rome, Italy:LNCS Springer.

Miller, R.; and Shanahan, M. 1999. The event-calculus in classical logic - alternative axiomatizations. In Electronic Transactions on Artificial Intelligence 3(1): 77-105.

Mueller, E.T. 2003. Story understanding through multi-representation model construction. *In Proceedings of the HLT-NAACL 2003 Workshop*. 46-53. East Stroudsburg, PA: Association for Computational Linguistics.

Pustejovsky, J.; Castaño, J.; Ingria, R.; Saurí, R.; Gaizauskas, R.; Setzer, A; and Katz, G. 2003. TimeML: A Specification Language for Temporal and Event Expressions. Netherlands, Kluwer Academic Publishers.

Reichenbach, H. 1947 *Elements of Symbolic Logic*. London, Mc Millan.

Shoham, Y. 1994. *Artificial Intelligence Techniques in Prolog*. Morgan Kaufmann Publishers.

Smil. 2005. Synchronized Multimedia Integration Language (SMIL 2.1) http://www.w3.org/TR/2005/REC-SMIL2-20051213/).

Swift, T.; Warren, D.S.; and Sagonas, K. 2009. The XSB system version 3.2 - volume 1 e 2 http://xsb.sourceforge.net.

Yang, G.; Kifer, M.; Wan, H.; and Zhao, C.. 2008. FLORA-2: An Object-Oriented Knowledge Base Language, http://flora.sourceforge.net/.