INTERNATIONAL ORGANIZATION FOR STANDARDIZATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND ASSOCIATED AUDIO

ISO/IEC JTC1/SC29/WG11
MPEG00/M6355
July 2000, Beijing, China

Title: **Report of CE on Semantic DS**
Status: **Contribution**

Authors: **Ana Benitez, Shih-Fu Chang (Columbia University)**
Rajiv Mehrotra (Eastman Kodak Co)
Koiti Hasida (Electrotechnical Lab)
Hawley Rising (Sony)
Corinne Joergensen (Univ. at Buffalo, State Univ. of New York)
Riccardo Leonardi, Alessandro Bugatti (University of Brescia)
Ed Hartley (University of Lancaster)
Murat Tekalp (University of Rochester)

1 Introduction

This experiment has evaluated the Semantic DS. The CE was started at the Maui meeting in December 1999 [15]. A progress report of the CE was provided at the Geneva meeting [1]. The result of the discussions in Geneva was to continue the CE by [17]:

- Further refining the specification of the Semantic DS.
- Assessing the descriptive power of the Semantic DS by establishing a correspondence between the Semantic DS and frameworks of known descriptive power, by instantiating example descriptions, and by trying to find descriptions that can not be expressed using the Semantic DS.
- Assessing the usability of the Semantic DS by proving that a proper set of types of combinations of descriptions are possible, by trying to find descriptions that can be expressed in structurally different ways preventing any matching between them, and by comparing the retrieval efficiency with text search engines.

This document reports on the results of this work.

2 Work Plan

The experiment for validating the *Semantic DS* re-started after the 52nd MPEG meeting in Geneva May 31-June 2, 2000 and has make a report of the results at the following meeting in Beijing, in July 2000.

2.1 Multiple Parties

The participants of this Core Experiment are:

Name	Company
Ana Benitez	Columbia University, USA (P1)
Rajiv Mehrotra	Eastman Kodak Co, USA (P2)
Koiti Hasida	Electrotechnical Lab, Japan (P3)
Hawley Rising	Sony, USA (P4)
Corinne Joergensen	Univ. at Buffalo, State Univ. of New York, USA (P5)
Riccardo Leonardi	Univ. of Brescia, Italy (P6)
Ed Hartley	Univ. of Lancaster, UK (P7)
Murat Tekalp	Univ. of Rochester, USA (P8)

2.2 Context

The context of the experiment shall use audio-visual programs or image catalogs, with a scenario where they get described for later querying and retrieval by expert and non-expert humans.

2.3 Test Material

The experiment shall use summaries of news programs, or newspaper articles, extracts of the following portions of the MPEG-7 content set and additional material (shared among all participants):

1) Sporting event video, e.g., soccer game (CD 14, CD 20, CD 28)
2) News videos (CD 14, CD 15)
3) Documentaries (CD 28)
4) Still image collection (provided by Eastman Kodak or Sony, U.S.A., and in default to be taken from MPEG-7 content set)
5) Audio material to be agreed upon.

Participants will generate or use preexisting (natural language) description of portions of such content and share them, at latest, by June 25, 2000 throughout the CE process.

2.4 Objectives

The output of the Semantic DS CE shall:

1. Update UML specifications, verify the correctness of the DDL syntax provided in sections 3 to 6, and add or revise some missing components (Term D, ConceptGraph DS, ConceptLink DS, Weight DS).
2. Assess the flexibility, expressiveness and completeness of MPEG-7 semantic DSs to translate (natural language) descriptions.
3. Assess the effectiveness of the Semantic DS for retrieval in comparison with current text search engine.
4. Try to assess the data entry difficulty to translate descriptions using the Semantic DS syntax.
5. Recommend changes to the proposed syntax if necessary to enable better description capabilities.

2.5 Demo

A demo should be provided indicating for each used description, the various instantiations of the Semantic DS. It should also demonstrate the type of responses to the proposed queries, supplied by the Semantic DS compliant descriptions.

2.6 Methodology

The CE will be conducted in order to estimate on one hand the descriptive power of the DS, on the other hand its usability for retrieval, and for easy creation of Semantic DS compliant descriptions. The validation will be performed using both empirical and formal assessment.

2.6.1 Preliminary Work

1. Verify the DDL for the Semantic DS components.

2. Share the portions of the audio-visual material not belonging to the MPEG-7 content set used in the experiment.
3. Share descriptions of selected content among CE participants, from simple to complex and metaphorical descriptions (e.g., titles, keywords, simple short natural language ones, long complex ones, etc.), exhibiting both denotative and cognitive aspects of the content.

2.6.2 Assessment of Descriptive Power

1. Formal validation (P3/P4)

   Establish the correspondence between the Semantic DS and frameworks of known descriptive power for a quantitative estimate of its descriptive power.

2. Empirical validation (all participants P1-P8)

   Instantiate the above description examples using the Semantic DS definition. Identify description examples which cannot be expressed by the MPEG-7 Semantic DSs (apart from using the TextAnnotationD). P4 (Sony, USA) will keep a record of all descriptions provided, and distribute them to the other participants when requested.

2.6.3 Assessment of Usability

1. Formal validation (P3/P4)

   Prove that proper set of types of combinations of descriptions are possible

2. Empirical validation (all participants P1-P8)

   a. Estimate data entry difficulty.

   b. Retrieval efficiency:

      i. Try to instantiate the same description example in two structurally different ways using the Semantic DS syntax preventing any matching to be established between the two descriptions.

      ii. Compare retrieval efficiency, i.e. number of correct responses (precision/recall) to queries, in comparison with the performance of state-of.-the-art search engines.

2.7 Time Table

The Semantic DS core experiment shall be conducted according to the following timetable:

June 15, 2000	**June 25, 2000**	**Jul. 10, 2000**
Verify DDL syntax and update pseudo UML diagrams		
Provide description examples in Semantic DS DDL syntax allowing to identifying counterexamples (i.e. descriptions that cannot be represented with the proposed syntax)		Comparison of retrieval performance Assessment of ease in instantiation of a semantic description.

Formal study of descriptive power and usability of Semantic DS

Table 1. Work plan timetable for Semantic DS core experiment.

2.8 Monitoring

The core experiment work shall be monitored by the Ad Hoc Group on MPEG-7 DS Validation and Core Experiment Planning [4].

3 Experiment Results

3.1 Specification of Semantic DS

Due to the size of the specification of the Semantic DS, it has been included in Annex A. Annex A contains the syntax, semantics, pseudo-UML representations, and examples of the different components of the Semantic DS. More examples of descriptions instantiating the Semantic DS generated by this CE are included in Annex B.

The specification of the Semantic DS has been modified since the Geneva meeting. However, a thorough review of the syntax and semantics of the Semantic DS for the past few months has identified several issues, which are highlight through out Annex A and summarized below.

- Harmonization with other components in MPEG-7:
  - The specification of some components of the Semantic DS is pending the resolution of several core experiments; examples of these components are the Weight DS, the SemanticGraph, the ConceptGraph, the SemanticLink, the SegmentSemanticLink, TextAnnotation, and the RegionLocator.
  - The specification of some components of the Semantic DS needs to be harmonized with the new version of the MDS XM/WD; examples of these components are the MediaLocator, the Stime, and the Slocation. The specification of Stime and Slocation should be synchronized with the previous work of the MedMet AHG to describe semantic time and places (e.g. Place DS). Stime and Slocation could also adopt the method to express equivalence among Segments.
  - The specification of some components of the Semantic DS are pending the resolution of open issues in the MPEG-7 DDL. For example, Object DS and Event DS could be derived from ConceptObject DS and ConceptEvent DS by adding the links to media; as SemanticDescription from Concept. These modifications would require restricting the type of child elements in a XML-Schema derivation by extension. This is currently not supported in XML-Schema; however, it was an open issue identified by the MPEG-7 DDL AHG. For example, when Object DS is derived from ConceptObject DS, then Object DS can only contain other Object DSs (no ConceptObject DSs).
- Points of discourse between participants:
  - The inclusion of Stime DS and Slocation DS in Object DS should be further investigated. Object DS has tools to locate the time and position of the object in the media (through MediaLocator); however, it is not allowed to have tools to describe its time and location

in the real world. This is clearly an inconsistency of the current specification of the Object DS.

Describing the time and location of objects in the real world is possible with the current Semantic DS by using Event DS and SemanticLink DS (see section on SemanticDescription DS in Annex A); however, these descriptions could be much more compact if Stime and Slocation are included in Object.

Arguments that oppose the inclusion of Stime and Slocation in Object are based on clarifying the semantics of Object and Event as an Object is not inherently specified by a time or a location.

- Requiring further work:
  - Some components of the Semantic DS need further specification and clarification; examples of these components are the ConceptLink, RegionLocator, ImageLocator, and State.
  - There are three semantic relationship graph elements: Semantic Graph DS, SemanticLink DS, and SegmentSemanticLink DS. However, it is still not clear how they differ: what relationships (and types of relationships) they describe, what is their scope, etc. There is a need to identify those elements that need to be related in the Semantic DS and what mechanism/s can be used for that purpose.
  - Concepts seem to be more than just template for objects and events. They could also be abstractions for semantic descriptions. Concept appears in two different places in the Semantic DS: within the Semantic DS and the SemanticDescription. This should be justified.
  - Investigate the definition of MediaModel as an Analytical Model DS. The MediaModel DS seems to contain two different types of current Model DSs: the Example DS explicitly and the Cluster DS implicitly (0..* Media Locators). Furthermore, MediaModel DS could also use Probability Model DS (another Model DS in MDS WD too).

In spite of the remaining issues in the specification of the Semantic DS, the participants in the CE believe that the current Semantic DS contains the basic tools needed to describe semantic information and that just refinements of their specification are needed.

3.2 Selection of Content

The content used in this experiment was selected from the one described in section 2.3. In addition, photographs and associated textural annotations were drawn from the Melbourne Photo Database collected for testing the speech-based recognition description schemes in the audio group (see http://www.cre.canon.co.uk/mpeg7/melbourne_photo_database.htm).

3.3 Generation of Descriptions

The generation of descriptions instantiating the Semantic DS was done following two different methods. The first method was to take natural-language descriptions and to encode them using the Semantic DS. The second method was to create descriptions of multimedia material from scratch using the Semantic DS. Examples of descriptions generated by both methods are included in Annex B.

3.4 Application of Descriptions

The descriptions generated for the multimedia material were assumed in a retrieval application for evaluation. No working system was implemented that used the descriptions; however, retrieval considerations were used in the evaluation of the Semantic DS (see section 3.5).

3.5 Evaluation and Recommendations

This section contains the evaluation of the Semantic DS in terms of its descriptive power and usability as described in the experiment work plan. A formal and empirical evaluation of the descriptive power and usability was carried out.

3.5.1 Formal Validation

The purpose of this section is to evaluate the parts of Semantic DS, and determine the expressive power, and usability of these parts, by formal means. Specifically, we analyze the parts of Semantic DS in terms of the ability to write arbitrary descriptions in a structured manner. Most of our attention will be focused on the larger DS, Object, Event, Concept, and the graphs, although we will discuss the others in turn.

The DS comprising Semantic DS are:

- STime: This DS deals with semantic time descriptions.
- SLocation: This DS deals with semantic place descriptions.
- MediaLocator: This DS connects the description to the media
- MediaOccurrence: This DS is a lightweight segment, as annotation is lightweight semantic.
- AnalyticModel: This DS allows the use of non-verbal material in construction of descriptions
- Object: This DS describes objects occurring in the described media
- Event: This DS describes events occurring in the described media
- SemanticDescription: This DS encapsulates a complete description.
- Concept: This DS is an abstraction tool that looks like SemanticDescription.
- SemanticGraph: This DS is a graph of the relations between the DS in SemanticDescriptions.
- Links, in particular SegmentSemanticLink.
- State: A bundle of attribute value pairs which allow the specification of parameter values at an instant of time or at a particular location.
- UsageDescription: A boolean indicating the purpose of a description, that is, whether it is intended as description or as an indexing element.

There are other DS, for instance, for each DS within SemanticDescription that has access to media, as well as for the graph, there are counterparts within Concept. We will analyze Concept as an entity, and deal with these as necessary. There is also a Semantic DS. This is used to hold one or several SemanticDescriptions or Concepts, or both. There is also the suggestion, made at the Geneva meeting, that the abstract descriptions, in the form of Concepts, be stored in Classification Schemes, as part of the description of controlled terms.

We will assume throughout this section that the changes made by the ER Graph DS, or compatible changes, have been accepted. Some of the mandates for that core experiment were dictated by needing certain graph functionality for this DS, so it is a reasonable assumption, since these were satisfied in that CE.

3.5.1.1 Simple DS

We will consider DS which do not participate in creating structures inherently, except as one end of a link, or as nodes in a graph, to be simple. This is a notational convenience, they do not affect our analysis of expressive power, except as maps from the expressive power of the structured elements, like Graph, to various description domains. For our purposes, a DS is simple if the following are true:

3. It contains no recursion. A recursive DS necessarily implies structure, in that it allows the construction of a tree of like elements.
4. It is not a graph. Graph DS and its derivatives, as the name implies delineate a graph structure.
5. It does not contain elements fitting 1, and 2. DS with recursive or graph elements are necessarily structured.

For simple DS then, we have STime, SLocation, MediaLocator, MediaOccurence, AnalyticModel, State, and UsageDescription. Using Graph, which according to the recommendations of the ER Graph CE should be made pervasive, structures may be put together from these, to create complex structured elements. For certain of these simple descriptors, this makes sense. Indeed, the reason for the construction of the STime and SLocation DS, is that it is anticipated that to create semantic descriptions which describe complex temporal or spatial situations, these will in fact be instantiated multiple times in a description, and related through links, or in the graph. A graph consisting entirely of STime can be used to construct a complex temporal description. That these occur in semantic descriptions of media can be seen in the following hypothetical example:

“In this scene, Harry cringes at the vivid memories of last Thanksgiving at his parents, when he flashed back at dinner, and thought he was back in ‘Nam.”

The description given has three time frames occurring within one AV sequence. The description

“In this scene, Harry runs into the girl he met in Chicago, while on the way to the zoo.”

likewise has three locations in its description.

On the other hand, the purpose of UsageDescription is to provide structural hints as to the organization of the whole description. It is highly unlikely that a use can be found for creating a complex structure of UsageDescriptions, other than to create a geometrical hierarchy – indexes of indexes.

3.5.1.2 Complex DS

By complex DS, we will mean those DS which have one or more of the properties cited above in the simple type definition, namely, recursion, or graphical properties. The DS with these properties are Semantic, SemanticDescription, Object, Event, Concept, and SemanticGraph. Of these, Semantic and SemanticDescription derive their structure by containing other complex DS only. This yields Object, Event, Concept and SemanticGraph.

3.5.1.2.1 Objects and Events

The syntax of Objects and Events are quite similar. Objects express a semantic object, an entity localized in space, for the purpose of referencing its occurrence in the media. Likewise, Event is used to express a temporally localized entity, for the same purpose. Both Object and Event recurse, but do not allow expression of the other major types within them. That is, there are no Events or Concepts within an Object hierarchy, and no Objects or Concepts within an Event hierarchy.

The purpose of the recursion in each case, and its default meaning is subdivision and refinement of description through subdivision. In this respect, these two DS are direct counterparts to the segments and regions within the Segment hierarchy. There is, however, no semantic counterpart to the spatiotemporal formulation available in the Segment trees. There is less formality in the structure than in the segment and region descriptions. There is no constraint on the size of a subobject, technically there is no constraint that it be a true subobject. Likewise, there is no semantic counterpart to expressing overlaps and gaps in Event.

Expressive power: Both these DS form trees, that is finite partially ordered sets with maximal elements. Since they may occur in large numbers within SemanticDescriptions, the sum total of the objects and events in a description form a forest of such trees. Because of the lack of constraints on these trees, we may add structure to either Object or Event, defining new objects or events from existing ones by union and intersection. This may be done by creating a sibling intersection of two (or more) objects at a particular level of the hierarchy. This object may then be linked to subobjects of each at the next level of refinement. Union is expressed by the parent node. In this way, either Object or Event may be construed as a lattice, and in effect, can be written to describe the lattice of subobjects or subevents of a particular object or event.

3.5.1.2.2 Semantic Graph

We next deal with the graph structures in the Semantic DS. A formal discussion of these structures is included in the report for the ER Graph CE, and will be referred to here, instead of repeating it. The text if this section uses terminology out of category theory and graph grammars, which are the proper tools for assessing the capabilities of this DS. It is not really possible to give an introduction to these subjects in a short space, suffice it to say, that the descriptive power of graph structures can be proven to be equal to other structures of known descriptive power, which is what is being described below. The Graph DS allows both the construction of graphs and the constructions of graph morphisms. It allows the referencing of subgraphs through these graph morphisms, and it allows the creation of discrete graphs. As such it allows the creation of product graphs through the creation of morphisms expressing the projection maps, it allows the creation of coproduct graphs through the creation of morphisms expressing injection maps. Through the injection maps, it allows the specification of subgraphs, and power sets of graphs. The discrete graph with a single node and edge is the terminal object in the category **Graph**, node lists in the Graph DS specifically allow the construction of this graph. Graph DS is therefore capable of replicating the category Graph (theoretically), and therefore is a topos as a functor category of the form $\mathbf{Set}^{\mathbf{C}}$ with **C** small. A more complete formulation is available in the ER Graph CE document.

Within the context of Semantic DS, this means that the expressive power of the DS is essentially limited only by the restriction of link and node types. We make the following assumptions:

1. Edges in Graph are allowed to be chosen from any controlled list, or allowed to be labeled with free text.
2. Textual descriptions of the nodes are likewise either controlled text, free text, or other media (in the case of AnalyticModel).

Then the Semantic DS are capable of producing any description from natural language, and, formally, any Turing computable string. This latter is a consequence of the fact (Lambek and Scott) that Cartesian closed categories are equivalent to lambda theories, and the category of graphs is isomorphic to the category of categories. Both of these categories are Cartesian closed, a consequence of the fact that Graph is a topos.

3.5.1.2.3 Concepts

Concept DS is intended for two purposes: to express information such as abstract objects or adjectival information for the purposes of a description, and to abstract SemanticDescriptions into templates. Such abstraction is useless unless it can then be instantiated, this requires building a set of correspondence links between descriptions (as embodied in the entities in SemanticDescription that access the described media), and the concepts they instantiate, or partially instantiate. Currently this ability is limited.

Recommendation: The documentation for Concept be amended to allow references to the interior structure of a Concept from all entities in a SemanticDescription.

Expressive power: As it currently is construed, Concept is a replication of the elements in SemanticDescription. As such each Concept is capable of generating any description abstractly. It is therefore a candidate for controlling the construction of descriptions, in essence a control for controlled terms. For this reason, and because of the template function for Concept, they should be archived in Classification Schemes, or other similar lists.

Recommendation: The description field in Classification Scheme be augmented to allow the description of a controlled term by a Concept. Reference to the term should allow optional retrieval of the Concept.

3.5.1.3 Support for Complex or Abstract Descriptions

Part of the reason for examining the expressive power of Semantic DS was to relate it to known descriptive abilities in natural language and semiotics. These were referred to in input document [18] in Noordwijkerhout. The constructs which support the construction of mental space objects, contexts and blends are the support for pullbacks, and the support for pushouts.

Contexts are supported through the ability to create substitutions. This may be efficiently done via rule based models, as in Jacquet. These constructions require pullbacks of graphs, with replacement of the appropriate nodes and their connecting structures by other nodes. This allows efficient replacement of one descriptive construct within another, for example in editing applications involving objects or regions. As described in Jacquet, rules and underlying alphabets are formed by graphs, which in Semantic DS would be ConceptGraphs, and the pullback is created which substitutes a construct for a node. This method takes care of the links that need to be rewired as a result, and the method requires that pullbacks can be constructed. These are taken care of by the graph morphism mechanism of Graph DS.

Blends are supported through the construction of pushouts. These can be thought of as correspondences which glue two descriptive graphs together. As described in [], these allow juxtaposition of two descriptions to form a third, and allow the construction of metaphorical structures, as described in Fauconnier. Pushouts are supported in Graph DS by constructing graph morphisms and partial graph morphisms.

3.5.1.4 Conclusions

We conclude that the Semantic DS is as expressive as it can be, modulo the restriction from expression possible by constraining relationships between entities in the Semantic DS. We have demonstrated (ER Graph DS CE) that the graph structures, and linking mechanisms allow the construction of items from the category Graph, which is a topos, isomorphic to the category of small categories. We have noted that the Semantic DS is capable of constructing contexts, and mental space blends, required to formulate descriptions by analogy or metaphor, and to establish compatible replacements by a methodology from graph grammar.

We end this section by repeating the main recommendations made:

Recommendation: The documentation for Concept be amended to allow references to the interior structure of a Concept from all entities in a SemanticDescription

Recommendation: The description field in Classification Scheme be augmented to allow the description of a controlled term by a Concept. Reference to the term should allow optional retrieval of the Concept.

Because the expressive power of the description schemes in Semantic DS is limited by the restraints on the relationship edges, we make one further recommendation:

Recommendation: The lists of acceptable relationships for SemanticGraph, ConceptGraph, and the Links between Semantic Entities should be construed as Classification Schemes, and created for application specific purposes. To the extent that MPEG-7 creates any of these lists, it should be from the point of view of establishing standard “profiles”, not constraining the standard as a whole.

3.5.1.5 Empirical Validation

3.5.1.5.1 Descriptions Breaking Semantic DS

Descriptions that could not be expressed using the Semantic DS were not found. Although some D/DSs in the Semantic DS require further work on their specification, some of its components, specially the Semantic Graph, add considerable descriptive power to represent semantic descriptions.

3.5.1.5.2 Retrieval Efficiency

This section evaluates the retrieval efficiency of the Semantic DS in to fronts: (1) investigating how the same description can be expressed by multiple structurally different instantiations of the Semantic DS and how this would affect retrieval, and (2) comparing retrieval of descriptions expressed using with Semantic DS to free text retrieval.

3.5.1.5.2.1 Different Ways to Instantiate Descriptions

Annex B includes description examples of the Semantic DS. Some of these examples include the same description encoded structurally different instantiations of the Semantic DS. The conclusions from deriving these descriptions are the following:

- The current Semantic DS is very flexible because it allows to encode the same description in many different ways using different components of the Semantic DS. For example, a city could be described as an object or as a SLocation. A city might play an important role in a movie and annotators might want to represent it as an object to describe interactions with other objects.
- This flexibility adds complexity but more power to the retrieval of semantic descriptions expressed with the Semantic DS. To achieve a good balance of complex vs powerful retrieval of semantic descriptions, the specification of the Semantic DS should be made clear and consistent.

3.5.1.5.2.2 Comparison with Free Text Retrieval

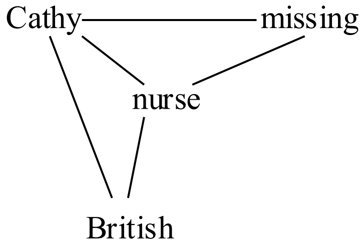
We have collected some examples of information retrieval (IR) tasks which are practically intractable for the traditional keyword-based search engines but tractable for interactive search engines which exploit semantic structures encoded by Linguistic DS or Semantic DS. The purpose here is to demonstrate that there are many enough such queries and thus Linguistic DS and Semantic DS are beneficial enough. In fact, many researchers have encountered the limitations of keyword search. We give some of their examples below:

Keyword search on free text generally cannot associate modifiers with an entity or action. One of the Melbourne photos is captioned "A long narrow boat on the river". Most search engines turn this caption (or query) into: "long, narrow, boat, river" which is indistinguishable from "A boat on a long narrow river" – a different concept entirely! So if a user wanted to find images of certain types of boats (e.g., long / narrow), it helps if we can specify that these words are modifying "boat" and not "river". Similarly, "A black swan and eels in the botanic gardens" is indistinguishable from "A swan and black eels in the botanic gardens" without some encoding of linguistic structure. (Example given by Tony Rose, Canon Research Centre Europe)

Keyword search on free text cannot detect relationships between search words, other than coincidental (the words appear in the same caption). When searching for an image of a woman riding a bicycle, a keyword search would return an image captioned "A woman reading a book on a bench. A boy riding a bicycle on the path behind the bench." (Example modified from one given by Erik Oltmans, Telematica)

Keyword search on free text cannot give more weight to matches on the entity or action vs. a modifier or to matches on an agent vs. an object. To search for banknotes in a heterogeneous image collection, we chose for QBE "English banknote" and got "English policeman beating up hooligan" as the third result, ranked much higher than other, non-English banknotes. Structured annotation enables *a priori* weighting: people are usually more interested in entities than in their modifiers. Someone looking for a "white cat" would probably prefer an image of a black cat over an image of a polar bear. (Example given by Wolfgang Müller, University of Geneva)

Here is an example. The query is `A foreign nurse disappears,' and a solution is found in http://www.webcom.com/~funhaus/pf/astd.html. This page begins with `Jane (Pamela Franklin) and Cathy (Michele Dotrice), two British nurses on a bicycle holiday through rural France, travel down the wrong road and before long Cathy is missing,' which does not syntactically but semantically matches with the query. It is practically impossible to reach this page from the above query by the existing WWW search engines, but this retrieval is tractable by interactively traversing the semantic dependency structure (corresponding to syntactic dependency, anaphora, and so on) such as follows:



It is very hard to think of expressions like `British nurse is missing' from query `a forein nurse disappears' in one shot, but it is rather easy to first grasp the`nurse' node in the above network and then interactively extend the query by adding `British' and `missing' while narrowing down the search space. Candidates for new keywords to add (such as `British' in this case) can be presented to the user together with other candidates (which are connected with `nurse' somewhere in the search space), and the user can choose among them. If course such an attachment of new keywords is applicable not just to the first keyword (`nurse' in this case) but other keyword in the query.

Anther example is query `Somebody sells a car to somebody else and buys it back for the same price' to retrieve http://www.wabba.net/grandpa.html. There are many such queries, which seem to constitute the most part of what we want to find, although these queries are not actually attempted now because people know that is useless under the current technology. If Linguistic DS and Semantic DS spread widely, IR will be a much more feasible task than it is now.

4 Conclusions

We conclude that the Semantic DS is as expressive and useful as it can be, modulo the restriction from expression possible by constraining relationships between entities in the Semantic DS. Further work on the specification of some components of the Semantic DS is necessary.

We summarize the results, the recommendations, and the open issues below.

4.1 Summary of Results

The major results of this experiment are the following:

- Revised syntax of the Semantic DS.
- Formal validation of the descriptive power and usability of the Semantic DS.
- Description examples of the Semantic DS.

4.2 Recommendations

The main recommendations of this CE are the following:

- The Semantic DS be promoted to MDS XM.
- The documentation for Concept be amended to allow references to the interior structure of a Concept from all entities in a SemanticDescription
- The description field in Classification Scheme be augmented to allow the description of a controlled term by a Concept. Reference to the term should allow optional retrieval of the Concept.
- The lists of acceptable relationships for SemanticGraph, ConceptGraph, and the Links between Semantic Entities should be construed as Classification Schemes, and created for application specific purposes. To the extent that MPEG-7 creates any of these lists, it should be from the point of view of establishing standard "profiles", not constraining the standard as a whole.

4.3 Open Issues and Future Work

We have identified the following open issues requiring further work:

- Specification of the Semantic DS. There are still issues to solve and components to clearly specify in the Semantic DS. Several components of the Semantic DS need to be harmonized with the new version of the MDS XM/WD and the outcome of other going CEs.
- Generate complex descriptions of semantic information using the Semantic DS.

References

[1] AHG on MPEG-7 Semantic information representation, "Report of the CE on the Semantic DS", ISO/IEC JTC1/SC29/WG11 MPEG00/M6062, Geneva, Switzerland, May/June 2000.

[2] Bauderon, M. "Parallel rewriting of graphs through the pullback approach." In SEGRAGRA'95, Lecture notes in Theoretical Computer Science, 1995.

[3] Bauderon, M. and H. Jacquet. "Node rewriting in graphs and hypergraphs: A categorical framework", U. Bordeaux technical report 1134-96, 1996.

[4] Corradini, A. U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Loewe. "Algebraic Approaches to Graph Transformation, Part I: Basic Concepts and Double Pushout Approach." U. di Pisa, Dipartimento di Informatica, Technical report: TR-96-17, 1996.

[5] Fauconnier, G. Mental Spaces. Cambridge: Cambridge University Press, 1994.

[6] Fauconnier, G. Mappings in Thought and Language. Cambridge: Cambridge University Press, 1997.

[7] Fauconnier, G. and M. Turner. “Conceptual Integration Networks.” Cognitive Science, (22)2,pp. 133-187.

[8] Goguen, J. “An Introduction to Algebraic Semiotics, with Application to User Interface Design.” Computation for Metaphor, Analogy and Agents, Chrystopher Nehaniv, ed Springer Lecture Notes in Artificial Intelligence, volume 1562, 1999, pages 242-291.

[9] Goldblatt, R. Topoi: The Categorical Analysis of Logic. New York: Elsevier Press, 1984.

[10] Grady, J.E., T. Oakley, and S. Coulson. “Blending and Metaphor.” (http://www.wam.umd.edu/~mturn/WWW/blendaphor.html).

[11] Jacquet, H. Une approche categorique de la reecriture de sommets dans les graphes. Ph.D. Thesis, U. Bordeaux, 1999.

[12] Lambek, J. and P.J. Scott. Introduction to higher order categorical logic. Cambridge: Cambridge University Press, 1986.

[13] Mac Lane, S. Categories for the Working Mathematician, 2nd ed. New York: Springer-Verlag, 1998.

[14] Mac Lane, S. and I. Moerdijk. Sheaves in Geometry and Logic: A First Introduction to Topos Theory. New York: Springer-Verlag, 1992.

[15] MDS Group, “Core Experiment on the Semantic DS”, ISO/IEC JTC1/SC29/WG11 MPEG99/N3123, Maui, Hawaii, December 1999.

[16] MDS Group, “Workplan for CE on Semantic DS”, ISO/IEC JTC1/SC29/WG11 MPEG00/N3263, Noordwijkerhout, The Netherlands, March 2000.

[17] MDS Group, “Workplan for CE on Semantic DS”, ISO/IEC JTC1/SC29/WG11 MPEG00/N3424, Geneva, Switzerland, May/June, 2000.

[18] Rising III, H. K. and A. Tabatabai, “Report on the Importance of Structure in Semantic Descriptions”, ISO/IEC JTC1/SC29/WG11 MPEG00/M5937, Noordwijkerhout, The Netherlands, March 2000.

Annex A: Specification of Semantic DS

In the next sections, we provide the syntactic, semantics, “pseudo UML”, and examples of the components of the Semantic DS, and “pseudo UML”. The pesudo UML is not completely in sync with DDL specification in some DSs.

1 Link to Media and Localization

4.4 References to Ds and DSs

4.4.1 ReferenceToSLocation D

This descriptor is a general tool for referencing the instantiation of SLocation DS.

4.4.1.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToSLocation D          -->
<!-- ################################################ -->

<complexType           name="ReferenceToSLocation"           base="Reference"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute      name="idref"      type="IDREF"      refType="SLocation"
use="optional"/>
</complexType>
```

4.4.1.2 Descriptor Semantics

Name	*Definition*
`ReferenceToSLocation`	Reference to the description of a SLocation. Shall refer to the id attribute of a SLocation.
`Href`	Reference to the description where the referenced SLocation is.
`Idref`	Identifier of the referenced SLocation that is contained within the description defined by `href`. This descriptor is derived from the Reference D and adds the constraint that the referenced element should be a SLocation.

4.4.1.3 Description Extraction

Not applicable.

4.4.1.4 Description Example

The ReferenceToSLocation D is used in several DSs that represent relationships among SLocations and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.1.5 Description Use

See section 6.3.

4.4.2 ReferenceToSTime D

This descriptor is a general tool for referencing the instantiation of STime DS.

4.4.2.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToSTime D              -->
<!-- ################################################ -->

<complexType name="ReferenceToSTime" base="Reference" derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute name="idref" type="IDREF" refType="STime" use="optional"/>
</complexType>
```

4.4.2.2 Descriptor Semantics

Name	*Definition*
ReferenceToSTime	Reference to the description of a STime. Shall refer to the id attribute of a Stime.
Href	Reference to the description where the referenced STime is.
Idref	Identifier of the referenced STime that is contained within the description defined by `href`. This descriptor is derived from the Reference D and adds the constraint that the referenced element should be a STime.

4.4.2.3 Description Extraction

Not applicable.

4.4.2.4 Description Example

The ReferenceToSTime D is used in several DSs that represent relationships among STimes and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.2.5 Description Use

See section 6.3.

4.4.3 ReferenceToSemanticElement D

This descriptor is a general tool for referencing the description of a semantic element. Examples of semantic elements are objects, concepts, semantic graphs, and states.

4.4.3.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToSemanticElement D    -->
<!-- ################################################ -->

<complexType        name="ReferenceToSemanticElement"        base="Reference"
derivedBy="restriction"/>
```

4.4.3.2 Descriptor Semantics

Name	*Definition*
ReferenceToSemanticElement	Reference to the description of a semantic element. Shall refer to the id attribute of a semantic element.
Href	Reference to the description where the referenced semantic element is.
Idref	Identifier of the referenced semantic element that is contained within the description defined by `href`. This descriptor is derived from the Reference D and adds the constraint that the referenced element should be a semantic element.

4.4.3.3 Description Extraction

Not applicable.

4.4.3.4 Description Example

The ReferenceToSemanticElement D is used in several DSs that represent relationships among semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.3.5 Description Use

See section 6.3.

4.4.4 ReferenceToConcept D

4.4.4.1 This descriptor is a general tool for referencing the description of a concept.Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToConcept D            -->
<!-- ################################################ -->

<complexType    name="ReferenceToConcept"    base="ReferenceToSemanticElement"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute name="idref" type="IDREF" refType="Concept" use="optional"/>
</complexType>
```

4.4.4.2 Descriptor Semantics

Name	*Definition*
ReferenceToConcept	Reference to the description of a semantic concept. Shall refer to the id attribute of a concept.
Href	Reference to the description where the referenced concept is.
Idref	Identifier of the referenced concept that is contained within the description defined by `href`. This descriptor is derived from the Reference D and adds the constraint that the referenced element should be a concept.

4.4.4.3 Description Extraction

Not applicable.

4.4.4.4 Description Example

The ReferenceToConcept D is used in several DSs that represent relationships among concepts and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.4.5 Description Use

See section 6.3.

4.4.5 ReferenceToConceptEvent D

This descriptor is a general tool for referencing the description of an concept event.

4.4.5.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToConceptEvent D       -->
<!-- ################################################ -->

<complexType  name="ReferenceToConceptEvent"  base="ReferenceToSemanticElement"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute     name="idref"     type="IDREF"     refType="ConceptEvent"
use="optional"/>
</complexType>
```

4.4.5.2 Descriptor Semantics

Name	*Definition*
ReferenceToConceptEvent	Reference to the description of a semantic event concept. Shall refer to the id attribute of an event concept.
Href	Reference to the description where the referenced event concept is.
Idref	Identifier of the referenced event concept that is contained within the description defined by href. This descriptor is derived from the ReferenceToSemanticElement D and adds the constraint that the referenced element should be an event concept.

4.4.5.3 Description Extraction

Not applicable.

4.4.5.4 Description Example

The ReferenceToConceptEvent D is used in several DSs that represent relationships among event concepts and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.5.5 Description Use

See section 6.3.

4.4.6 ReferenceToConceptObject D

This descriptor is a general tool for referencing the description of an object concept.

4.4.6.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToConceptObject D      -->
<!-- ################################################ -->

<complexType name="ReferenceToConceptObject" base="ReferenceToSemanticElement"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute     name="idref"     type="IDREF"     refType="ConceptObject"
use="optional"/>
</complexType>
```

4.4.6.2 Descriptor Semantics

Name	*Definition*
`ReferenceToConceptObject`	Reference to the description of a semantic object concept. Shall refer to the id attribute of an object concept.
`Href`	Reference to the description where the referenced object concept is.
`Idref`	Identifier of the referenced object concept that is contained within the description defined by `href`. This descriptor is derived from the ReferenceToSemanticElement D and adds the constraint that the referenced element should be an object concept.

4.4.6.3 Description Extraction

Not applicable.

4.4.6.4 Description Example

The ReferenceToConceptObject D is used in several DSs that represent relationships among object concepts and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.6.5 Description Use

See section 6.3.

4.4.7 ReferenceToConceptGraph D

This descriptor is a general tool for referencing the description of an object concept.

4.4.7.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToConceptGraph D      -->
<!-- ################################################ -->

<complexType  name="ReferenceToConceptGraph"  base="ReferenceToSemanticElement"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute     name="idref"     type="IDREF"     refType="ConceptGraph"
use="optional"/>
</complexType>
```

4.4.7.2 Descriptor Semantics

Name	*Definition*
`ReferenceToConceptGraph`	Reference to the description of a semantic graph concept. Shall refer to the id attribute of an graph concept.
`Href`	Reference to the description where the referenced graph concept is.
`Idref`	Identifier of the referenced graph concept that is contained within the description defined by `href`. This descriptor is derived from the ReferenceToSemanticElement D and adds the constraint that the referenced element should be a graph concept.

4.4.7.3 Description Extraction

Not applicable.

4.4.7.4 Description Example

The ReferenceToConceptGraph D is used in several DSs that represent relationships among graph concepts and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.7.5 Description Use

See section 6.3.

4.4.8 ReferenceToConceptLink D

This descriptor is a general tool for referencing the description of an object concept.

4.4.8.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToConceptLink D      -->
<!-- ################################################ -->

<complexType  name="ReferenceToConceptLink"  base="ReferenceToSemanticElement"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute     name="idref"     type="IDREF"     refType="ConceptLink"
use="optional"/>
```

```
</complexType>
```

4.4.8.2 Descriptor Semantics

Name	*Definition*
`ReferenceToConceptLink`	Reference to the description of a semantic link concept. Shall refer to the id attribute of an link concept.
`Href`	Reference to the description where the referenced link concept is.
`Idref`	Identifier of the referenced link concept that is contained within the description defined by `href`. This descriptor is derived from the ReferenceToSemanticElement D and adds the constraint that the referenced element should be a link concept.

4.4.8.3 Description Extraction

Not applicable.

4.4.8.4 Description Example

The ReferenceToConceptLink D is used in several DSs that represent relationships among link concepts and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.8.5 Description Use

See section 6.3.

4.4.9 ReferenceToEvent D

This descriptor is a general tool for referencing the description of an event.

4.4.9.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToEvent D              -->
<!-- ################################################ -->

<complexType     name="ReferenceToEvent"     base="ReferenceToSemanticElement"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute name="idref" type="IDREF" refType="Event" use="optional"/>
</complexType>
```

4.4.9.2 Descriptor Semantics

Name	*Definition*
`ReferenceToEvent`	Reference to the description of a semantic Event. Shall refer to the id attribute of an event.
`Href`	Reference to the description where the referenced Event is.
`Idref`	Identifier of the referenced Event that is contained within the

	description defined by `href`. This descriptor is derived from the ReferenceToSemanticElement D and adds the constraint that the referenced element should be an event.

4.4.9.3 Description Extraction

Not applicable.

4.4.9.4 Description Example

The ReferenceToEvent D is used in several DSs that represent relationships among events and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.9.5 Description Use

See section 6.3.

4.4.10 ReferenceToObject D

This descriptor is a general tool for referencing the description of an object.

4.4.10.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToObject D             -->
<!-- ################################################ -->

<complexType     name="ReferenceToObject"     base="ReferenceToSemanticElement"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute name="idref" type="IDREF" refType="Object" use="optional"/>
</complexType>
```

4.4.10.2 Descriptor Semantics

Name	*Definition*
`ReferenceToObject`	Reference to the description of a semantic object. Shall refer to the id attribute of an object.
`Href`	Reference to the description where the referenced object is.
`Idref`	Identifier of the referenced object that is contained within the description defined by `href`. This descriptor is derived from the ReferenceToSemanticElement D and adds the constraint that the referenced element should be an object.

4.4.10.3 Description Extraction

Not applicable.

4.4.10.4 Description Example

The ReferenceToObject D is used in several DSs that represent relationships among objects and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.10.5 Description Use

See section 6.3.

4.4.11 ReferenceToState D

This descriptor is a general tool for referencing the description of a state.

4.4.11.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToState D              -->
<!-- ################################################ -->

<complexType     name="ReferenceToState"     base="ReferenceToSemanticElement"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute name="idref" type="IDREF" refType="State" use="optional"/>
</complexType>
```

4.4.11.2 Descriptor Semantics

Name	*Definition*
`ReferenceToState`	Reference to the description of a semantic state. Shall refer to the id attribute of a state.
`Href`	Reference to the description where the referenced state is.
`Idref`	Identifier of the referenced state that is contained within the description defined by `href`. This descriptor is derived from the ReferenceToSemanticElement D and adds the constraint that the referenced element should be a state.

4.4.11.3 Description Extraction

Not applicable.

4.4.11.4 Description Example

The ReferenceToState D is used in several DSs that represent relationships among states and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.11.5 Description Use

See section 6.3.

4.4.12 ReferenceToSemanticGraph D

This descriptor is a general tool for referencing the description of a semantic graph.

4.4.12.1 Descriptor Syntax

```
<!-- ################################################ -->
<!--   Definition the ReferenceToSemanticGraph D      -->
<!-- ################################################ -->
```

```
<complexType name="ReferenceToSemanticGraph" base="ReferenceToSemanticElement"
derivedBy="restriction">
      <attribute name="href" type="uriReference" use="optional"/>
      <attribute     name="idref"     type="IDREF"     refType="SemanticGraph"
use="optional"/>
</complexType>
```

4.4.12.2 Descriptor Semantics

Name	*Definition*
ReferenceToSemanticGraph	Reference to the description of a semantic semantic graph. Shall refer to the id attribute of a semantic graph.
Href	Reference to the description where the referenced semantic graph is.
Idref	Identifier of the referenced semantic graph that is contained within the description defined by `href`. This descriptor is derived from the ReferenceToSemanticElement D and adds the constraint that the referenced element should be a semantic graph.

4.4.12.3 Description Extraction

Not applicable.

4.4.12.4 Description Example

The ReferenceToSemanticGraph D is used in several DSs that represent relationships among semantic graphs and other semantic elements. Typical examples are semantic relationship graphs. See section 6.3.

4.4.12.5 Description Use

See section 6.3.

4.5 Media Locators

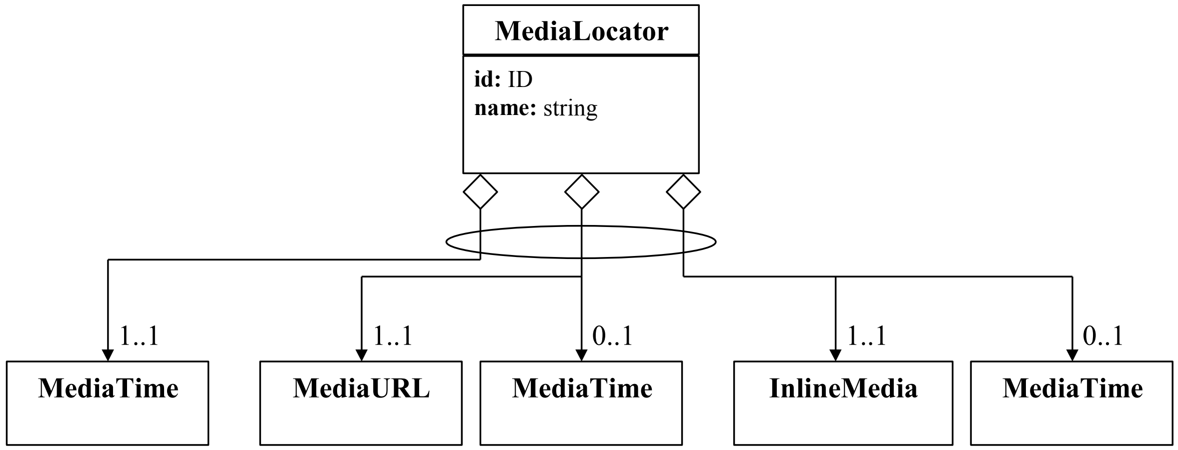
Media locators are used to specify the “location" of AV content.

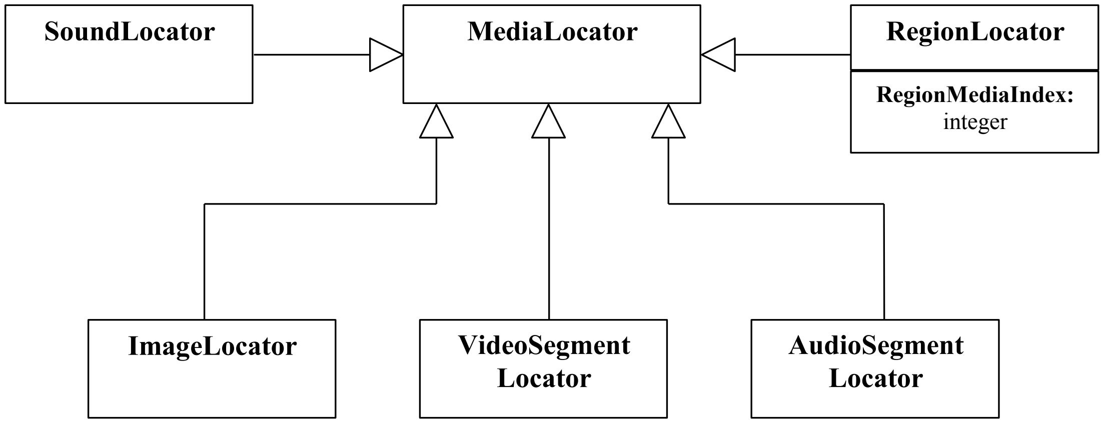
Note: Harmonize with MDS WD

4.5.1 MediaLocator DS

The MediaLocator DS is used to specify the “location" of a particular image, audio or video segment by referencing the media data. Five types of MediaLocators have been defined: the VideoSegmentLocator, the AudioSegmentLocator, the ImageLocator, the SoundLocator, and the RegionLocator.

4.5.1.1 Description Scheme Pseudo UML





4.5.1.2 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of MediaLocator DS                    -->
<!-- ################################################ -->

<complexType name="MediaLocator" abstract="true”>
      <choice>
            <sequence>
                  <element   name="MediaURL"   type="MediaURL"   maxOccurs=”1”
maxOccurs=”1”/>
                  <element   name="MediaTime"   type="MediaTime"   minOccurs="0"
maxOccurs=”1”/>
            </sequence>
            <element     name="MediaTime"     type="MediaTime"     maxOccurs="1"
maxOccurs="1"/>
      </choice>
</complexType>
```

4.5.1.3 Description Scheme Semantics

Name	*Definition*
MediaLocator	A description scheme to specify the location of media segments by referencing the media data.
MediaTime	A description scheme to specify time points or segments using the time information of the AV content.
MediaURL	A descriptor specifying the location of AV content using an URI.

4.5.1.4 Description Extraction

Since the MediaLocator DS is abstract, it cannot be instantiated as such. Extraction is therefore irrelevant for this DS.

4.5.1.5 Description Example

Since the MediaLocator DS is abstract, it cannot be instantiated as such. Examples are therefore irrelevant for this DS.

4.5.1.6 Description Use

Since the MediaLocator DS is abstract, it cannot be instantiated as such. Usage is therefore irrelevant for this DS.

4.5.2 VideoSegmentLocator DS

4.5.2.1 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of VideoSegmentLocator DS             -->
<!-- ################################################ -->

<complexType name="VideoSegmentLocator" base="MediaLocator"
             derivedBy="extension" />
```

4.5.2.2 Description Scheme Semantics

Name	*Definition*
VideoSegmentLocator	A media locator for video segments.

4.5.2.3 Description Extraction

Not applicable.

4.5.2.4 Description Example

In the example below the location of a video segment is specified by the URI of a video file and the relative start time with respect to the beginning of the file and the duration of the segment.

```
<VideoSegmentLocator>
      <MediaURL> http://www.mpeg7.org/demo.mpg </MediaURL>
      <MediaTime>
            <RelTime> <s>3</s> </RelTime>
            <Duration> <s>10</s> </Duration>
      </MediaTime>
</VideoSegmentLocator>
```

4.5.2.5 Description Use

This description scheme can be used whenever there is a need to specify the location of video segments with one of the contained mechanisms.

4.5.3 ImageLocator DS

Note: There is no VideoProgramLocator or AudioProgramLocator. Why do we need an ImageLocator DS? We can consider substituting the ImageLocator for the RegionLocator (moving or still)?

4.5.3.1 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of ImageLocator DS                    -->
<!-- ################################################ -->

<complexType name="ImageLocator" base="MediaLocator"
             derivedBy="extension" />
```

4.5.3.2 Description Scheme Semantics

Name	*Definition*
`ImageLocator`	A media locator for images.

4.5.3.3 Description Extraction

Not applicable.

4.5.3.4 Description Example

See VideoSegmentLocator DS in section 4.5.2.

4.5.3.5 Description Use

This description scheme can be used whenever there is a need to specify the location of images with one of the contained mechanisms.

4.5.4 AudioSegmentLocator DS

4.5.4.1 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of AudioSegmentLocator DS             -->
<!-- ################################################ -->

<complexType name="AudioSegmentLocator" base="MediaLocator"
             derivedBy="extension" />
```

4.5.4.2 Description Scheme Semantics

Name	*Definition*
`AudioSegmentLocator`	A media locator for audio segments.

4.5.4.3 Description Extraction

Not applicable.

4.5.4.4 Description Example

See VideoSegmentLocator DS in section 4.5.2.

4.5.4.5 Description Use

This description scheme can be used whenever there is a need to specify the location of audio segments with one of the contained mechanisms.

4.5.5 SoundLocator DS

The SoundLocator DS is used to locate a specific type of audio segments that by themselves describe scenes or events by sound effects.

4.5.5.1 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of SoundLocator DS                    -->
<!-- ################################################ -->

<complexType name="SoundLocator" base="AudioSegmentLocator"
             derivedBy="extension"/>
```

4.5.5.2 Description Scheme Semantics

Name	*Definition*
`SoundLocator`	A media locator for sound effects.

4.5.5.3 Description Extraction

Not applicable.

4.5.5.4 Description Example

See VideoSegmentLocator DS in section 4.5.2.

4.5.5.5 Description Use

This description scheme can be used whenever there is a need to specify the location of sound effects with one of the contained mechanisms.

4.5.6 RegionLocator DS

The RegionLocator DS is used to locate a specific region in an image or video program.

Note: RegionLocator is not in the MDS XM or WD. Its relationship to Region Locator (CE in The Netherlands) and Spatio-Temporal Locator (Visual XM) needs to be clarified.

4.5.6.1 Description Scheme Syntax

```
<!-- ############################################### -->
<!-- Definition of RegionLocator DS                  -->
<!-- ############################################### -->

<complexType name="RegionLocator" base="MediaLocator" derivedBy="extension">
      <attribute name=“RegionMediaIndex” type=“integer”/>
</complexType>
```

4.5.6.2 Description Scheme Semantics

Name	*Definition*
`RegionLocator`	A media locator for regions in visual material.
`RegionMediaIndex`	Definition is needed

4.5.6.3 Description Extraction

Not applicable.

4.5.6.4 Description Example

```
<!--  Example is needed  -->
```

4.5.6.5 Description Use

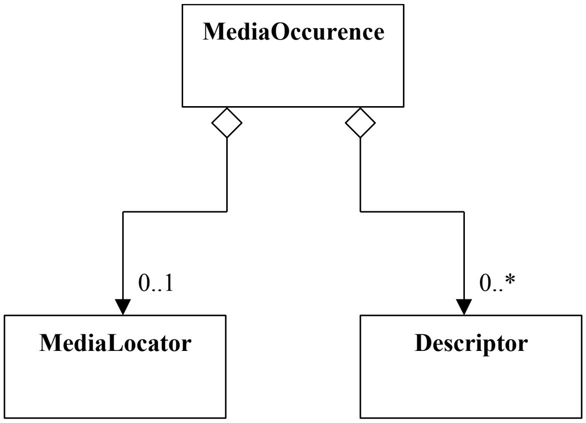
This description scheme can be used whenever there is a need to specify the location of regions with one of the contained mechanisms.

4.5.7 MediaOccurence DS

The MediaAnnotation DS records one appearance of an object/event in the media with a media locator and/or a set of descriptor values. It is the dual concept for Semantic DS of annotation DS for Segment DS.

Note: Clarity last sentence.

4.5.7.1 Description Scheme Pseudo UML



4.5.7.2 Description Scheme Syntax

```
<!-- ################################################## -->
<!-- Definition of MediaOccurence DS                    -->
<!-- ################################################## -->

<complexType name="MediaOccurence">
      <element    name="MediaLocator"    type="MediaLocator"    minOccurs="0"
maxOccurs="1"/>
      <element      name="Descriptor"      type="Examples"      minOccurs="0"
maxOccurs="unbounded"/>
</complexType>
```

4.5.7.3 Descriptor Semantics

Name	*Definition*
MediaOccurence	A description scheme that indicates the physical instance of an object/event with a media locator and/or a set of descriptor values. It is the dual concept for Semantic DS of annotation DS for segment DSs.
MediaLocator	A locator to the physical instances of semantic object/events.
Descriptor	A set of descriptor values describing the properties of the physical instance of an object/event located by Media Locator (i.e. state of object based on structure features), or a Descriptor. .It is of type Examples (see MDS WD)

4.5.7.4 Description Extraction

Manual instantiation is possible. For Descriptor DS, see the Examples DS section in MDS WD.

4.5.7.5 Description Example

See Object DS in section 6.2.5 for an example.

4.5.7.6 Description Use

This DS is used in the Object DS and the Event DS described in sections 6.2.5 and 6.2.4.

5 Basic Elements

5.1 Textual Description

5.1.1 TextAnnotation D

The TextAnnotation D is a tool for free text annotation.

5.1.1.1 Descriptor Syntax

```
<!-- #####################################################      -->
<!-- Definition of TextAnnotation D                             -->
<!-- #####################################################      -->

<complexType name="TextAnnotation" base="string" derivedBy="extension">
    <attribute name="mpeg7:lang" type="language" />
</complexType>
```

5.1.1.2 Descriptor Semantics

Name	*Definition*
`TextAnnotation`	Free textual annotation.
`xml:lang`	Attribute which specifies the language of the annotation with TextAnnotation D.

5.1.1.3 Description Extraction

Manual annotation.

5.1.1.4 Description Example

See Object DS in section 6.2.5 for examples.

5.1.1.5 Description Use

Retrieval application.

5.1.2 Term D

The Term D is a tool to choose from free text annotation and controlled term annotation.

5.1.2.1 Descriptor Syntax

```
<!--  Already available elsewhere  -->
```

5.1.2.2 Descriptor Semantics

Name	*Definition*
`Term`	Free or controlled text annotation.

5.1.2.3 Description Extraction

Manual annotation.

5.1.2.4 Description Example

See Object DS in section 6.2.5 for examples.

5.1.2.5 Description Use

Retrieval application.

5.2 Description of Places

5.2.1 Place DS

The Place DS contains the description tools (Ds and DSs) intended for a description of locations.

5.2.1.1 Description Scheme Syntax

```
<!-- #####################################################      -->
<!-- Definition of Place DS                                -->
<!-- #####################################################      -->

<element name="PlaceName">
      <complexType base="string" derivedBy="extension">
            <attribute name="mpeg7:lang" type="language"/>
      </complexType>
</element>
<element name="GPSCoordinates">
      <complexType base="string" derivedBy="extension">
            <attribute name="GPSsystem" type="ControlledTerm"/>
      </complexType>
</element>
<element name="Region"  type="ISO3166-2RegionCode"/>
<element name="PostingIdentifier" type="string"/>
<element name="AdministrativeUnit" type="string"/>
<element name="AddressLine" type="string"/>
<element name="PostalAddress">
      <complexType>
            <element ref="AddressLine"  maxOccurs="unbounded"/>
      </complexType>
</element>
<element name="InternalCoordinates" type="string"/>

<complexType name="Place">
      <element ref="PlaceName" maxOccurs="unbounded"/>
      <element name="PlaceRole" type="controlledTerm" minOccurs="0"/>
      <element name="Planet" type="controlledTerm" minOccurs="0"/>
      <element ref="GPSCoordinates" minOccurs="0" />
      <element ref="Country" minOccurs="0" />
      <element ref="Region" minOccurs="0" />
      <element ref="PostingIdentifier"/>
      <element ref="AdministrativeUnit"/>
      <element ref="PostalAddress"/>
      <element ref="InternalCoordinates"/>
      <attribute name="id" type="ID"/>
</complexType>
```

5.2.1.2 Description Scheme Semantics

Name	*Definition*
`Place`	DS describing a location.
`Id`	Identification of the instance of the location description.
`PlaceName`	The name of the place. Multiplicity is allowed for multilinguality.
`PlaceRole`	The role of the place: real –location, address- or fictional –setting- (e.g., Shooting location, Represented location, Postal Address, General Locator).
`Planet`	The name of the planet. Default value is "Earth".
`GPSCoordinates`	The coordinates of the place in a GPS system specified by an attribute.
`Country`	The country code using ISO 3166-1.
`Region`	The region code using ISO 3166-2.
`PostingIdenfier`	The ZIP or Postal code.
`AdministrativeUnit`	The name of the administrative unit (e.g., city, town, village, or even the USS Enterprise).
`PostalAddress`	The postal address including street (or square) and house number or name.
`InternalCoordinates`	Text used to identify location within unit with specified postal address (e.g., Flat 4, the drawing room, etc).

5.2.1.3 Description Extraction

Manual instantiation.

5.2.1.4 Description Example

```
<!-- specification -->
<DSTypeRef name='Location' type='Place'/>

<!-- description -->
<Location>
      <PlaceName xml:lang='en'>Rome</PlaceName>
      <PlaceName xml:lang='it'>Roma</PlaceName>
      <Country>it</Country>
      <PlaceRole>shooting location</PlaceRole>
</Location>

<Address>
      <PlaceName xml:lang='en'>Madrid</PlaceName>
      <Country>es</Country>
      <PlaceRole>postal address</PlaceRole>
      <Planet>Earth</Planet>
      <GPSCoordinates GPSsystem=''>XXXX</GPSCoordinates>
      <Region>cam</Region>
      <PostingIdentifier>E-28040</PostingIdentifier>
      <AdminstrativeUnit>city</AdminstrativeUnit>
      <PostalAddress>
```

```
            Universidad Politécnica de Madrid,
            Ciudad Universitaria s/n
      </PostalAddress>
      <InternalCoordinates>C-306</InternalCoordinates>
</Address>
```

5.2.1.5 Description Use

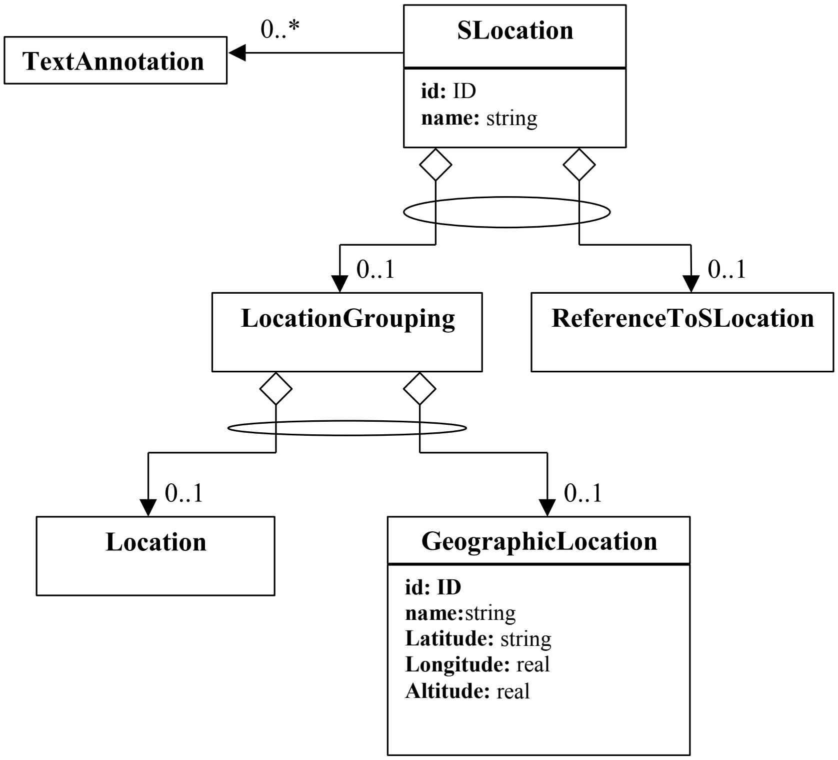
The Place DS is used as a primitive DS to allow the description of places, either real –addresses, locations- or fictional –setting-.

5.2.2 SLocation DS

The SLocation DS describes location information associated with an event.

Note: Relationship of SLocation DS and Place DS (in MDS WD) needs to be clarified. Slocation should also be synchronized with previous work of the Med/Met in describing real locations.

5.2.2.1 Description Scheme Pseudo UML



5.2.2.2 Description Scheme Syntax

```
<!-- ################################################## -->
```

```
<!-- Definition of SLocation DS                          -->
<!-- ################################################ -->

<!--     Recommendation:   Define   the   choice   between   SlocationDS   and
ReferenceToSLocation in element that include Slocation DS (i.e. take it ouside
the definion of Slocation itself).
      Reply: One motivation for keeping the syntax unchanged is to have the
possibility to incorporate in a (concept)event several Slocations, some of
which are references, and other new definitions
      Note: That does not make a difference  -->

<complexType name="SLocation">
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <choice>
            <element name="LocationGrouping">
                  <complexType>
                        <attribute       name="location"       type="string"
use=”optional”/>
                        <element                   name="GeographicLocation"
type="GeographicLocation"/>
                  </complexType>
            </element>
            <element name="ReferenceToSLocation" type="ReferenceToSLocation"/>
      </choice>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
</complexType>

<complexType name="GeographicLocation">
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
      <attribute name="latitude" type=“float”/>
      <attribute name="longitude" type=“float”/>
      <attribute name="altitude" type=“float”/>
</complexType>
```

5.2.2.3 Description Scheme Semantics

Note: There is a need to clearly define the functionality of all the name attributes and annotation elements (e.g. Text Annotation) in Slocation DS.

Semantics for the SLocation DS.

Name	*Definition*
`Slocation`	Description scheme that represents the location information associated with an event. It includes mandatorily a name or a geographic location (latitude/longitude/altitude) or a reference to an already defined SLocation DS.
`Id`	Unique identifier for an instantiation of SLocation DS.
`Name`	Name of the location. Definition is needed.
`TextAnnotation`	Textual annotation and description of the location represented by SLocation DS.
`LocationGrouping`	Container for either a location or a geographic location. If this is just a container, it can be removed.

Name	*Definition*
Location	Name of the location. Definition is needed. These could be more than a name it could be an address, please, see Place DS above.
GeographicLocation	Geographic information associated with the location represented by Slocation DS.
ReferenceToSLocation	Reference to already existing description of SLocation DS.

Semantics for the Geographic Location DS.

Name	Definition
GeographicLocation	Geographic information associated with a location.
Id	Unique identifier for an instantiation of SLocation DS. Evaluate necessity of this attribute.
Name	Name of the geographic location. Definition is needed. What is the difference with Name attribute in Slocation.
Latitude	Real value describing the latitude of the geographic location.
Longitude	Real value describing the longitude of the geographic location.
Altitude	Real value describing the altitude of the geographic location.

5.2.2.4 Description Extraction

Manual instantiation.

5.2.2.5 Description Examples

```
<!--  Example 1  -->
<SLocation id=”loc1” name=”my vacation site”>
      <LocationGrouping location=”Paris”/>
</SLocation>

<!--  Example 2  -->
<SLocation id="Loc 1" name="My vacation site">
      <TextAnnotation>My vacation site <TextAnnotation>
      <LocationGrouping location="Paris">
<GeographicLocation     id="Geo     1"     name="Paris"     latitude="48.867"
longitude="2.333"/>
      </LocationGrouping>
</SLocation>
```

5.2.2.6 Description Use

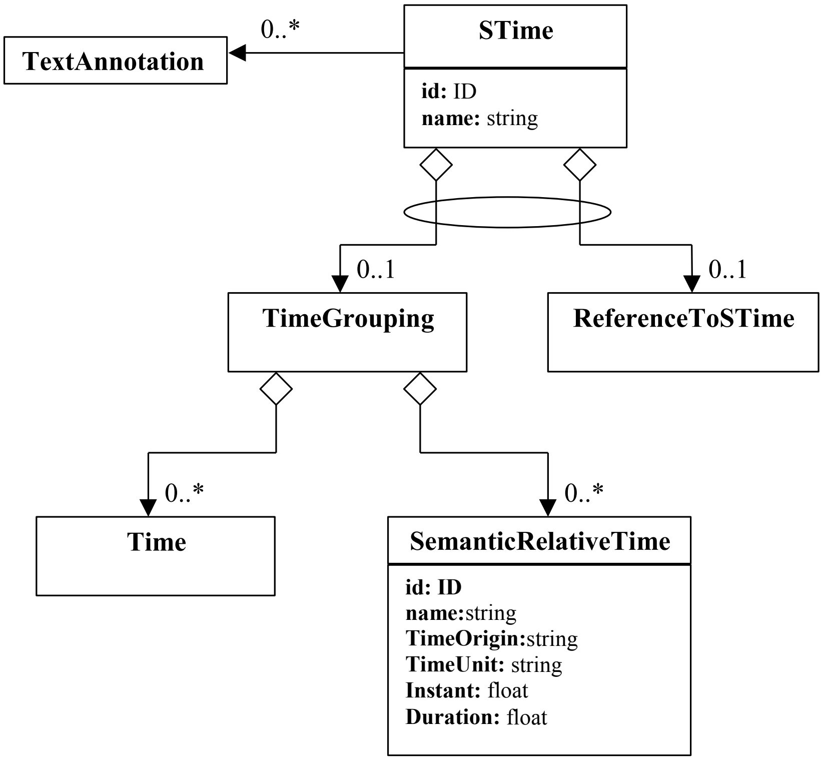
5.3 Description of Time

5.3.1 STime DS

The STime DS represents time information associated with an event.

Note: STime DS should be syncronized with the previous work on the Media/Meta DSs to describe semantic time information.

5.3.1.1 Description Scheme Pseudo UML



5.3.1.2 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of STime DS                           -->
<!-- ################################################ -->

<!—-   See  comments  for  SLocation  DS  about  moving  the  choice  outside  the
element. Possibly ok this time; see answer for SLocation -->

<complexType name="STime">
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <choice>
            <element name="TimeGrouping">
                  <complexType>
                        <element    name="Time"    type="Time"    minOccurs="0"
maxOccurs="unbounded"/>
                        <element                   name="SemanticRelativeTime"
type="SemanticRelativeTime" minOccurs="0" maxOccurs="unbounded"/>
                  </complexType>
            </element>
            <element name="ReferenceToSTime" type="ReferenceToSTime"/>
      </choice>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
</complexType>

<complexType name="SemanticRelativeTime">
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
      <attribute name="TimeOrigin" type="string"/>
      <attribute name="TimeUnit" type="string"/>
      <attribute name="Instant" type=“float” use=”optional”/>
      <attribute name="Duration" type=“float” use=”optional”/>
</complexType>
```

5.3.1.3 Description Scheme Semantics

Semantics for the STime DS.

Name	*Definition*
Stime	Description scheme that represents semantic time information associated with an event. It can either contain a Time DS or a SemanticRelativeTime DS. If no such information is available the TextAnnotation D may be used. It is also possible to refer to a previously defined description of the semantic time through a reference to STime DS.
Id	Unique identifier for an instantiation of STime DS.
Name	Name of time. Definition is needed.
TextAnnotation	Textual annotation and description of the time represented by STime DS.
TimeGrouping	Container for either a Time DS or a SemanticRelativeTime DS. If this is just a container, it can be removed.
Time	Represents a time point or duration.
SemanticRelativeTime	Description scheme that includes a time origin, a unit time interval, and either a duration or a time instant expressed in terms of these unit time intervals.
ReferenceToSTime	Reference to already existing description of STime DS.

Semantics for the Semantic Relative Time DS.

Name	*Definition*
SemanticRelativeTime	A description scheme to describe semantic relative time. It includes a time origin, a unit time interval, and either a duration or a time instant expressed in terms of these unit time intervals. Definition needs improvement
Id	Unique identifier for an instantiation of SemanticRelativeTime DS.
Name	Name of semantic relative time. Definition? What is the difference with Name of Stime?

Name	*Definition*
TimeOrigin	String specifying the time origin. It is a time point defined in semantic terms, e.g.“the day before yesterday”.
TimeUnit	String specifying the time unit. It is a time unit defined in semantics terms, e.g. “clock cycle”.
Instant	Float specifying time instant. It may contain fractions of interval units.
Duration	Float specifying a temporal duration. It may contain fractions of interval units. Note that when a duration is defined, it is not be necessary to have a TimeOrigin. How do all the four attribute above related to each other?

5.3.1.4 Description Extraction

Manual instantiation.

5.3.1.5 Description Example

```
<STime id="Time 1" name="My vacation">
      <TextAnnotation> My vacation </TextAnnotation>
      <TimeGrouping>
            <Time>
                  <TimePoint TZD="+100">
                        <Y>1989</Y>
                        <M>10</M>
                        <D>3</D>
                        <h>14</h>
                        <m>13</m>
                  </TimePoint>
                  <Duration>
                        <No_D>10</No_D>
                  </Duration>
            </Time>
      </TimeGrouping>
</STime>
```

5.3.1.6 Description Use

5.4 Description of Usage

5.4.1 UsageLabel D

The UsageLabel D specifies the usage type of a description as accessing or descriptive.

Note: Further explanation of accessing and descriptive is needed

5.4.1.1 Descriptor Syntax

```
<!-- ################################################ -->
<!-- Definition of Usage Label D                       -->
<!-- ################################################ -->
```

```
<simpleType name="UsageLabel" base="string">
      <enumeration value=”descriptive”/>
      <enumeration value=”accessing”/>
</simpleType>
```

5.4.1.2 Descriptor Semantics

Name	*Definition*
`UsageLabel`	Binary flag specifying whether a description is accessing or descriptive.

5.4.1.3 Description Extraction

5.4.1.4 Description Example

See Semantic Description DS in section 6.1.2.

5.4.1.5 Description Use

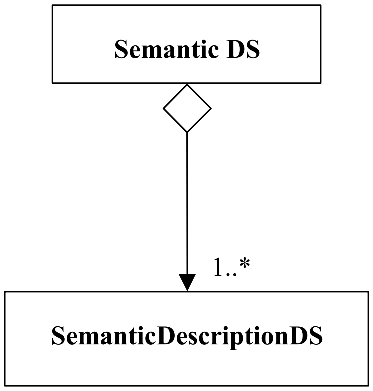
6 Description of the conceptual aspects of the content

6.1 Container Elements

6.1.1 Semantic DS

The Semantic DS contains all the semantic descriptions of a video program, an image, or any multimedia item.

6.1.1.1 Description Scheme Pseudo UML (needs update:0.. concepts)*



6.1.1.2 Description Scheme Syntax

```
<!-- ################################################ -->
```

```
<!-- Definition of Semantic DS                              -->
<!-- ############################################ -->

<complexType name="Semantic">
<element name="Concept" type="Concept" minOccurs="0" maxOccurs="unbounded"/>
      <element      name="SemanticDescription"      type="SemanticDescription"
minOccurs="1" maxOccurs="unbounded"/>
</complexType>
```

Note: What is the rationalization of keeping the Concept DS at this level as well as under the SemanticDescription? The UML at this level does not contain the Concept DS. Another question is on the semantics section. What do you mean by "a partial description of a piece of a multimedia document"? Are you forbidding the use of the SemanticDescription DS for a complete description of full document?

6.1.1.3 Description Scheme Semantics

Name	*Definition*
`Semantic`	Description scheme containing all semantic descriptions of a multimedia document.
`Concept`	Concept defined in a semantic description. Concepts represent templates for semantic objects and events.
`SemanticDescription`	Description scheme whose instantiation represent a partial description of a piece of a multimedia document.

6.1.1.4 Description Extraction

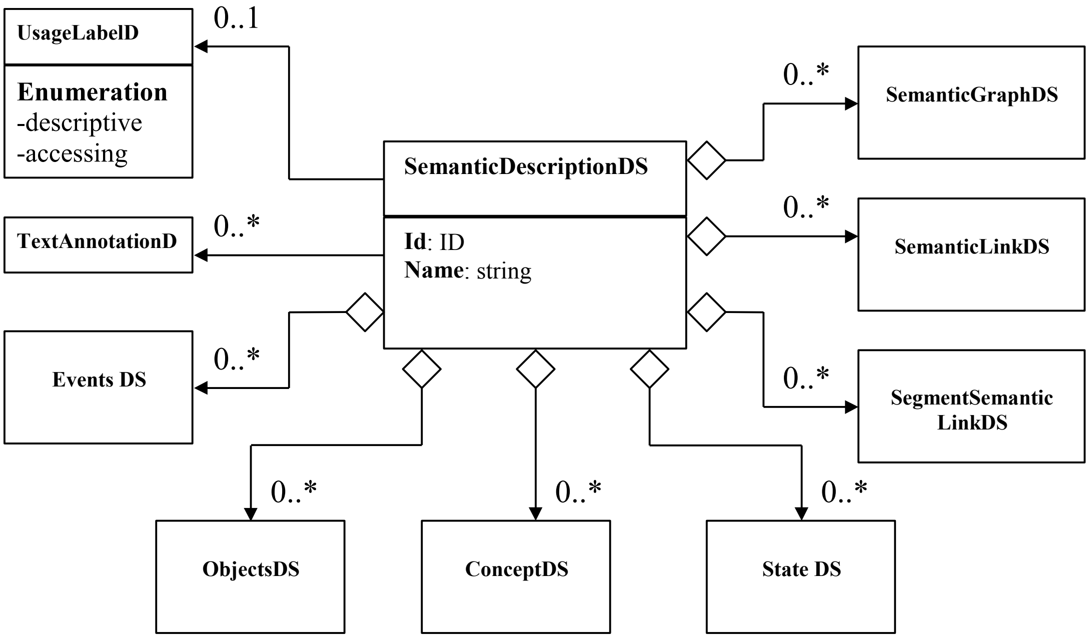
6.1.1.5 Description Example

6.1.1.6 Description Use

6.1.2 Semantic Description DS

The SemanticDescription DS represents a partial description of a portion of a multimedia document.

6.1.2.1 Description Scheme Pseudo UML



6.1.2.2 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of Semantic Description DS            -->
<!-- ################################################ -->

<complexType name="SemanticDescription">
      <element      name=”UsageLabel”      type="UsageLabel"      minOccurs="0"
maxOccurs=”1”/>
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <element        name="Concept"        type="Concept"        minOccurs="0"
maxOccurs=“unbounded”/>
      <element name="Event" type="Event" minOccurs="0" maxOccurs=“unbounded”/>
      <element         name="Object"         type="Object"        minOccurs="0"
maxOccurs=“unbounded”/>
      <element name="State" type="State" minOccurs="0" maxOccurs=“unbounded”/>
      <element    name="SemanticGraph"    type="SemanticGraph"    minOccurs="0"
maxOccurs=“unbounded”/>
      <element     name="SemanticLink"     type="SemanticLink"    minOccurs="0"
maxOccurs=“unbounded”/>
      <element      name="SegmentSemanticLink"      type="SegmentSemanticLink"
minOccurs="0" maxOccurs=“unbounded”/>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
</complexType>
```

6.1.2.3 Description Scheme Semantics

Name	*Definition*
`SemanticDescription`	Description scheme containing a partial description of a piece of multimedia document. Clarify partial description
`Id`	Unique identifier for an instantiation of SemanticDescription DS. It allows a semantic description to be referenced by other items. Can semantic descriptions be referenced?
`Name`	String describing the semantics of a semantic description. Evaluate if id and name should be mandatory.
`UsageLabel`	Binary flag specifying whether a description of SemanticDescription DS is accessing or descriptive (optional, i.e. if it is not used, it could be either).
`TextAnnotation`	Free text annotation describing the semantic description.
`Concept`	Concept defined in a semantic description. Concepts represent templates for semantic objects and events.
`Event`	Event defined in a semantic description. Events represent semantically meaningful entities in time.
`Object`	Object defined in a semantic description. Objects represent physical or abstract objects that are present or related to the multimedia document.
`State`	State defined in a semantic description. States are sets of numerical and verbal attributes that can be attached to semantic entities such as objects and events, or other semantic elements such a semantic relationship graphs.
`SemanticGraph`	Graph structures describing semantic relationships among semantic elements such as object, events, and concepts.
`SemanticLink`	Description scheme that defines the type of linked components and the relationships (with possible attached weights) between components of the Semantic DS, and between components of the Semantic DS.
`SegmentSemanticLink`	Description scheme that defines the type of linked components and the relationships (with possible attached weights) between semantic elements and segments.

6.1.2.4 Description Extraction

6.1.2.5 Description Example

The example below represents the following description: “John met Rita in New York on Saturday Night”.

```
<SemanticDescription>
<!-- Object John -->
<Object id="543628" name"John">
```

```
      <MediaAnnotation>
            <MediaLocator>
                     <MediaURL>http://www.mpeg7.org/Jonh.mpg </MediaURL>
                     <MediaTime>
                             <RelTime> <s>234</s> </RelTime>
                             <Duration> <s>10</s> </Duration>
                     </MediaTime>
            </MediaLocator>
            <MediaLocator>
                     <MediaURL>http://www.mpeg7.org/John.mpg </MediaURL>
                     <MediaTime>
                             <RelTime> <s>346</s> </RelTime>
                             <Duration> <s>10</s> </Duration>
                     </MediaTime>
            </MediaLocator>
      </MediaAnnotation>
</Object>

<!-- Object girl -->
<Object id="543328" name"Rita">
      <MediaAnnotation>
            <MediaLocator>
                     <MediaURL>http://www.mpeg7.org/Jonh.mpg </MediaURL>
                     <MediaTime>
                             <RelTime> <s>234</s> </RelTime>
                             <Duration> <s>10</s> </Duration>
                     </MediaTime>
            </MediaLocator>
      </MediaAnnotation>
</Object>

<!-- Event: Meeting between John and Rita -->
<Event id="8582374" name="John mets a girl" -->
      <MediaAnnotation>
            <MediaLocator>
                     <MediaURL>http://www.mpeg7.org/Jonh.mpg </MediaURL>
                     <MediaTime>
                             <RelTime> <s>234</s> </RelTime>
                             <Duration> <s>10</s> </Duration>
                     </MediaTime>
            </MediaLocator>
      </MediaAnnotation>
      <!-- Place DS is used instead of SLocation -->
      <Place>
            <PlaceName xml:lang=’en’>New York</PlaceName>
            <Country>usa</Country>
            <Planet>Earth</Planet>
            <GPSCoordinates GPSsystem=’’>XXXX</GPSCoordinates>
            <PostingIdentifier>E-28040</PostingIdentifier>
            <PostalAddress>
                  Ferlito's Pub, 122 5th Street, NY
            </PostalAddress>
            <InternalCoordinates>C-306</InternalCoordinates>
      </Place>
      <STime>
            <TimeGrouping>
                  <SemanticRelativeTime id="7842389"
                        name="Last Saturday Night"
                        TimeOrigin=”Wednesday”
                        TimeUnit=”Day”
                        Instant=”-4”/>
            </TimeGrouping>
      </STime>
</Event>
```

```
<SemanticLink>
      <Relationship>is present during</Relationship>
      <ReferenceToObject idref="543628">
      <ReferenceToEvent idref="8582374">
</SemanticLink>

<SemanticLink>
      <Relationship>is present during</Relationship>
      <ReferenceToObject idref="543328">
      <ReferenceToEvent idref="8582374">
</SemanticLink>
</SemanticDescription>
```

6.1.2.6 Description Use

6.2 Semantic Entities

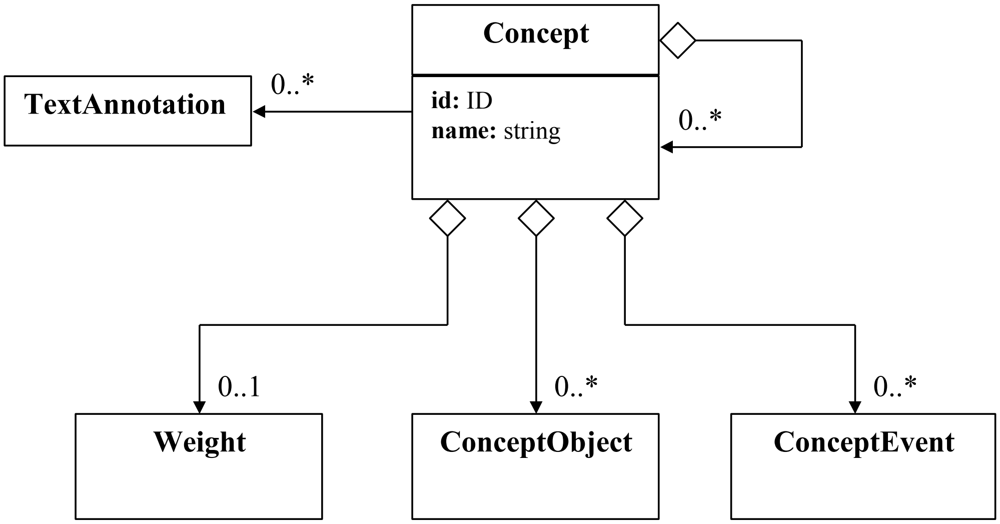
6.2.1 Concept DS

The Concept DS describes a template for objects or events.

Note: There are concepts which do not correspond to either an object or an event. Then, it seems that semantic descriptions could be instantiations of some concepts (not only objects and events). In this case, we should allow to describe relationships among concepts and semantic descriptions.

Note: Do we want to distinguish between object concepts and event concepts? ConceptObject and ConceptEvent do not seem to have that functionality.

6.2.1.1 Description Scheme Pseudo UML (needs update: ConceptGraph, ConceptLink)



6.2.1.2 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of Concept DS                         -->
<!-- ################################################ -->

<complexType name="Concept">
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <element name="Weight" type="Weight" minOccurs="0" maxOccurs="1"/>
      <element       name="Concept"       type="Concept"        minOccurs="0"
maxOccurs=“unbounded”/>
      <element    name="ConceptEvent"    type="ConceptEvent"    minOccurs="0"
maxOccurs=“unbounded”/>
      <element    name="ConceptGraph"    type="ConceptGraph"    minOccurs="0"
maxOccurs=“unbounded”/>
      <element     name="ConceptLink"     type="ConceptLink"     minOccurs="0"
maxOccurs=“unbounded”/>
      <element    name="ConceptObject"    type="ConceptObject"    minOccurs="0"
maxOccurs=“unbounded”/>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
</complexType>
```

6.2.1.3 Description Scheme Semantics

Name	*Definition*
`Concept`	Description scheme that describes a template for objects and events. The instantiation of concept should coherent with the definition of the concept itself. Objects and events contained in a Concept DS can only be linked to real data to serve as examples for the concept itself. As such, they should not refer to the content being described. Relationships relating objects and events within a concept DS are allowed; however relationships to events and objects outside the concept itself are not allowed.
`Id`	Unique identifier for an instantiation of Concept DS.
`Name`	String describing the semantics of a concept.
`TextAnnotation`	Free text annotation describing a concept.
`Weight`	Description scheme that enables the description of relative ordering, confidence, and perceptual significance of the different elements in the description such as concepts, concept events, and concept objects within this DS. Which exact elements at this level? Only other concept* DSs?
`Concept`	Concepts resulting from the decomposition of the parent concept. The Concept DS is allowed to recuerse.
`ConceptEvent`	Event contained in the concept definition. Concept events can only be linked to real data to serve as examples for concepts themselves. As such, they can not refer to the content being

Name	*Definition*
	described. Relationships among object and events within a concept can be described; however, relationships to events and objects outside a concept are not allowed.
`ConceptGraph`	SemanticGraph contained in the concept definition. It is allowed to connect through relationships only components of the concept in which it is defined.
`ConceptLink`	List of semantic links attached to the concept connecting only its individual components through basic relationships
`ConceptObject`	Object contained in the concept definition. Concept objects can only be linked to real data to serve as examples for concepts themselves. As such, they can not refer to the content being described. Relationships among object and events within a concept can be described; however, relationships to events and objects outside a concept are not allowed.

6.2.1.4 Description Extraction

6.2.1.5 Description Example

In the example below, the concept "natural father" is described instantiating Concept DS.

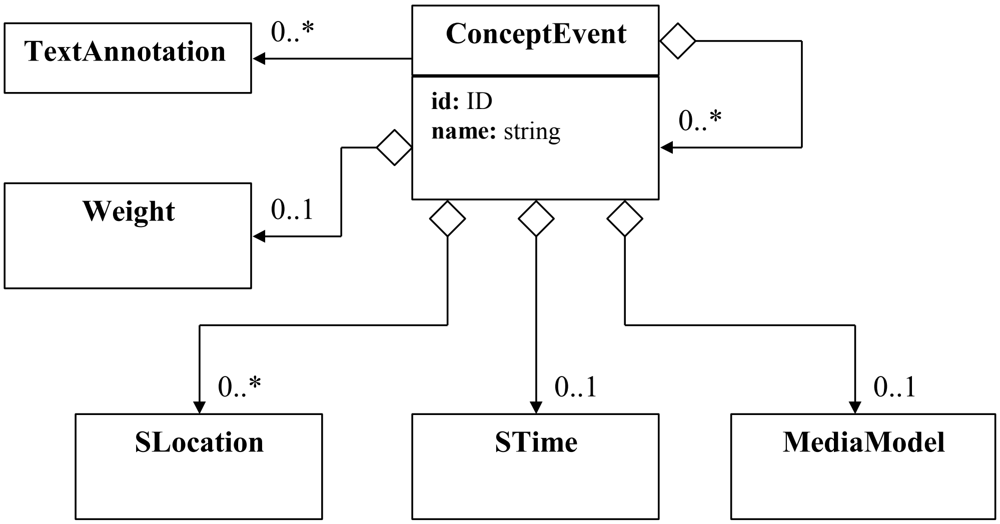
```
<!--  There is a problem in this definition: the relationship between the two
objects (man and child) and the event (the birth) is not clear.  -->

<Concept id="Concept 1" name="Natural father">
      <TextAnnotation>
            Natural father is a man has begotten a child
      </TextAnnotation>
      <ConceptObject id="ConObj 1" name="Man">
            <!-- the same things like lion (for a man obviously..)-->
      </ConceptObject>
      <ConceptObject id="ConObj 2" name="Child">
            <!-- ... -->
      </ConceptObject>
      <ConceptEvent id="ConEv 1" name="Birth">
            <!-- ... -->
      </ConceptEvent>
</Concept>
```

6.2.1.6 Description Use

6.2.2 ConceptEvent DS

The ConceptEvent DS represents an event that participates in the definition of a concept. As such, it can not refer to the media content itself.

6.2.2.1 Description Scheme Pseudo UML (needs update: 0.. MediaModel)*



6.2.2.2 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of Concept Event DS                   -->
<!-- ################################################ -->

<complexType name="ConceptEvent">
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <element    name="MediaModel"    type="AnalyticModel"    minOccurs="0"
maxOccurs="unbounded"/>
      <element name="STime" type="STime" minOccurs="0" maxOccurs="1"/>
      <element     name="SLocation"     type="SLocation"     minOccurs="0"
maxOccurs=”unbounded”/>
      <element name="Weight" type="Weight" minOccurs="0" maxOccurs="1"/>
maxOccurs=“unbounded”/>
      <element    name="ConceptEvent"    type="ConceptEvent"    minOccurs="0"
maxOccurs=“unbounded”/>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
</complexType>
```

6.2.2.3 Description Scheme Semantics

Name	*Definition*
`ConceptEvent`	Description scheme that represents an event that participates in the definition of a concept. These concept events can only be linked to real data to serve as examples for concepts themselves. As such, they can not refer to the content being described. The syntax of ConceptEvent DS is identical to Event DS apart from not containing any description of the media, i.e., MediaAnnotation DSs or links to Segment DSs.

Name	*Definition*
`Id`	Unique identifier for an instantiation of ConceptEvent DS.
`Name`	String describing the semantics of a concept event.
`TextAnnotation`	Free text annotation describing a concept event.
`MediaModel`	Description scheme that specifies an abstract description of a concept event by a set of media locators and/or a set of value specifications for descriptors. There is clearly only one model per ConceptEvent ; a large number of MediaModel in the definition is the only way to attach more than one media locator and/ or one descriptor value.
`STime`	Semantic time information associated with the concept event.
`SLocation`	Semantic location information associated with the concept event.
`Weight`	Description scheme that enables the description of relative ordering, confidence, and perceptual significance of the different elements in the description such as concept events within this DS. Which exact elements at this level? Only other concept events?
`ConceptEvent`	Concept events resulting from the decomposition of the parent concept event. The ConceptEvent DS can be recursive.

6.2.2.4 Description Extraction

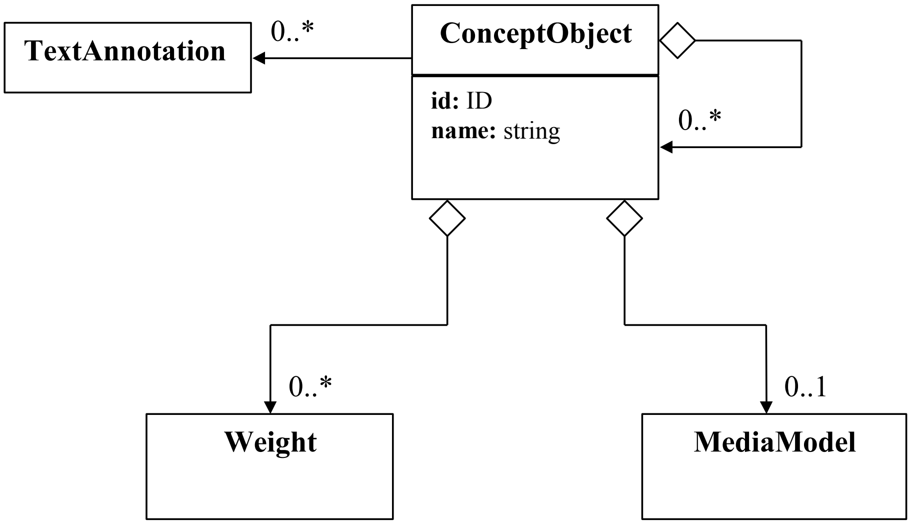
6.2.2.5 Description Example

6.2.2.6 Description Use

6.2.3 ConceptObject DS

The ConceptObject DS represents an object that participates in the definition of a concept. As such, it can not refer to the media content itself.

6.2.3.1 Description Scheme Pseudo UML (needs update: 0.. for MediaModel)*



6.2.3.2 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of Concept Object DS                  -->
<!-- ################################################ -->

<complexType name="ConceptObject"/>
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <element    name="MediaModel"    type="AnalyticModel"     minOccurs="0"
maxOccurs=“unbounded”/>
      <element name="Weight" minOccurs="0" maxOccurs="1"/>
      <element   name="ConceptObject"   type="ConceptObject"   minOccurs="0"
maxOccurs="unbounded"/>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
</complexType>
```

6.2.3.3 Description Scheme Semantics

Name	*Definition*
`ConceptObject`	Description scheme that represents an object that participates in the definition of a concept. These concept objects can only be linked to real data to serve as examples for concepts themselves. As such, they can not refer to the content being described. The syntax of ConceptObject DS is identical to Object DS apart from not containing any description of the media, i.e., MediaAnnotation DSs or links to Segment DSs.
`Id`	Unique identifier for an instantiation of ConceptObject DS.
`Name`	String describing the semantics of a concept object.
`TextAnnotation`	Free text annotation describing a concept object.

Name	*Definition*
MediaModel	Description scheme that specifies an abstract description of a concept object by a set of media locators and/or a set of value specifications for descriptors. There is clearly only one model per ConceptObject ; a large number of MediaModel in the definition is the only way to attach more than one media locator and/ or one descriptor value.
Weight	Description scheme that enables the description of relative ordering, confidence, and perceptual significance of the different elements in the description such as concept objects within this DS. Which exact elements at this level? Only other concept objects?
ConceptObject	Concept objects resulting from the decomposition of the parent concept object. The ConceptObject DS can be recursive.

6.2.3.4 Description Extraction

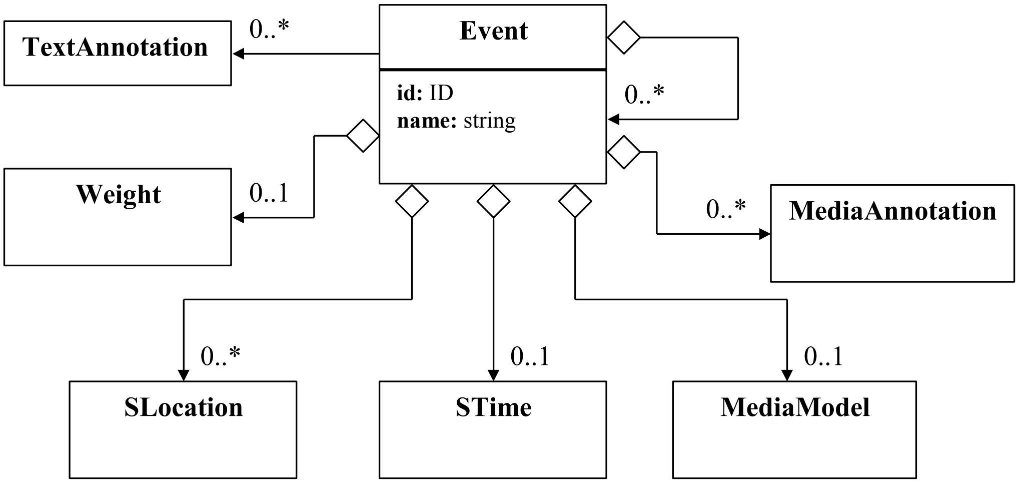
6.2.3.5 Description Example

6.2.3.6 Description Use

6.2.4 Event DS

The Event DS represents a semantically meaningful entity in time.

6.2.4.1 Description Scheme Pseudo UML (needs update 0..* MediaModel; 0..* MediaOccurence)



6.2.4.2 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of Event DS                           -->
<!-- ################################################ -->

<!--  Recommendation: Remove ReferenceToSTime within STime and have a choice
between the two in Event DS. The same for Slocation DS. This would simplify
the definition of Stime and Slocation.  ->

<complexType name="Event">
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <element     name="MediaModel"     type="AnalyticModel"     minOccurs="0"
maxOccurs="unbounded"/>
      <element   name="MediaOccurence"   type="MediaOccurence"   minOccurs="0"
maxOccurs="unbounded"/>
      <element name="STime" type="STime" minOccurs="0" maxOccurs="unbounded"/>
      <element      name="SLocation"      type="SLocation"      minOccurs="0"
maxOccurs="unbounded"/>
      <element name="Weight" type="Weight" minOccurs="0" maxOccurs="1"/>
      <element name="Event" type="Event" minOccurs="0" maxOccurs="unbounded"/>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
</complexType>
```

Note: Events can be recursive because sub-objects may not need to be defined outside the parent object. Event DS basically extends Object DS with multiple SLocation DSs to individuate where the object takes place and an optional STime DS to individuate when the object happens.

6.2.4.3 Description Scheme Semantics

Name	*Definition*
Event	Description scheme that represents a semantically meaningful entity in time. Events can be recursive because sub-events may not need to be defined outside the parent event. Event DS basically extends Object DS with multiple SLocation DSs to individuate where the event takes place and an optional STime DS to individuate when the event happens.
Id	Unique identifier for an instantiation of Event DS.
Name	String describing the semantics of an event.
TextAnnotation	Free text annotation describing an event.
MediaModel	Description scheme that specifies an abstract description of an event by a set of media locators and a set of value specifications for descriptors. There is clearly only one model per ConceptObject ; a large number of MediaModel in the definition is the only way to attach more than one media locator and/ or one descriptor value.
MediaOccurence	A description scheme that indicates the physical instance of an object/event with a media locator and/or a set of descriptor values. It is the dual concept for Semantic DS of annotation DS for segment DSs.
Stime	Semantic time information associated with the event.

Name	*Definition*
`Slocation`	Semantic location information associated with the event.
`Weight`	Description scheme that enables the description of relative ordering, confidence, and perceptual significance of the different elements in the description such as events within this DS. Which exact elements at this level? Only other events?
`Event`	Events resulting from the decomposition of the parent event. The Event DS can be recursive.

6.2.4.4 Description Extraction

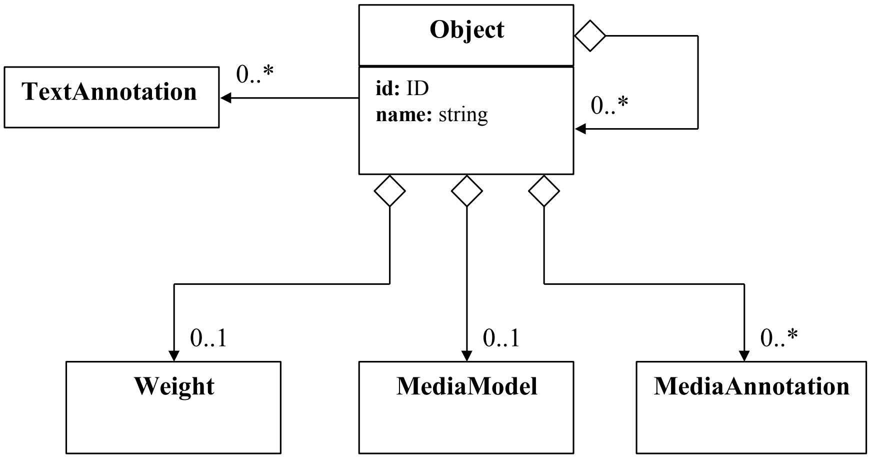
6.2.4.5 Description Example

6.2.4.6 Description Use

6.2.5 Object DS

In a description, Object DS represents a physical or abstract object that is present or is related to the multimedia document.

6.2.5.1 Description Scheme Pseudo UML (needs update: 0..* MediaModel, 0..* MediaOccurence)



6.2.5.2 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of Object DS                          -->
<!-- ################################################ -->

<!—- Note: Some participants think that Object DS should include STime and SLocation DSs as Event DS.
```

```
<!--  Note: Object DS and Event DS could be derived from ConceptEvent DS and
ConceptObject DS using extension if MPEG-7 DDL supports the restriction or
extension of the element within elements types. This is currently an open
issue of the MPEG-7 DDL.  -->

<complexType name="Object"/>
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <element    name="MediaModel"    type="AnalyticModel"    minOccurs="0"
maxOccurs=“unbounded”/>
      <element   name="MediaAnnotation"   type="MediaOccurence"   minOccur="0"
maxOccurs=“unbounded”/>
      <element name="Weight" type="Weight" minOccurs="0" maxOccurs="1"/>
      <element        name="Object"        type="Object"        minOccurs="0"
maxOccurs="unbounded"/>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
</complexType>
```

6.2.5.3 Description Scheme Semantics

Name	*Definition*
`Object`	Description scheme that represents physical or abstract object that is present or is related to the multimedia document. Events can be recursive because sub-objects may not need to be defined outside the parent object. Event DS basically extends Object DS with multiple SLocation DSs to individuate where the object takes place and an optional STime DS to individuate when the object happens.
`Id`	Unique identifier for an instantiation of Event DS.
`Name`	String describing the semantics of an object.
`TextAnnotation`	Free text annotation describing an object.
`MediaModel`	Description scheme that specifies an abstract description of an object by a set of media locators and a set of value specifications for descriptors. There is clearly only one model per ConceptObject ; a large number of MediaModel in the definition is the only way to attach more than one media locator and/ or one descriptor value.
`MediaOccurence`	A description scheme that indicates the physical instance of an object/event with a media locator and/or a set of descriptor values. It is the dual concept for Semantic DS of annotation DS for segment DSs.
`Weight`	Description scheme that enables the description of relative ordering, confidence, and perceptual significance of the different elements in the description such as objects within this DS. Which exact elements at this level? Only other objects?
`Object`	Objects resulting from the decomposition of the parent object. The Object DS can be recursive.

6.2.5.4 Description Extraction

6.2.5.5 Description Example

The example below describes a lion in a video program using the object DS. A model of a lion is described instantiating MediaModel DS. Two physical appearances of the lion in the video program are described using two instantiations of MediaAnnotation DS. Finally, the object lion is decomposed into lion head and lion body by using recursion among Object DSs.

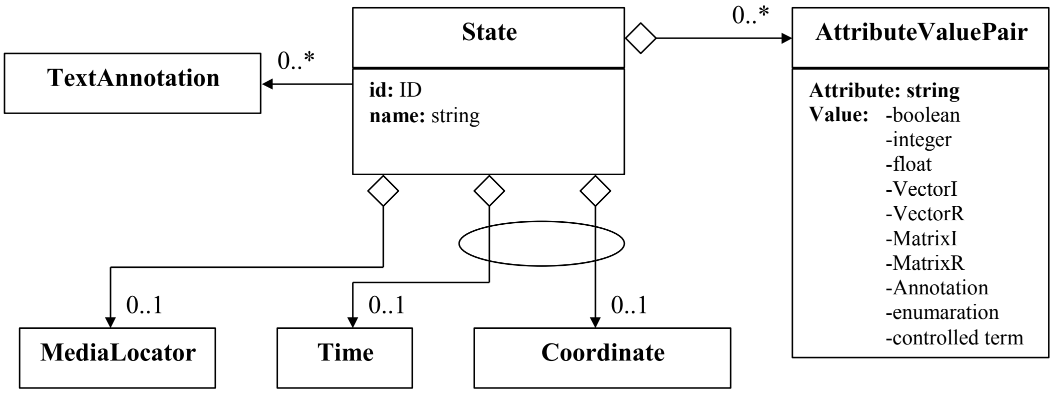
```
<Object id="Object 1" name="Lion">
      <TextAnnotation>
Large carnivore (Panthera Leo) of the cat family, found in a open country in
Africa, with a few surviving in India
      </TextAnnotation>

      <MediaModel>
            <MediaLocator>
                  <MediaURL> http://www.lion.org/cats.mpg </MediaURL>
                  <MediaTime>
                        <RelTime> <s>3</s> </RelTime>
                        <Duration> <s>10</s> </Duration>
                  </MediaTime>
            </MediaLocator>
            <ExamplesDS SemanticLabel="lion" Length="1"
Confidence="1.0" DescriptorName="ColorHistogram">
                  <Descriptor>
                        4617 11986 938 2628 458 1463 5178 2258 444 134 69 456
                        9300 2810 121 21 14 18 48 107 277 53 47 1926 8281 793
                        38 11 0 5 201 28 0 1 1 2 23 252 122 6 3 433 1517 46 1
                        1 0 0 0 0 0 0 0 0 2 55 13560 3326 678 221 1610 5602
                        916 32 8 1 21 58 11 1 0 0 2 61 331 179 14 7 2388 6213
                        51 0 0 0 0 0 0 0 0 0 0 2 337 243 0 0 220 194 0 0 0 0 0
                        0 0 0 0 0 0 0 383 3172 1072 51 20 91 128 0 0 0 0 0 2 4
                        0 0 0 0 89 757 694 0 0 217 39 0 0 0 0 0 0 0 0 0 0 0 0
                        912 210 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 55
                  </Descriptor>
            </Examples>
      </MediaModel>

      <MediaAnnotation>
            <MediaLocator>
                  <MediaTime>
                        <RelTime> <s>234</s> </RelTime>
                        <Duration> <s>105</s> </Duration>
                  </MediaTime>
            </MediaLocator>
            <Example SemanticLabel="lion in Savana" Length="1"
Confidence="1.0" DescriptorName="ColorHistogram">
                  <Descriptor>
                        4617 11986 938 2628 458 1463 5178 2258 444 134 69 456
                        9300 2810 121 21 14 18 48 107 277 53 47 1926 8281 793
                        38 11 0 5 201 28 0 1 1 2 23 252 122 6 3 433 1517 46 1
                        1 0 0 0 0 0 0 0 0 2 55 13560 3326 678 221 1610 5602
                        916 32 8 1 21 58 11 1 0 0 2 61 331 179 14 7 2388 6213
                        51 0 0 0 0 0 0 0 0 0 0 2 337 243 0 0 220 194 0 0 0 0 0
                        0 0 0 0 0 0 0 383 3172 1072 51 20 91 128 0 0 0 0 0 2 4
                        0 0 0 0 89 757 694 0 0 217 39 0 0 0 0 0 0 0 0 0 0 0
                        912 210 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 55
                  </Descriptor>
            </Example>
      </MediaAnnotation>

      <MediaAnnotation>
            <MediaLocator>
                  <MediaTime>
                        <RelTime> <s>1023</s> </RelTime>
                        <Duration> <s>234</s> </Duration>
                  </MediaTime>
            </MediaLocator>
            <Example SemanticLabel="lion in India" Length="1"
Confidence="1.0" DescriptorName="ColorHistogram">
                  <Descriptor>
                        9742 15760 1455 2216 475 1356 4771 2328 714 329 193
                        420 6954 6087 298 15 15 22 35 119 74 115 24 1253 7629
                        352 14 5 1 3 85 99 0 0 0 0 0 11 0 6 0 335 717 9 0 0 0
                        0 0 0 0 0 0 0 0 0 12332 3066 991 157 1048 4836 469 14
                        1 0 0 160 80 4 0 0 0 13 217 101 53 0 3450 6079 12 0 0
                        0 0 0 0 0 0 0 0 0 6 0 0 0 338 64 0 0 0 0 0 0 0 0 0 0
                        0 0 2439 718 15 0 81 41 0 0 0 0 0 0 0 0 0 0 0 0 65 0 0
                        0 447 43 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
                        0 0 0 0 0 0 0 0
                  </Descriptor>
            </Examples>
      </MediaAnnotation>

      <Object id="Object 2" name="Lion head">
            <TextAnnotation>The lion head<TextAnnotation>
      </Object>

      <Object id="Object 3" name="Lion body">
            <TextAnnotation>The lion body<TextAnnotation>
      </Object>

</Object>
```

6.2.5.6 Description Use

6.2.6 State DS

The State DS identifies semantic attributes of the entity at a given time, in a given spatial location, or in a given media location. It is a set of numerical and verbal attributes that can be attached to semantic entities such as objects and events and other semantic elements such as semantic relationship graphs.

6.2.6.1 Description Scheme Pseudo UML



6.2.6.2 Description Scheme Syntax

```
<!-- ############################################### -->
<!-- Definition of State DS                          -->
<!-- ############################################### -->

<!-- Note: Coordinate DS is not in the MDS XM or WD yet. -->
<!-- Note: Should Time and Coordinate not be Stime and Slocation instead? -->

<!-- Question: States associated with moving regions may required Time DS and
Coordinate DS at the same time. This is currently not supported.
      Reply: Agreed, may need to replace the time/coordinate definition with a
generalized coordinate representation -->


<complexType name="State">
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <element       name="AttributeValuePair"       type="AttributeValuePair"
minOccurs="0" maxOccurs=“unbounded”/>
      <choice minOccurs="1" maxOccurs="1">
            <element name="Time" type="Time"/>
            <element name="Coordinate" type="Coordinate"/>
      </choice>
      <element    name="MediaLocator"    type="MediaLocator"    minOccurs="1"
maxOccurs="1"/>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
</complexType>

<!--  Note: The following specification may not be correct (the same name
“value” for several element definitions). Check with MPEG-7 DDL group.  -->

<complexType name="AttributeValuePair"/>
      <element name="attribute" type="string" minOccurs=”1” maxOccurs=”1”/>
      <choice minOccurs="1" maxOccurs="1">
            <element name="value" type="boolean"/>
            <element name="value" type="integer"/>
            <element name="value" type=“float”/>
            <element name="value" type="VectorI"/>
            <element name="value" type="VectorR"/>
            <element name="value" type="MatrixI"/>
            <element name="value" type="MatrixR"/>
            <element name="value" type="AnnotationDS"/>
            <element name="value" type="enumeration"/>
            <element name="value" type="controlledTerm"/>
      </choice>
</complexType>
```

6.2.6.3 *Description Scheme Semantics*

Semantics for the State DS.

Name	*Definition*
`State`	Description scheme that identifies semantic attributes of the entity at a given time, in a given spatial location, or in a given media location. It is a set of numerical and verbal attributes that can be attached to semantic entities such as objects and events and other semantic elements such as semantic relationship graphs.
`Id`	Unique identifier for an instantiation of State DS.
`Name`	String describing the semantics of the state.
`TextAnnotation`	Free text annotation describing the state.
`AttributeValuePair`	Description scheme that contains a pair formed by an attribute and a value being the value one of a series of types (e.g. boolean, integer, real, matrix, string, and controlled text).
`Time`	Time information associated with the state.
`Coordinate`	Coordinate information associated with the state. Definition is needed.
`MediaLocator`	Locator to the physical instances of the state. Justify occurrence 1..1.

Semantics for the Attribute Value Pair DS.

Name	*Definition*
`AttributeValuePair`	Description scheme that contains a pair formed by an attribute and a value being the value one of a series of types (e.g. boolean, integer, real, matrix, string, and controlled text).
`Attribute`	String that specifies the name of the pair's attribute
`Value`	Specifies the value of the attribute. The type of this element is of a series of types (e.g. boolean, integer, real, and matrix).

6.2.6.4 Description Extraction

6.2.6.5 Description Example

6.2.6.6 Description Use

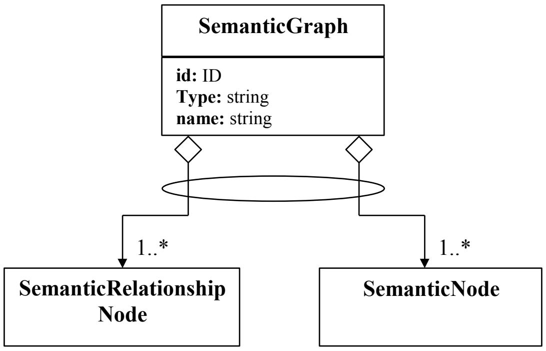
6.3 Semantic Relationship Graphs

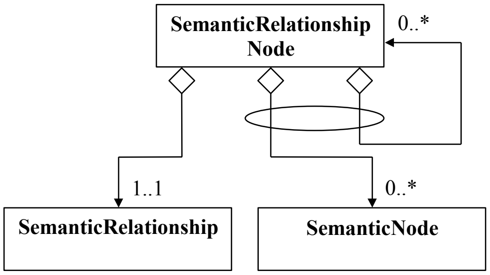
<!—- Question: What is the difference between Container Graph DS, Semantic Link DS, and Segment Semantic Link DS. All of these seem to describe relationships among content elements. Why are there 3 different mechanisms needed? However, it is not clear yet how to link a concept with an object instantiating it, an object with its states, the relationships among concept objects to form a concept, etc, etc.-->

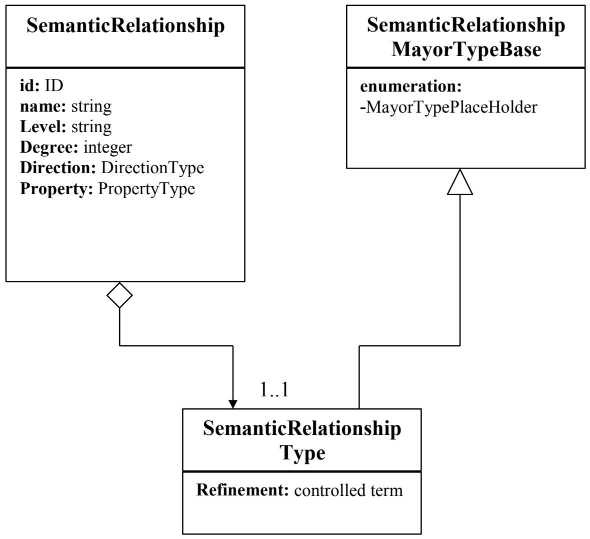
6.3.1 SemanticGraph DS

A semantic graph describes relationships among semantic entities, elements, and concepts such as objects, events, and other semantic graphs.

6.3.1.1 Description Scheme Pseudo UML







6.3.1.2 Description Scheme Syntax

```
<!-- ############################################### -->
<!-- Definition of Semantic Graph DS                 -->
<!-- ############################################### -->

<complexType name="SemanticGraph">
      <choice minOccurs=”1” maxOccurs=“unbounded”>
            <element                          name="SemanticRelationshipNode"
type="SemanticRelationshipNode"/>
            <element name="SemanticNode" type="SemanticNode"/>
      </choice>
      <attribute name="id" type="ID"/>
      <attribute name="type" type="string"/>
      <attribute name="name" type="string"/>
</complexType>


<!-- ############################################### -->
<!-- Definition of Semantic Relationship DS          -->
<!-- ############################################### -->

<!—- Note: SemanticRelationship could be derived from Relationship (in the MDS
WD). Then, the definition below will be greatly simplified. What is the need
for defining the type of relationship as an element and for it to be
compulsory? Is this is also convenient for the Segment Relationship, maybe
this mechanism can be included in Relationship DS. -->

<complexType name="SemanticRelationship">
      <element name="SemanticRelationshipType" type="SemanticRelationshipType"
minOccurs="1" maxOccurs="1"/>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string" use=”required”/>
      <attribute name="level" type="string"/>
      <attribute name="degree" type="integer"/>
      <attribute name="direction" type="DirectionType"/>
      <attribute name="property" type="PropertyType"/>
</complexType>

<complexType                                   name="SemanticRelationshipType"
base="SemanticRelationshipMajorTypeBase" derivedBy="extension">
      <attribute name="refinement" type="controlledTerm"/>
</complexType>

<!--  The definition below is correct in MPEg-7 DDL  -->
<complexType name="SemanticRelationshipMajorTypeBase" base="string">
      <!--The following list is a place holder, likely to be slightly extended
according to current state-of-the-art solutions --!>
      <simpleType name="SemanticRelationshipMajorType">
            <enumeration value="is_a"/>
      <!-- to be interpreted as "is an instance of" -->

            <enumeration value="is_composed_of"/>
            <enumeration value="is_next_to"/>
            <enumeration value="is_involved_in">
            <enumeration value="is_similar_to"/>
      </simpleType>
      <simpleType name="SourceParticipationType" >
            <enumeration value="agent" />
            <enumeration value="patient" />
            <enumeration value="recipient"/>
            <!-- Other types to be provided if necessary --!>
      </simpleType>
</complexType>


<!-- ################################################ -->
<!-- Definition of Semantic Relationship Node DS      -->
<!-- ################################################ -->

<complexType name="SemanticRelationshipNode">
      <element     name="SemanticRelationship"     type="SemanticRelationship"
minOccurs="1" maxOccurs="1"/>
      <choice minOccurs="1" maxOccurs="unbounded">
            <element                          name="SemanticRelationshipNode"
type="SemanticRelationshipNode"/>
            <element name="SemanticNode" type="SemanticNode"/>
      </choice>
</complexType>


<!-- ################################################ -->
<!-- Definition of Semantic Node DS                   -->
<!-- ################################################ -->

<complexType name="SemanticNode">
      <sequence minOccurs="1" maxOccurs="unbounded">
            <choice minOccurs="0" maxOccurs="unbounded">
                  <element                   name="ReferenceToSemanticElement"
type="ReferenceToSemanticElement"/>
                  <element name="SemanticNode" type="SemanticNode"/>
            </choice>
            <element                            name="SemanticRelationshipNode"
type="SemanticRelationshipNode" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="id" type="ID" use="optional"/>
</complexType>
```

<-- Editor's note: There seems to be an inconsistency to define the source and the destination of any edge in the graph, connecting 2 semantic nodes. To be fixed! -->

6.3.1.3 Description Scheme Semantics

Semantics of the SemanticGraph DS.

Name	*Definition*
SemanticGraph	Description scheme that specifies relationships among different semantic entities. Its nodes are allowed to be the following DS: Objects, Events, States, Concepts, and SemanticGraphs. It is nearly a subDS of the entity relationship graph in the MDS WD. The edges of a SemanticGraph are semantic relationships between the node entities. They are specified with several attributes. There is an attribute that specifies the semantic role of the source node, e.g. agent, patient, recipient, etc. There is an enumeration of major relationship types, derived from for example WordNet. These major types include is_a, is similar to, … and so forth. These may be refined by specifying a refinement. The refinement is a Term D, and is intended to point into a list of controlledTerms, which may include, or be represented by, a list of Concept DS. The exact specification of the syntax of SemanticGraph is dependent on its near parent, ER Graph DS in the WD. That graph structure is currently under review. Clearly specify the elements it can relate and the relationships it can express among them.
Id	Identifier of an instance of the SemanticGraph DS.
Type	String which specifies the type of the semantic-relationship graph (e.g. "temporal"). For a possible taxonomy for semantic relationship types see section 8.2.2.2.
Name	String which specifies the semantics of the semantic-relationship graph. Specific types of graphs can be assigned a name.
SemanticRelationshipNode	Node defining the set of semantic relationships involved in the graph.
SemanticNode	Node defining the set of semantic involved in the relation graph.

Semantics of the SemanticRelationship DS.

Name	*Definition*
SemanticRelationship	Definition of a semantic relationship.
PropertyType	The kind of relationship property. It specifies two possible values for the relationship property: "AscendingOrder" and "DescendingOrder". When a relationship describes an order of entities based on a specific criterion, the order criterion will be included in the attribute name of the Relationship DS. The attribute property (of type PropertyType) of the Relationship DS will specify that the relationship is an order in ascending or descending form. Need to specify the role of a semantic element in a relationship, for example, recipient.
Id	Identifier for an instance of the Relationship DS
Name	String which describes the semantics of the relationship (e.g. "before"). This attribute is mandatory.
Level	String which specifies the degree of generality of the information described by the relationship (e.g. "generic" and "specific").
Degree	Integer which specifies the degree of a relationship. For example, degree will equal 2 for a binary relationship. The degree of a relationship specifies the number of vertices of a relationship, i.e. the number of entity nodes participating in the relationship. By default, in a description, the degree of a relationship is given by the number of entity nodes to which a relationship is related.
Direction	Attribute which specifies the direction of a relationship.
Property	Attribute witch specifies the property type for ordered relationships. In this case, the name of the relationship defines the order criterion.
SemanticRelationshipType	Definition of the relationship type as a base type and a refinement.

Semantics of the SemanticRelationshipType and SemanticRelationshipTypeBase DSs.

Name	*Definition*
SemanticRelationshipMajorTypeBase	Definition of major relationship types.
Refinement	Attribute further refining the type of a relationship.
SourceParticipationType	Definition to be provided
SemanticRelationshipType	Definition of relationship types as a base type and a refinement.

Semantics of the SemanticRelationshipNode DS.

Name	*Definition*

Name	*Definition*
SemanticRelationshipNode	Relationship of any degree (e.g. binary) and cardinality (e.g. 1:N) among the set of semantic elements. In a description, a semantic relationship node contains one semantic relationship and can also contain an arbitrary number of semantic nodes and other semantic relationship nodes to allow the definition of efficient and nested graphs of relationships among semantic.
SemanticRelationship	Definition of a semantic relationship.
SemanticNode	SemanticNode grouping the set of references to semantic (or SemanticNode) that are involved in the semantic relationship node.
SemanticRelationshipNode	Node that groups the semantic relationship nodes involved in the semantic relationship node.

Semantics of the SemanticNode DS.

Name	*Definition*
SemanticNode	The semantic node includes the set of references to semantic elements that participate in one vertex of a relationship. The number of vertices of a relationship is given by the relationship's degree. In a description, a semantic node can also contain an arbitrary number of semantic relationship nodes and other semantic nodes to allow the definition of efficient and nested graphs of relationships among semantic elements.
Id	Identifier of an instance of the SemanticNode DS.
ReferenceToSemanticElement	Reference to a semantic element that participates in the relationship. Semantic elements can be concepts, concept events, concept objects, objects, events, semantic graphs, and states at the moment.
SemanticNode	SemanticNode grouping the set of references to semantic elements (or SemanticNode) that participate in the relationship.
SemanticRelationshipNode	Relationship between the semantic elements.

6.3.1.4 Description Extraction

Manual instantiation is possible.

6.3.1.5 Description Example

Figure 1 shows the key frame of a video shot capturing a goal in a soccer game. The description of this video shot at the semantic level could be as follows. The entire video shot could be described by an event with id “Goal-ev” in the Event DS. The kick and not-catch events in the video shot could be represented by two events with ids “Kick-ev” and “Not-Catch-ev”, respectively. The video objects corresponding to the goal, the forward, and the goalkeeper could be described by objects in the Object DS with ids “Goal-ob”, “Forward-ob”, and “Goalkeeper-ob”, respectively. The forward and the goalkeeper could be described as instantiations of the concept of person (ConceptObject DS). The person concept is composed of a head, two arms, two legs, and a body concepts objects where head is on top of the body, the arms are on the side of the body, and the legs are below the body.

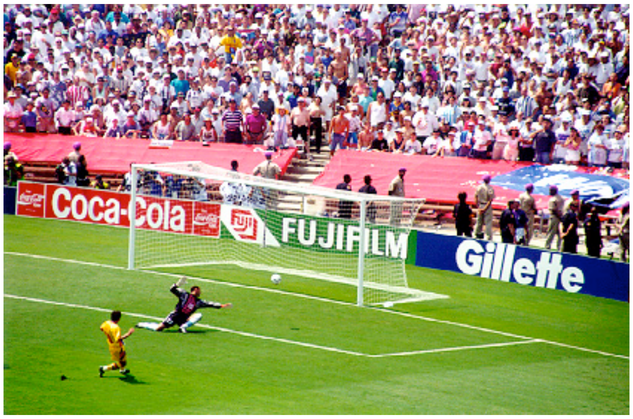


Figure 1: Key frame of a video shot depicting a soccer goal.

Once these events, objects, and concepts are defined using the Event, Object DS, and the ConceptObject DSs, the Semantic Graph DS could be used to describe relationships among them. Some examples of semantic relations among these elements are as follows. The “Kick-ev” event is composed of the “Forward-ob” and the “Ball-ob” objects. Similarly, the “Not-Catch-ev” event is composed of the “Ball-ob” and the “Goalkeeper-ob” objects. The “Forward-ob” object is related to the “Ball-ob” and the “Goal-ob” objects by an action relationship, “Kicks”. The “Goalkeeper-ob” object is related to the “Ball-ob” object by an action relationship, “Does not catch”. The “Ball-ob” object is related to the “Goal-ob” object by an action relationship, “Enters”.

The semantic description for Figure 1 is expressed in XML below.

```
<SemanticGraph>
      <SemanticNode>
            <ReferenceToEvent idref=”Kick-ev”/>
            <SemanticRelationshipNode>
                  <SemanticRelationship name=”Composed of”>
                        <SemanticRelationshipType … />
                  </SemanticRelationship>
                  <SemanticNode>
                        <ReferenceToObject idref=”Forward-ob”/>
                        <ReferenceToObject idref=“Ball-ob”/>
                  </SemanticNode>
            </SemanticRelationshipNode>
      </SemanticNode>
      <SemanticNode>
            <ReferenceToEvent idref=”Not-Catch-ev”/>
            <SemanticRelationshipNode>
                  <SemanticRelationship  name=”Composed of”>
                        <SemanticRelationshipType … />
                  </SemanticRelationship>
                  <SemanticNode>
                        <ReferenceToObject idref=”Goalkeeper-ob”/>
                        <ReferenceToObject idref=“Ball-ob”/>
                  </SemanticNode>
            </SemanticRelationshipNode>
      </SemanticNode>
      <SemanticNode>
            <ReferenceToEvent idref=”Forward-ob”/>
            <SemanticRelationshipNode>
                  <SemanticRelationship  name=”Kicks”>
                        <SemanticRelationshipType … />
                  </SemanticRelationship>
                  <SemanticNode>
                        <ReferenceToObject idref=“Ball-ob”/>
                  </SemanticNode>
                  <SemanticNode>
                        <ReferenceToObject idref=“Goal-ob”/>
                  </SemanticNode>
            </SemanticRelationshipNode>
      </SemanticNode>
      <SemanticNode>
            <ReferenceToEvent idref=”Goalkeeper-ob”/>
            <SemanticRelationshipNode>
                  <SemanticRelationship  name=”Does not catch”>
                        <SemanticRelationshipType … />
                  </SemanticRelationship>
                  <SemanticNode>
                        <ReferenceToObject idref=“Ball-ob”/>
                  </SemanticNode>
            </SemanticRelationshipNode>
      </SemanticNode>
      <SemanticNode>
            <ReferenceToEvent idref=”Ball-ob”/>
            <SemanticRelationshipNode>
                  <SemanticRelationship name=”Enters”>
                        <SemanticRelationshipType … />
                  </SemanticRelationship>
                  <SemanticNode>
                        <ReferenceToObject idref=“Goal-ob”/>
                  </SemanticNode>
            </SemanticRelationshipNode>
      </SemanticNode>
</SemanticGraph>
```

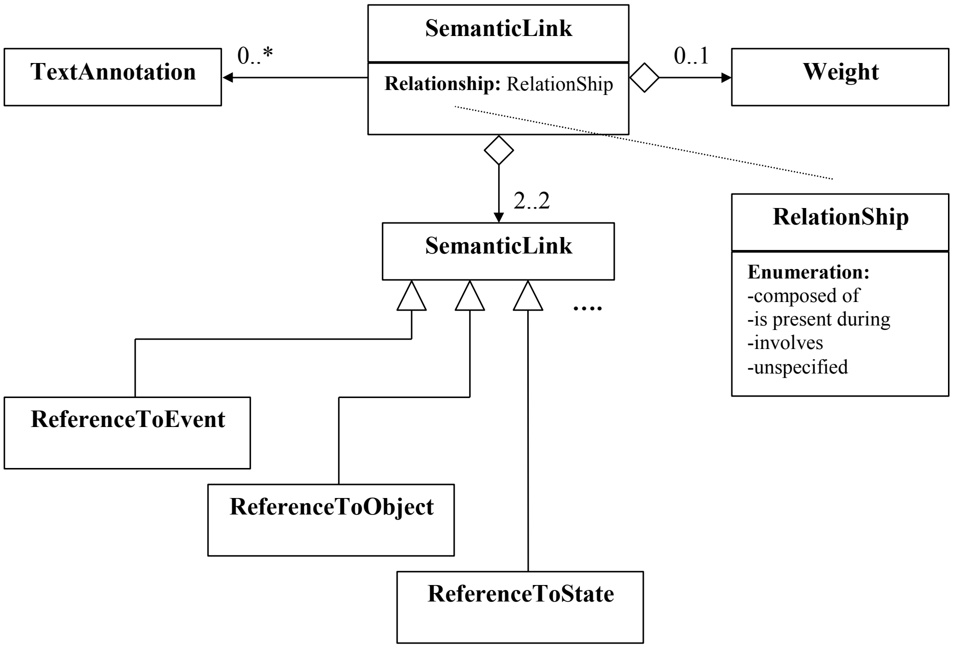
6.3.1.6 Description Use

Retrieval and browsing are possible uses of semantic graph descriptions.

6.3.2 SemanticLink DS

The SemanticLink DS defines the type of linked components and the relationships (with possible attached weights) between components of the Semantic DS.

6.3.2.1 Description Scheme Pseudo UML (Relationships to be reviewed)



6.3.2.2 Description Scheme Syntax

```
<!-- ############################################### -->
<!-- Definition of Semantic Link DS                  -->
<!-- ############################################### -->

<complexType name="SemanticLink">
      <element name="Weight" type="Weight" minOccurs="0" maxOccurs="1"/>
      <element   name="TextAnnotation"   type="TextAnnotation"   minOccurs="0"
maxOccurs="unbounded"/>
      <element name="ReferenceToSemanticElement"
            type="ReferenceToSemanticElement" minOccurs="2" maxOccurs="2"/>
      <attribute name="relationship" type="RelationShip"/>
      <attribute name="SourceParticipation" type="SourceParticipationType"/>
</complexType>

<-- Editor's note: There seems to be an inconsistency to define who is the
source and who is the destination of the link connecting 2 semantic elements.
To be fixed by specifying which is the link source! -->

<!–-  Note: The name of the DS below should be changed to something less
generic. -->

<simpleType name="RelationShip" basetype="string">
      <enumeration value="is_a"/>
            <!-- to be interpreted as "is_an_instance_of" -->

      <enumeration value="is_composed_of"/>
      <enumeration value="is_involved_in"/>
      <enumeration value="is_similar_to"/>
      <enumeration value="unspecified"/>
<!-- other to be added if necessary -->
<!-- possible to define relationship through refinement of -->
<!-- unspecified by TextAnnotation D -->
</simpleType>

      <simpleType name="SourceParticipationType" >
            <enumeration value="agent"/>
            <enumeration value="patient"/>
            <enumeration value="recipient"/>
            <!-- Other types to be provided if necessary --!>
      </simpleType>
```

6.3.2.3 Description Scheme Semantics

Name	*Definition*
SemanticLink	Description scheme that defines the type of linked components and the relationships (with possible attached weights) between components of the Semantic DS. Clearly specify the elements it can relate and the relationships it can express among them.
RelationShip	Definition is needed
relationship	Definition is needed Need to specify the role of a semantic element in a relationship, for example, recipient.
SourceParticipation	Definition is needed
SourceParticipationType	Definition is needed
Weight	Description scheme that enables the description of relative ordering, confidence, and perceptual significance of the different elements in the description such as within this DS. Clarification is needed.
TextAnnotation	Free text annotation describing the semantic link. Definition is needed.
ReferenceToSemanticElement	Reference to semantic element participating in the relationship.

6.3.2.4 Description Extraction

6.3.2.5 Description Example

6.3.2.6 Description Use

6.3.3 SegmentSemanticLink DS

The SegmentSemanticLink DS defines the type of linked components and the relationships (with possible attached weights) between components of the Semantic DS and Segment DSs.

6.3.3.1 Description Scheme Syntax

```
<!-- ################################################ -->
<!-- Definition of Segment Semantic Link DS           -->
<!-- ################################################ -->

<complexType     name="SegmentSemanticLink"     base=”SegmentSemanticLinkBase”
derivedBy=”extension”>
      <attribute name="refinement" type="controlledTerm"/>
</complexType >

<!—- Question: Some clarification on the definition of the
SegmentSemanticLinkBase is needed. Does this element contain a string and
multiple Link elements? I do not think XML-Schema allows to define an
enumerated type for the content of a mixed element. “Link” is not defined
anywhere. Is href supposed to reference an element in the same file
description or a different file description? In the former case, type IDREF
could be used. -->

<complexType name="SegmentSemanticLinkBase" base=”string”>
      <enumeration value=”represents”/>
      <enumeration value=”is represented by”/>

      <element name="Link" type=”?” minOccurs="0" maxOccurs=“unbounded”/>
      <attribute name="id" type="ID"/>
      <attribute name="href" type="uriReference"/>
</complexType>
```

6.3.3.2 Description Scheme Semantics

Name	*Definition*
SegmentSemanticLink	Description scheme that defines the type of linked components and the relationships (with possible attached weights) between components of the Semantic DS and Segment DSs. Which components? Does it only define types of relationships, i.e. rules of usage of relationships?
Finish definition	

6.3.3.3 Description Extraction

6.3.3.4 Description Example

6.3.3.5 Description Use

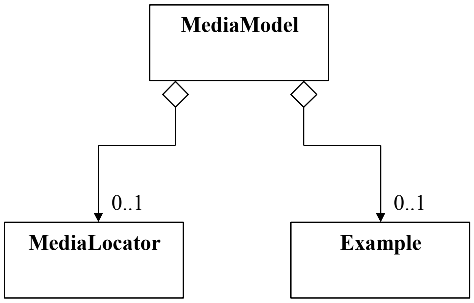
7 Organization of Content

7.1 Analytical Model

7.1.1 MediaModel DS

The MediaModel DS specifies an abstract description of a semantic entity of type Event DS, Object DS, ConceptEvent DS, and Concept Object DS.

7.1.1.1 Description Scheme Pseudo UML (needs update)



7.1.1.2 Description Scheme Syntax (needs revision!!! → it is an AnalyticModel)

```
<!-- ################################################ -->
<!-- Definition of Media Model DS                     -->
<!-- ################################################ -->

<!--  Note: There is a Model DS, the Cluster DS, which includes Media Locators
to occurrences of semantic concepts. Is that the purpose of MediaLocator in
MediaModel and MediaAnnotation? If so, MediaModel should contain Examples and
Cluster DS. Other probability models in the WD may also be relevant to
MediaModel DS such as the Probability Model DS.  -->

<complexType name="MediaModel">
      <element    name="MediaLocator"    type="MediaLocator"    minOccurs="0"
maxOccurs="unbounded"/>
      <element name="Examples" type="Examples" minOccurs="0" maxOccurs="1"/>
</complexType>
```

7.1.1.3 Description Scheme Semantics

Name	*Definition*
`MediaModel`	Description scheme that specifies an abstract description of an

Name	*Definition*
	object, an event, a concept object, or a concept event by one or more media locators to physical instances of the semantic element, and by a set of value specifications associated with these instances.
`MediaLocator`	A locator to the physical instances of semantic element. Clarify relationship with Cluster DS.
`Examples`	A set of descriptor values attached to the physical instance of a semantic element. Needs clarification

7.1.1.4 Description Extraction

7.1.1.5 Description Example

7.1.1.6 Description Use

Annex B: Description Examples of Semantic DS

1 Portuguese News

Content:

- CD Number: 14 (MPEG-7 Content Set)
- File name: jornaldanoite1.mpg
- Start time: 31'40"
- End time: 36'00"

Description generation:

- Natural language description (consider that we don't understand portugese language)

Description:

- The sequence contains a dialog (interview) between the speaker and a psychiatrist about drugs.

Set of supported queries

- All about drug
- Retrieve all dialogs in this program
- Retrieve all dialogs about drug.
- Retrieve all the occurrences of the speaker inside the program
- Retrieve all the occurrences of the psychiatrist inside the program
- How many people are present in a dialog

```
<SemanticDescription id="1" name="Brescia experiment">
      <Concept id="1" name="Dialogue about drug">
            <TextAnnotation>This represents the concept of dialog about drug
            </TextAnnotation>
            <Concept id="2" name="Dialogue">
                  <TextAnnotation>Two people speak among them
                  </TextAnnotation>
                  <ConceptObject id="1" name="Speaker"/>
                  <ConceptObject id="2" name="Speaker"/>
                  <ConceptEvent id="1" name="To speak about…"/>
            </Concept>
            <Concept id="3" name="Drug"/>
      </Concept>
      <ConceptObject id="3" name="Person"/>
      <SemanticLink>
            <Relationship>is a</Relationship>
            <ReferenceToConceptObject idref="3"/>
            <ReferenceToConceptObject idref="1"/>
      </SemanticLink>
```

```
<SemanticLink>
      <Relationship>is a</Relationship>
      <ReferenceToConceptObject idref="3"/>
      <ReferenceToConceptObject idref="2"/>
</SemanticLink>
<SemanticLink>
      <Relationship>is a</Relationship>
      <ReferenceToConceptObject idref="3"/>
      <ReferenceToConceptObject idref="1"/>
</SemanticLink>
<SemanticLink>
      <Relationship>is present during</Relationship>
      <ReferenceToObject idref="1"/>
      <ReferenceToEvent idref="1"/>
</SemanticLink>
<SemanticLink>
      <Relationship>is present during</Relationship>
      <ReferenceToObject idref="2"/>
      <ReferenceToEvent idref="1"/>
</SemanticLink>
<SemanticLink>
      <Relationship>is a</Relationship>
      <ReferenceToEvent idref="1"/>
      <ReferenceToConcept idref="1"/>
</SemanticLink>
<Object id="1" name="Pedro">
      <TextAnnotation>A speaker of RTV portugese channel
      </TextAnnotation>
      <MediaAnnotation>
            <MediaLocator>
            <MediaURL>file:///D:/portugese/jornaldanoite1.mpg
            </MediaURL>
                     <MediaTime>
                             <RelTime> <s>234</s> </RelTime>
                             <Duration> <s>65</s> </Duration>
                     </MediaTime>
            </MediaLocator>
            <MediaLocator>
            <MediaURL> file:///D:/portugese/jornaldanoite1.mpg
            </MediaURL>
                     <MediaTime>
                             <RelTime> <s>456</s> </RelTime>
                             <Duration> <s>34</s> </Duration>
                     </MediaTime>
            </MediaLocator>
            <MediaLocator>
            <MediaURL> file:///D:/portugese/jornaldanoite1.mpg
            </MediaURL>
                     <MediaTime>
                             <RelTime> <s>1900</s> </RelTime>
                             <Duration> <s>260</s> </Duration>
                     </MediaTime>
            </MediaLocator>
      </MediaAnnotation>
</Object>
<Object id="2" name="Psychiatrist">
      <TextAnnotation>Serginho Beninos, a psychiatrist of St. Onofrio
      Hospital, Porto </TextAnnotation>
```

```
            <MediaAnnotation>
                  <MediaLocator>
                  <MediaURL> file:///D:/portugese/jornaldanoite1.mpg
                  </MediaURL>
                           <MediaTime>
                                   <RealTime> <s>1900</s> </RealTime>
                                   <Duration> <s>260</s> </Duration>
                           </MediaTime>
                  </MediaLocator>
            </MediaAnnotation>
      </Object>
      <Event id="3" name="Interview">
            <TextAnnotation>This is an interview about drug. It is contained
            inside the news program 'Jornal da noite' of
            04/05/1997"</TextAnnotation>
            <MediaAnnotation>
                  <MediaLocator>
                  <MediaURL> file:///D:/portugese/jornaldanoite1.mpg
            </MediaURL>
                        <MediaTime>
                              <RelTime> <s>1900</s> </RelTime>
                              <Duration> <s>260</s> </Duration>
                        </MediaTime>
                  </MediaLocator>
            </MediaAnnotation>
            <!-- Place DS is used instead of SLocation because    -->
            <!-- Place DS is more useful in this case.            -->
            <!-- Harmonization between SLocation and Place DS     -->
            <!-- should be made. -->
            <Place>
                  <PlaceName xml:lang=’en’>Lisboa</PlaceName>
                  <Country>PT</Country>
                  <Planet>Earth</Planet>
                  <GPSCoordinates GPSsystem=’’>XXXX</GPSCoordinates>
                  <PostingIdentifier>E-28040</PostingIdentifier>
                  <PostalAddress>
                        Porto TV, 19 Rua Preta Da Montagem, Lisboa
                  </PostalAddress>
                  <InternalCoordinates>C-306</InternalCoordinates>
            </Place>
            <STime>
                  <TimeGrouping>
                  <Time>
                        <TimePoint TZD="+100">
                              <Y>1997</Y>
                              <M>04</M>
                              <D>05</D>
                              <h>17</h>
                              <m>30</m>
                        </TimePoint>
                        <Duration>
                              <m>30</m>
                        </Duration>
                  </Time>
                  </TimeGrouping>
            </STime>
      </Event>
</SemanticDescription>
```

1.1 Photographs

Content:

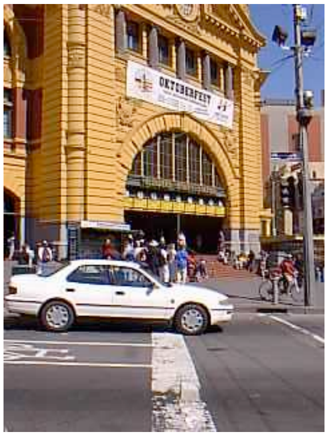
- Melbourne Photo Database
- http://www.cre.canon.co.uk/mpeg7/melbourne_photo_database.htm
- image001.jpg

Description generation:

- Natural language description available from images
- Translation to Semantic DS adding additional temporal and spatial information for objects

Description:

- “Flinders Street station across the road from the tram stop”



```
<!--  Description 1  -->
<!--
“Flinders Street station” is represented as an Object. “across the road from
the tram stop” is included in TextAnnotation of Slocation in the event “is”.
The Event and the Object are linked using a Semantic Graph.
-->

<Semantic>
      <SemanticDescription>
            <Object name=”Flinders Street station” id=”object1”>
                  <!-– Flinders Street has an associated inherent location;
                       however, no direct mechanism exits in Object DS
                       to describe this. The only way to associate Object
                       with Location is to use a State of an Event. -->
                  <TextAnnotation>
                        Flinders Street station is a station in Melbourne
                  </TextAnnotation>
            </Object>
            <Event name=”is” id=”event1”>
                  <SLocation>
                        <TextAnnotation>Across the tram stop</TextAnnotation>
                  </SLocation>
            </Event>
            <SemanticGraph>
                  <SemanticNode>
                        <ReferenceToObject idref=”object1”/>
                        <SemanticRelationshipNode>
                              <SemanticRelationship name=”is involved in”>
                                    <SemanticRelationshipType … />
                              </SemanticRelationship>
                              <SemanticNode>
                                    <ReferenceToObject idref=”event1”/>
                              </SemanticNode>
                        </SemanticRelationshipNode>
                  </SemanticNode>
            </SemanticGraph>
      </SemanticDescription>
</Semantic>



<!--  Description 2  -->
<!--
“Flinders Street station” and “tram stop” are represented by Objects. “across
the road” is represented by a relation among the two objects in a semantic
graph. The location information of “Flinders Street station” is described
State. Location and time information can not be included together in the same
sate; therefore, no time information is specified. The link between State and
Object is established using another relation in the semantic graph.
-->

<Semantic>
      <SemanticDescription>
            <Object name=”Flinders Street station” id=”object1”>
                  <!—- Flinders Street has an associated inherent location;
                       however, no direct mechanism exits in Object DS
                       to describe this. The only way to associate Object
                       with Location is to use a State of an Event. -->
                  <TextAnnotation>
                        Flinders Street station is a station in Melbourne
                  </TextAnnotation>
            </Object>
            <Object name=”tram stop” id=”object2”>
                  <!—- Same problem of inherent location as with
                       Flinders Street station -->
            </Object>
            <State name=”location of Flinders Street Station” id=”state1”>
                  <Coordinate>
                        <!--  This has not been specified yet  -->
                  </Coordinate>
            </State>
            <SemanticGraph>
                  <SemanticNode>
                        <ReferenceToObject idref=”object1”/>
                        <SemanticRelationshipNode>
```

```
                              <SemanticRelationship name=”across the road”>
                                    <SemanticRelationshipType … />
                              </SemanticRelationship>
                              <SemanticNode>
                                    <ReferenceToObject idref=”object2”/>
                              </SemanticNode>
                        </SemanticRelationshipNode>
                  </SemanticNode>
                  <SemanticNode>
                        <ReferenceToObject idref=”object1”/>
                        <SemanticRelationshipNode>
                              <SemanticRelationship name=”state”>
                                    <SemanticRelationshipType … />
                              </SemanticRelationship>
                              <SemanticNode>
                                    <ReferenceToObject idref=”state1”/>
                              </SemanticNode>
                        </SemanticRelationshipNode>
                  </SemanticNode>
            </SemanticGraph>
      </SemanticDescription>
</Semantic>



<!--  Description 3  -->
<!--
“Flinders Street station” and “tram stop” are represented by Objects.
“across the road” is represented using an event. The time information of
“Flinders Street station” is associated to the event “across the road”.
However, the location information of “Flinders Street station” can not be
associated with the event: the two objects involved in the event have indeed
different locations. The event would have two different locations but there is
no mechanism to link “Flinders Street station” with only one of them. The way
chosen to describe the location of “Flinders Street station” is to define a
new event “Is” with associated location information. The links between the
objects and the events are described using the SemanticLink.
-->

<Semantic>
      <SemanticDescription>
            <Object name=”Flinders Street station” id=”object1”>
                  <!—- Flinders Street has an associated inherent location;
                       however, no direct mechanism exits in Object DS
                       to describe this. The only way to associate Object
                       with Location is to use a State of an Event. -->
                  <TextAnnotation>
                        Flinders Street station is a station in Melbourne
                  </TextAnnotation>
            </Object>
            <Object name=”tram stop” id=”object2”>
                  <!—- Same problem of inherent location as with
                       Flinders Street station -->
            </Object>
            <event name=”across the road” id=”event1”>
                  <STime>
                        <!-- Semantic time of object 1 and object 2 -->
            </STime>
        </event>
        <event name=”is” id=”event2”>
            <SLocation>
                <!--  Semantic location of object1  -->
            </SLocation>
        </event>
        <SemanticLink>
            <Relationship>is involved in</Relationship>
            <ReferenceToObject idref=”object1”/>
            <ReferenceToObject idref=”event1”/>
        </SemanticLink>
        <SemanticLink>
            <Relationship>is involved in</Relationship>
            <ReferenceToObject idref=”object2”/>
            <ReferenceToObject idref=”event1”/>
        </SemanticLink>
        <SemanticLink>
            <Relationship>is involved in</Relationship>
            <ReferenceToObject idref=”object1”/>
            <ReferenceToObject idref=”event2”/>
        </SemanticLink>
    </SemanticDescription>
</Semantic>
```

Content:

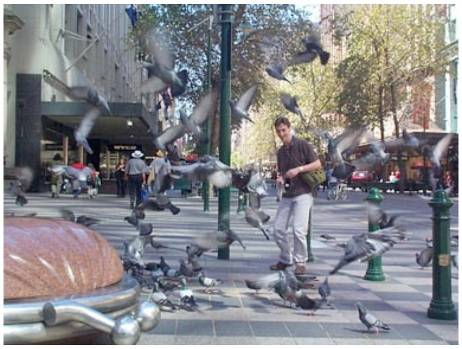
- Melbourne Photo Database
- http://www.cre.canon.co.uk/mpeg7/melbourne_photo_database.htm
- DCP01960.JPG

Description generation:

- Description of the image using Semantic DS directly

Description:

- Phil scares pigeons near the statue of a purse in Melbourne in the afternoon of October 15th, 1999.



```
<Semantic>
      <SemanticDescription>
            <Concept id="concept1" name="person"/>
            <Concept id="concept2" name="animal"/>

            <Object name="Phil" id="object1">
                  <!--  Object description  -->
            </Object>
            <Object name="pigeons" id="object2">
                  <!--  Object description  -->
                  <!--  How could pigeons be described in terms of
                        one pigeon?  -->
            </Object>
            <event name="across the road" id="event1">
                  <Stime>
                        <Time>
                  </STime>
            </event>
            <event name="scares" id="event1">
                  <STime>
                        <TimeGrouping>
                              <!--  This is not conformant with
                                    the specification of Stime, however
                                    Stime should support this type
                                    of semantic descriptions
                                    of time -->
                              <SemanticRelativeTime
                                    TimeOrigin="October 15th, 1999"
                                    TimeInstant="afternoon"/>
                        </TimeGrouping>
                  </STime>
                  <!--  What if we wanted to desribe Melbourne as
                        san object?  -->
                  <Slocation name="Melbourne"/>
            </event>

            <SemanticLink>
                  <Relationship>is a</Relationship>
                  <ReferenceToObject idref="object1"/>
                  <ReferenceToObject idref="concept1"/>
            </SemanticLink>
            <SemanticLink>
                  <Relationship>is a</Relationship>
                  <ReferenceToObject idref="object2"/>
                  <ReferenceToObject idref="concept2"/>
            </SemanticLink>
            <SemanticLink sourceParticipation="agent">
                  <Relationship>is involved in</Relationship>
                  <ReferenceToObject idref="object1"/>
                  <ReferenceToObject idref="event1"/>
            </SemanticLink>
            <SemanticLink sourceParticipation="recipient">
                  <Relationship>is involved in</Relationship>
                  <ReferenceToObject idref="object2"/>
                  <ReferenceToObject idref=”event1”/>
            </SemanticLink>

      </SemanticDescription>
</Semantic>
```