Bootstrapping Ontology Evolution with Multimedia Information Extraction

D4.1: Methodology and Architecture for Multimedia Ontology Evolution

Version 1.0 Final

S. Castano[1], K. Dalakleidi[4], S. Dasiopoulou[4], S. Espinosa[2], A. Ferrara[1], G. N. Hess[1], V. Karkaletsis[3], A. Kaya[2], S. Melzer[2], R. Möller[2], S. Montanelli[1], G. Petasis[3]

[1]Università degli Studi di Milano (UNIMI)
[2]Hamburg University of Technology (TUHH)
[3]National Centre for Scientific Research "Demokritos" (NCSR)
[4]Centre for Research and Technology Hellas (CERTH)

Project ref.no.:	*FP6-027538*
Workpackage:	*WP4*
WP / Task responsible:	*UNIMI*
Other contributors:	*TUHH, NCSR, CERTH*
Quality Assurance:	*George Paliouras, Sergios Petridis*
Document type:	*Report*
Security (distribution level):	*Restricted*
Status & version:	*Final Rel. 1.0*
Contractual date of delivery:	*M08*
Actual date of delivery:	
Deliverable number:	*D4.1*
Deliverable name:	*Methodology and Architecture for Multimedia Ontology Evolution*
Date:	*19 December 06*
Number of pages:	*88*
EC Project Officer:	*Johan Hagman*
Keywords:	*Ontology Evolution, Population, Enrichment, Coordination, Matching, Reasoning*
Abstract (for dissemination):	*Ontology evolution is generally defined as the timely adaptation of an ontology to changing requirements and the consistent propagation of changes to dependent artifacts. In BOEMIE, a novel, pattern-driven methodology for multimedia ontology evolution is defined to control the bootstrapping process. The methodology uses the information extracted from the multimedia content in order to populate, enrich and coordinate the domain ontology and to provide the necessary semantic information back to the information extraction modules. This document describes the methodology for ontology evolution adopted in BOEMIE together with the associated open architecture.*

Contents

EXECUTIVE SUMMARY **6**

1 INTRODUCTION **9**

2 THE MULTIMEDIA SEMANTIC MODEL AND THE BOOTSTRAPPING PROCESS OF BOEMIE **10**
2.1 The Multimedia Semantic Model of BOEMIE 10
2.1.1 The Athletics Events Ontology 11
2.1.2 The Geographic Information Ontology 12
2.1.3 Modality Element Ontology 12
2.1.4 Spatio-temporal Relations Ontology 13
2.1.5 Multimedia Content Ontology 13
2.1.6 Multimedia Descriptor Ontology 14
2.1.7 Core ontology 14
2.2 The BOEMIE bootstrapping process 14

3 METHODOLOGY FOR ONTOLOGY EVOLUTION **16**
3.1 Design principles 16
3.2 Overview of the methodology 17
3.3 Input 19
3.4 Output 19
3.5 Evolution patterns 19
3.6 Activities and tasks 21
3.7 Phases 23
3.8 Supporting services 23
3.9 Roles: the ontology expert 23

4 LOGICAL FOUNDATIONS AND REASONING SERVICES **25**
4.1 Expressive Description Logics: Syntax and Semantics 25
4.1.1 Very Expressive Description Logics 26
4.1.2 Other Decidable Fragments of First-Order Logic 27
4.1.3 Rules 27
4.2 Standard Inference Services 27
4.3 Retrieval Inference Services 28
4.4 Nonstandard Inference Services 29
4.5 Examples of interpretation in BOEMIE 30

5 ONTOLOGY POPULATION **35**
5.1 Single concept explanation 36
5.1.1 Instance matching 36
5.1.2 Instance grouping 39
5.1.3 ABox validation 39
5.1.4 ABox assimilation 39
5.2 Multiple concept explanation 39
5.2.1 Instance matching and ABox refinement 39
5.2.2 Example: pole vault or high jump 40
5.3 Measurable objectives 45

6 ONTOLOGY ENRICHMENT **49**
6.1 Missing concept with metadata explanation 50
6.1.1 Concept learning 50
6.1.2 Concept enhancement 52
6.1.3 Concept definition and validation 55
6.1.4 Concept assimilation 57
6.2 Missing concept without metadata explanation 57

6.2.1 Concept learning . . . 58
6.2.2 Concept definition . . . 58
6.2.3 Concept validation . . . 58
6.2.4 Concept assimilation . . . 58
6.3 Example: learning a new concept . . . 58
6.4 Measurable objectives . . . 61

7 ONTOLOGY COORDINATION **62**
7.1 Ontology versioning and management . . . 62
7.2 Ontology alignment . . . 63
7.2.1 Definition of new mappings . . . 63
7.2.2 Maintenance of existing mappings . . . 65
7.3 Example: ontology versioning and alignment . . . 66
7.4 Measurable objectives . . . 69

8 ARCHITECTURE FOR ONTOLOGY EVOLUTION **70**
8.1 Evolution tools and interactions . . . 70
8.1.1 Population and enrichment tool . . . 70
8.1.2 Coordination tool . . . 71
8.2 Evolution services . . . 72
8.2.1 Reasoning service . . . 72
8.2.2 Standards for ontology specification . . . 73
8.2.3 Standards for Invoking Reasoning Services . . . 73
8.2.4 Matching service . . . 74

9 INNOVATION **76**
9.1 Population and enrichment . . . 76
9.2 Coordination and matching . . . 77
9.3 Reasoning . . . 77

10 EVALUATION **79**
10.1 Measurable objectives . . . 79
10.1.1 Population . . . 79
10.1.2 Enrichment . . . 80
10.1.3 Coordination . . . 80

11 RISK ANALYSIS **81**
11.1 Risk factors . . . 81
11.2 Risk handling . . . 81

12 CONCLUDING REMARKS **82**

REFERENCES **83**

List of Figures

1 The BOEMIE evolution methodology . . . 7
2 The BOEMIE Multimedia Semantic Model . . . 10
3 The representation of the concept Athlete in the T-Box of the Athletics Ontology taking into account text and image analysis requirements . . . 11
4 The definition of the Pole Vault in the TBox of the athletics domain . . . 12
5 A tiny example of the TBox of the Geographic Information Ontology . . . 12
6 A part of the TBox of the Modality Element Ontology . . . 13
7 Additional restrictions for $Pole\_Vault$ in the form of rules. . . . 13
8 A tiny example of the TBox of the Multimedia Content Ontology . . . 14
9 A tiny example of the TBox of the Multimedia Descriptor Ontology . . . 14
10 The BOEMIE Bootstrapping process . . . 15
11 The BOEMIE evolution methodology . . . 17
12 Syntax and Semantics of $\mathcal{ALCQHI}_{R^+}(\mathcal{D})^-$. . . . 26
13 Still image displaying a pole vault event. . . . 31
14 An ABox Γ representing the result of the image analysis phase. . . . 31
15 An tiny example TBox Σ for the athletics domain . . . 32
16 Additional restrictions for $Pole\_Vault$ and $High\_Jump$ in the form of rules. . . . 32
17 One possible solution of the abduction equation. . . . 33
18 An ABox representing the result of the image interpretation phase. . . . 33
19 Image displaying a snapshot of a high jump or pole vault (where the pole it outside the image). . . . 34
20 An ABox Γ representing the result of the analysis of the image in Figure 19. . . . 34
21 An ABox representing the first result of the image interpretation process. . . . 34
22 An ABox representing the second result of the image interpretation process. . . . 35
23 Example of mid-level concept instances and a high-level concept instance. . . . 36
24 Example of instances extracted from another ABox. . . . 37
25 An Abox already stored in the ontology. . . . 40
26 Example of a multimedia document . . . 41
27 Information extracted from an image . . . 42
28 Information extracted from a text . . . 42
29 Graphical representation of similar instances of the example . . . 45
30 Groups selected for population . . . 46
31 Population of the multimedia content ontology module . . . 47
32 Population of modality elements . . . 47
33 Population of relation ontology module . . . 47
34 Population of the domain ontology . . . 48
35 Bottom-up approach to learn new concept definitions . . . 51
36 Γ ABoxes as basis for learning new concept definitions . . . 52
37 Obtaining the LCS from MSC . . . 52
38 Example of probe queries . . . 53
39 Example of an external knowledge source (OWL representation of a Wikipedia portion) 54
40 Result of the evaluation of PQ_1 and PQ_2 . . . 55
41 Current version of the ontology . . . 56
42 The new concepts $Pole\_Vault$ and $High\_Jump$ . . . 56
43 Example of the enriched intermediate version of the ontology . . . 57
44 Definition and linking of a new visual element . . . 59
45 High-level representation of the matching process . . . 64
46 The target ontology . . . 66
47 Graphical representation of the external knowledge sources (1) . . . 67
48 Graphical representation of the external knowledge sources (2) . . . 68
49 Evolution toolkit component and interactions diagram . . . 70
50 Population and enrichment tasks . . . 71
51 Coordination tasks . . . 72

List of Tables

1 Organization of the activities and tasks with respect to the evolution patterns and phases . . . 8
2 Typical situations that occur in ontology evolution . . . 20
3 Organization of the activities and tasks with respect to the evolution patterns and phases . . . 23
6 Similar instances retrieved in the ontology . . . 44
8 Input of the enrichment activity . . . 60
9 Results of matching the target ontology against external sources . . . 67
10 Evaluation for ontology evolution . . . 79

EXECUTIVE SUMMARY

Ontology evolution is a complex activity that is required in order to ensure the consistency of an ontology when any modifications are applied. Ontology evolution becomes even more complex in the context of BOEMIE, due to the management of multimedia resources and ontologies and to the multi-modal semantic information extraction to be enabled. To this end, starting from the state of the art in the field of ontology evolution methodologies and techniques, we define an evolution methodology that takes into account the new requirements posed by multimedia information extraction and management. The BOEMIE methodology for multimedia ontology evolution determines the manner in which the interlinked ontologies of the BOEMIE semantic model will be updated, preserving their semantic consistency. At the same time, the methodology complies with the bootstrapping process, using the information extracted from the multimedia content and providing the necessary semantic information back to the information extraction modules. The ontology is enriched with new knowledge as a consequence of the extraction process and the new evolved ontology is used to further improve the extraction process, thus triggering a new boostrapping cycle.

Design principles. The design choices of the overall methodology implementing ontology evolution have been driven by the core ontology-oriented architecture of the BOEMIE project as follows.

- *Evolution support in presence of multiple ontological modules.* In order to provide the necessary background knowledge for the automated processes of semantics extraction from multimedia content and for ontology evolution, the Multimedia Semantic Model consists of several interlinked ontological modules, each of them serving a different purpose in the knowledge space structure. Maintaining such a division of functionalities enables the independent use of each ontology knowledge space, and furthermore, allows for their linking, combination and extension. This requires to develop an evolution methodology capable of managing the coordinated evolution of such interlinked ontological modules, by which the modification applied to a part of the ontology will be automatically and consistently propagated to other interlinked ontological modules, to keep them aligned and consistent each other.

- *Pattern-driven evolution.* The extraction process has the goal of understanding what is represented into a set of multimedia resources. This activity is called *interpretation*. In particular, interpretation has the goal of associating the contents retrieved in the resources with the concepts contained in the ontology. In other terms, the ontology is the background knowledge that we use to explain the resources. The expected result of the interpretation is to find a single explanation for a resource, but, if the background knowledge is not sufficient to explain a resource, it could happen that more than one possible explanation is found, or that no explanations at all are available. Finally, it is also possible that neither the explanation of the media objects contained in a resource is available. These different results of the interpretation process lead to different situations in ontology evolution. These different situations are managed by means of four *evolution patterns*, namely:

  - P1: Single concept explanation.
  - P2: Multiple concept explanation.
  - P3: Missing concept with metadata explanation.
  - P4: Missing concept without metadata explanation.

- *Open system approach.* When background ontology knowledge is not sufficient to interpret new incoming instances, new knowledge, such as a new concept, has to be added to the ontology. Concept definition is usually a manual, human-intensive activity which requires the ontology expert to find the most appropriate specification of the missing concept. In BOEMIE, we support a knowledge discovery approach typical of open networked systems for enriching the local knowledge of a given node. Thus, we design the evolution methodology to provide a concept discovery functionality which comes into play for acquiring new knowledge by probing external knowledge sources for the same domain, looking for candidate concepts semantically matching the incoming instance. This way, the ontology knowledge space is open towards external knowledge sources, which can provide useful knowledge for evolving the ontology adequately.

- *Minimization of human involvement.* It has to be noted a completely automated evolution process is not really feasible neither recommended, due to the intrinsically interactive nature of the evolution process, which eventually requires a qualified supervision of the ontology expert. However, it is commonly recognized that a key issue for an evolution methodology is the minimization of human involvement, especially for the labor-intensive manual activities related to new concept identification and definition. To this end, we design a methodology providing a set of learning, matching, and reasoning techniques that offer support in the various evolution activities, to allow the ontology expert to refine proposed working knowledge (e.g., discovered candidate concepts) and/or to validate/choose among proposed alternative choices, thus limiting the human involvement as much as possible.

Overview of the methodology and activities. The ontology evolution methodology of BOEMIE is shown in Figure 1.

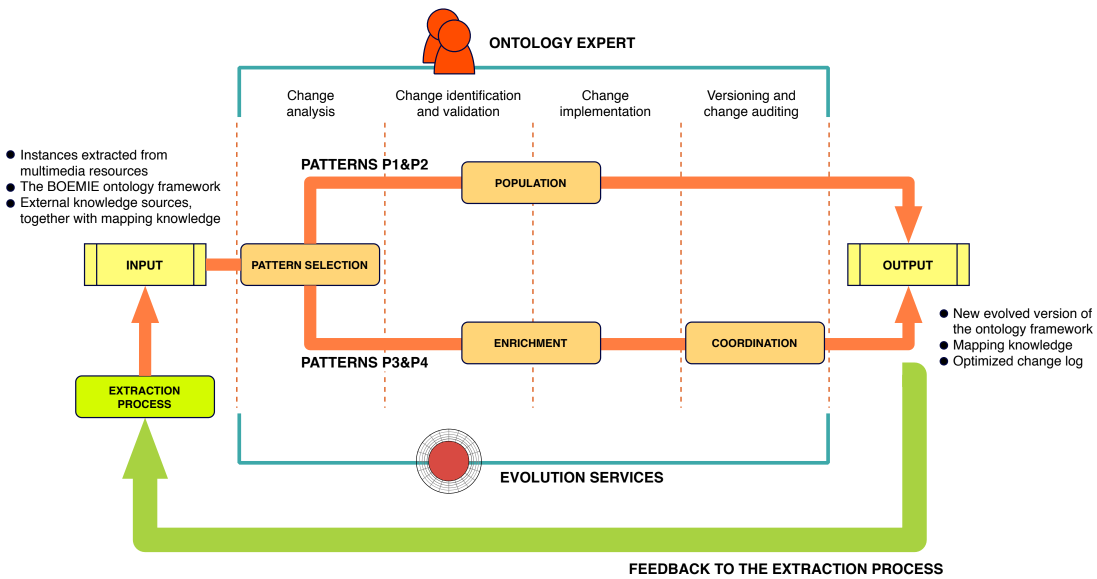

Figure 1: The BOEMIE evolution methodology

The BOEMIE evolution methodology is articulated in the following activities:

Pattern selection. The goal of the pattern selection activity is to analyze the input information in order to understand which pattern has to be triggered.

Population. Ontology population is the activity of adding new concept instances into the ontology. Ontology population is triggered by P1 and P2 evolution patterns, meaning that either a single (P1) or more (P2) explanations have been found during the interpretation activity to explain the extraction results for a single multimedia resource.

Enrichment. Ontology enrichment is the activity of extending the ontology, through the addition of new concepts/relations. Ontology enrichment is performed every time the background knowledge is not sufficient to explain the extracted information from the processed multimedia documents. Thus, the ontology enrichment activity is expected to extend this background knowledge through the addition of new concepts/relations, in order to better explain extracted information in subsequent cycles of the bootstrapping process.

Coordination. Coordination is the activity of producing a log of the changes introduced into the new evolved version of the ontology with respect to the previous version and of defining and updating the mapping knowledge between the domain ontology and other related external knowledge sources. Since this activity is affected by the changes of the background knowledge, it is executed after patterns P3 and P4.

Evolution services. The evolution process is based on a *reasoning service* and on a *matching service*. The reasoning service provides a set of standard and non-standard techniques. Standard techniques are used to ensure consistency upon population and enrichment activities. Non-standard reasoning techniques are used to support the activity of ontology enrichment. In particular, non-standard description logic inference problems are used for formalizing the learning problem in BOEMIE (e.g., LCS, MSC, rewriting). The matching service is used to perform all the ontology matching operations required by the methodology, namely: instance matching during the population activity; concept matching during the enrichment activity for knowledge discovery; concept matching during the coordination activity in order to define and update mappings between the BOEMIE ontology and the external knowledge sources.

Phases and activities. The evolution methodology activities that are executed in a cycle of the bootstrapping process are differently depending on the evolution pattern which is activated, that is the situation we have to deal with, and on the current evolution phase, that is what we are doing in that moment in order to evolve the ontology. This organization is shown in Table 1.

<table>
<tr><th>Phases/Patterns</th><th>Pattern 1</th><th>Pattern 2</th><th>Pattern 3</th><th>Pattern 4</th></tr>
<tr><td>Change analysis</td><td colspan="4">Pattern selection</td></tr>
<tr><td>Change identification and validation</td><td>Population (Instance matching, Instance grouping, ABox validation)</td><td>Population (Instance matching, Instance grouping, ABox refinement, ABox validation)</td><td>Enrichment (Concept learning, Concept enhancement, Concept definition)</td><td>Enrichment (Concept definition)</td></tr>
<tr><td>Change implementation</td><td colspan="2">Population
(ABox assimilation)</td><td colspan="2">Enrichment
(Concept assimilation)</td></tr>
<tr><td>Versioning and change auditing</td><td>-</td><td>-</td><td colspan="2">Coordination
(Ontology versioning and management,
Ontology alignment)</td></tr>
</table>

Table 1: Organization of the activities and tasks with respect to the evolution patterns and phases

The outcome of the proposed methodology is an open architecture that will use the BOEMIE multimedia semantic model and specifies the interfaces for the various types of ontology evolution tools and services (see Section 8).

Innovation. The development of the proposed evolution methodology compliant with the design principles described above requires the investigation of several research issues that are matter of investigation in several research fields, such as innovative non-standard reasoning services (see Section 4), advanced instance matching and clustering (see Section 5), “knowledge discovery”-enabled ontology enrichment (see Section 6). A more detailed description of the contribution to innovation is given in Section 9.

1 INTRODUCTION

Ontology evolution is generally defined as the timely adaptation of an ontology to changing requirements and the consistent propagation of changes to dependent artifacts [Maedche et al., 2003; Stojanovic et al., 2002]. Ontology evolution is a complex process whose ultimate goal is to ensure the consistency of the ontology and its related artifacts after applying any kind of ontology modification.

In BOEMIE, the goals and results of the ontology evolution methodology are defined in the project technical annex, p.55:

> A novel methodology for multimedia ontology evolution is required that will determine the manner in which the interlinked ontologies will be updated, preserving their semantic consistency. At the same time, the methodology will control the bootstrapping process, using the information extracted from the multimedia content and providing the necessary semantic information back to the information extraction modules. The outcome of the proposed methodology will be an open architecture that will use the BOEMIE multimedia semantic model and specify the interfaces for the various types of ontology evolution tools.

According to the BOEMIE Multimedia Semantic Model, the ontology is organized into several inter-linked ontological modules, where domain knowledge, multimedia feature knowledge, and spatio-temporal knowledge related to multimedia resource management are properly specified. Furthermore, according to the BOEMIE bootstrapping process, the ontology is enriched with new knowledge as a consequence of the extraction process and the new evolved ontology is used to improve the extraction process, thus triggering a new boostrapping cycle. The more the ontology evolves, the more it can extract; the more is extracted, the more the ontology evolves.

Thus, the BOEMIE ontology evolution methodology models a complex process involving the following activities:

- *ontology population and enrichment*, concerning the identification of the required ontology modifications (like the addition of new concepts/relations or the addition of new instances) and their implementation;
- *ontology coordination*, concerning the alignment against other external knowledge sources for the same domain that are exploited for knowledge discovery;
- *maintenance of semantic consistency*, concerning the validation of the ontology, to guarantee that no inconsistencies are introduced in any part of the ontology as a consequence of ontology modification(s).

The outcome of the proposed methodology is an open architecture, which will communicate with the semantic extraction modules described in the project deliverable D2.1 - Methodology and architecture for semantics extraction, by acquiring multimedia resources with different levels of associated semantic information and by providing back an evolved ontology, containing new information to improve the semantic extraction process. The architecture will also specify the interface for the evolution tools (namely, the population, enrichment, and coordination tools) and the support services (namely, the reasoning service and the matching service) defined in the present document. Consequently, the architecture will be completely open to the replacement of the tools with new ones in the future.

Additionally, the methodology will cover the evaluation of the evolution process based on the separate evaluations of the components for ontology population and enrichment and for ontology coordination. The integration of the whole evolution process into the BOEMIE framework will also be evaluated.

The deliverable is organized as follow. In Section 2, we present the design principles and requirements for the ontology evolution methodology. In Section 3, the evolution methodology is outlined. Section 4 is devoted to provide the logical foundations of the proposed methodology. Sections 5, 6, and 7 describe in detail the population, enrichment, and coordination activities of the methodology, respectively. In Section 8, we describe the open architecture for the ontology evolution. In Section 10, evaluation criteria are proposed while, in Section 12, conclusions are given.

2 THE MULTIMEDIA SEMANTIC MODEL AND THE BOOTSTRAPPING PROCESS OF BOEMIE

2.1 The Multimedia Semantic Model of BOEMIE

The Multimedia Semantic Model plays the role of providing the necessary background knowledge for the processes of semantics extraction from multimedia content and of ontology evolution in BOEMIE. To account for the different types of knowledge involved and for the functionalities required, the proposed Multimedia Semantic Model consists of several ontological modules, shown in Figure 2, each of them serving a different purpose in the knowledge structure. Maintaining such a separation of functionalities enables the independent use of each ontology, and furthermore, it allows for their linking, combination and extension. The Multimedia Semantic Model enables the implementation of a mixed bottom-up and top-down approach to media interpretation which is useful also in the ontology evolution process. The model represents the structure and content of multimedia documents, athletics events and their sub phases, geographic information, spatiotemporal relations, the various modalities, i.e. image, text, audio, video and finally the background knowledge for fusion purposes during the semantic extraction process.

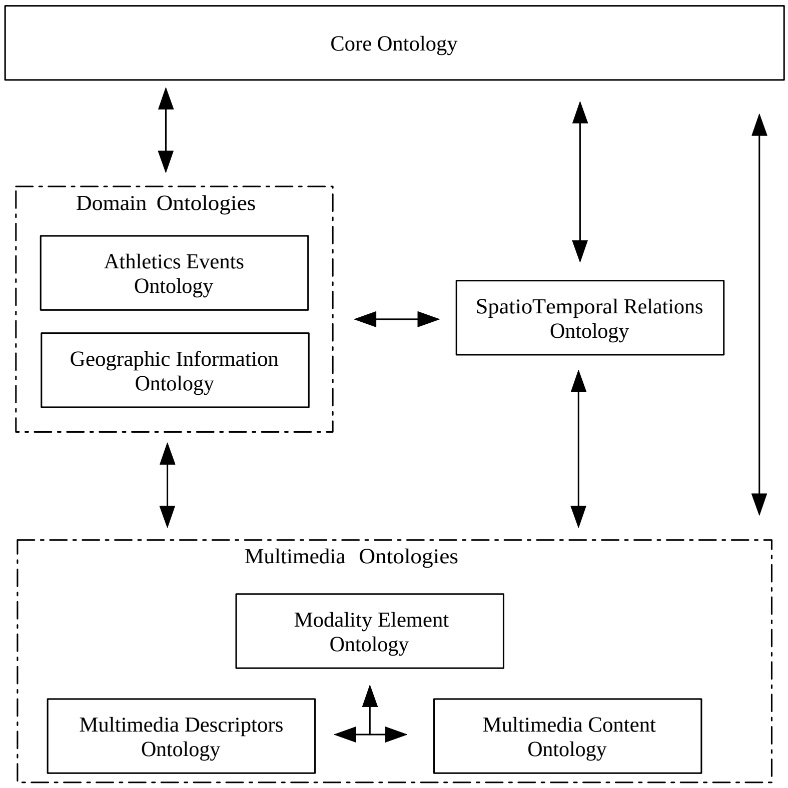

Figure 2: The BOEMIE Multimedia Semantic Model

The Multimedia Semantic Model is implemented with ontology languages deriving from the Ontology Web Language (OWL) family, such as OWL-DL, OWL 1.1 and SWRL. These ontology languages are based on Description Logics, knowledge representation formalisms that belong to the family of logic-based representation formalisms. Knowledge Representation Systems (KRS) using Description Logics describe the notions of the domain of interest using a variant of first order predicate calculus by first defining the relevant concepts of the domain (the terminology), and then using these concepts to specify properties of objects and individuals occurring in the domain (the world description), for a more detailed presentation see [Baader et al., 2003]. Another important feature of KRSs based on Description Logics is that they provide reasoning services, i.e. verification of logical consequence and ability to find implicit consequences of the explicitly represented knowledge. A more detailed introduction to Description Logics can be found in Section 4. In the following subsections the ontological modules of the Multimedia Semantic Model are presented, and by focusing on the definition of the sport event Pole Vault the interrelations-links between the modules are shown.

2.1.1 The Athletics Events Ontology

The Athletics Events Ontology represents the conceptualization of the domain of interest of the BOEMIE use case scenario which is related to public athletics events (i.e., jumping, running and throwing events held in European cities) at human level, i.e., the concepts and events of interest, their logical relations, their attributes and so on. The Athletics Ontology serves as the background knowledge Σ for the interpretation process of the various modalities multimedia documents, i.e. still images, videos, audio and text.

As an example, we provide hereafter the specification of the concept *Athlete* by taking into account the requirements that its description has to comply with the semantics extraction requirements of different modalities. For example, according to the textual analysis an athlete can be perceived as an entity that has one name, one age, one nationality, one gender, one ranking and participates in one round of a sports event, whereas, according to the visual analysis, an athlete can be perceived as an entity that has a face or a body. The following Description Logics axioms, shown in Figure 3, of the TBOx of the Athletics Events Ontology define the concept *Athlete* taking into account the requirements for both modalities.

$$
\begin{array}{lcl}
Person & \sqsubseteq & ((\exists has\_Part.Face) \sqcup (\exists has\_Part.Body)) \sqcup \\
 & & (\exists has\_Name.Name \sqcap \\
 & & \exists has\_Age.nonNegativeInteger \sqcap \\
 & & \exists has\_Gender.Gender \sqcap \\
 & & \exists has\_Nationality.Country) \\
Athlete & \equiv & Person \sqcap ((\exists participates\_At\_Round.Sports\_Event\_Round) \\
 & & \sqcup (\exists has\_Ranking.nonNegativeInteger)) \\
Person & \sqsubseteq & ((\exists_{=1}\ has\_Name) \sqcap \\
 & & (\exists_{=1}\ has\_Age) \sqcap \\
 & & (\exists_{=1}\ has\_Nationality) \sqcap \\
 & & (\exists_{=1}\ has\_Gender) \sqcup \\
 & & ((\exists_{=1}\ has\_Part.Face) \sqcup (\exists_{=1}\ has\_Part.Body)) \\
Athlete & \sqsubseteq & (\exists_{=1}\ has\_Ranking) \sqcap \\
 & & (\exists_{=1}\ participates\_At\_Round))
\end{array}
$$

Figure 3: The representation of the concept Athlete in the T-Box of the Athletics Ontology taking into account text and image analysis requirements

A detailed description of the concepts, relations and properties currently contained in the Athletics Events Ontology can be found in the Deliverable 3.2 Domain Ontologies.

Another important feature for the semantics extraction process is the distinction between Mid Level Concepts (MLCs) and High Level Concepts (HLCs), where the MLCs are base concepts that can be associated directly with automatically extracted multimedia features, and HLCs are concepts defined in terms of other concepts, either MLCs or HLCs. For instance the concept *Horizontal_Bar* is an MLC for image as it is associated directly with a visual representation of shape, size, position, color, texture. HLCs are concepts denoting aggregate entities in the domain, such as athletics events. An example of a HLC in the athletics domain, is the concept *Pole_Vault* which is defined as an aggregation of the MLCs *Pole* and *Horizontal_Bar* and of the HLC *Athlete* , shown in Figure 4.

$$\begin{array}{rcl} Pole\_Vault & \sqsubseteq & Jumping\_Event \sqcap \\ & & \exists has\_Part.Pole \sqcap \\ & & \exists has\_Part.Horizontal\_Bar \sqcap \\ & & \exists has\_Part.Foam\_Mat \sqcap \\ & & \exists has\_Part.Athlete \\ Jumping\_Event & \sqsubseteq & \exists takes\_Place\_In.Stadium \end{array}$$

Figure 4: The definition of the Pole Vault in the TBox of the athletics domain

2.1.2 The Geographic Information Ontology

According to the BOEMIE use case scenario it is important to consider also knowledge about geographic information such as Points Of Interest (POI), geographic areas, metrics, coordinates and so on. Some examples of classes describing sports POIs are the *Stadium*, *Sports_Center*, which are defined as subclasses of the class *Sports_POI*. Considering the example of the definition of the Pole Vault event in Figure 4, the use of the geographic ontology is here to provide the exact location where the event has taken place. This is done through the relation *takes_Place_In*, which relates the event with the stadium ,where it has taken place. The definition of the Stadium is shown in Figure 5, where also the city in which it is located is provided through the relation *is_Located_In* and the range of the relation *City*. More details about the classes, relations and properties included in the Geographic Information Ontology can be found in D3.2.

$$\begin{array}{rcl} Geopolitical\_Area & \sqsubseteq & Geographical\_Area \\ Country & \sqsubseteq & Geopolitical\_Area \\ City & \sqsubseteq & Geopolitical\_Area \\ Sports\_POI & \sqsubseteq & POI \\ Stadium & \sqsubseteq & Sports\_POI \sqcap (\exists is\_Located\_In.City) \end{array}$$

Figure 5: A tiny example of the TBox of the Geographic Information Ontology

2.1.3 Modality Element Ontology

As already mentioned, an important feature for the semantics extraction process is the distinction between mid-level concepts (MLCs) and high-level concepts (HLCs). Since this distinction constitutes critical knowledge for both analysis and reasoning tasks, appropriate definitions have been introduced in the Modality Element ontology to determine whether a concept constitutes an MLC for a particular modality. More specifically, the *MLC* concept has been defined as the set of domain concepts that instantiate the *has_Modality_Representation* relation. The subsumed, with respect to the different modalities, relations enable to further distinguish between *MLC_Visual*, *MLC_Textual* etc. concepts.

This ontology also models the representations that an MLC may have across the different modalities. Therefore, the *Modality_Representation* concept is subclassed with respect to the supported modalities, i.e., it is subclassed by the *Visual_Representation*, *Textual_Representation*, *Auditory_Representation* and *Video_Representation* concepts. Additional relations such as the *has_Visual_Representation* for the visual modality are defined with respect to the supported modalities. For each MLC a corresponding concept is introduced with respect to the considered modality. For example, in the Pole Vault definition the domain concept *Pole* might have a visual and a video representation, therefore the concepts *Pole_Visual* and *Pole_Textual* are defined as subclasses of the *Visual_Representation* and *Textual_Representation* concepts respectively in the Modality Element Ontology.

To account for the detection by analysis of modality representations that do not match to any of the currently available MLCs representation, the *unknown* concept is introduced as an additional subclass to each of the four modalities top level representation concept, e.g. the *Unknown_Visual* concept subclasses the *Visual_Representation* concept. It is important to preserve the modality distinction among the unknown representations extracted by analysis so that when at a later stage evolution is performed the correct modality representation will be associated with a newly introduced MLC. To support also the insertion of new HLCs, during the ontology population and enrichment process, the *UnknownMLC* and *UnknownHLC* concepts have been introduced as subclasses of the MLC and HLC classes respectively.

The part of the TBox of the Modality Element Ontology describing all the aforementioned is shown in Figure 6.

$$\begin{aligned}
MLC &\sqsubseteq \exists has_Modality_Representation.Modality_Representation \\
Visual_Representation &\sqsubseteq Modality_Representation \\
Textual_Representation &\sqsubseteq Modality_Representation \\
Auditory_Representation &\sqsubseteq Modality_Representation \\
Video_Representation &\sqsubseteq Modality_Representation \\
MLC_Visual &\sqsubseteq MLC \\
MLC_Textual &\sqsubseteq MLC \\
MLC_Auditory &\sqsubseteq MLC \\
MLC_Video &\sqsubseteq MLC \\
MLC_Visual &\sqsubseteq \exists_{\geq 1} has_Modality_Representation.Visual_Representation \\
MLC_Textual &\sqsubseteq \exists_{\geq 1} has_Modality_Representation.Textual_Representation \\
MLC_Auditory &\sqsubseteq \exists_{\geq 1} has_Modality_Representation.Auditory_Representation \\
MLC_Video &\sqsubseteq \exists_{\geq 1} has_Modality_Representation.Video_Representation \\
Pole_Visual &\sqsubseteq Visual_Representation \\
Pole &\sqsubseteq \exists has_Modality_Representation.Pole_Visual \\
Pole_Textual &\sqsubseteq Textual_Representation \\
Pole &\sqsubseteq \exists has_Modality_Representation.Pole_Textual \\
MLC_Unknown &\sqsubseteq MLC \\
HLC_Unknown &\sqsubseteq HLC
\end{aligned}$$

Figure 6: A part of the TBox of the Modality Element Ontology

2.1.4 Spatio-temporal Relations Ontology

This ontology module includes the definitions of the spatial and temporal relations which are exploited by all the other ontological modules of the Semantic Model. Some examples of spatial directional relations are the following *is_Left*, *is_Right*, *is_Above* and Topological *touches*,*overlaps*, etc. Some examples of temporal relations are the following: *before*, *starts*, *during*. Because THE expressiveness of Description Logics is not adequate enough for the representation of these relations, the Multimedia Semantic Model will be extended with so called DL-safe rules. An example of the definition of the Pole Vault event of the Athletics Events ontology using also the spatial relations *near* and *touches* is shown in Figure 7 and are explained in more detail in sections 4 and 5 of this document.

$$\begin{aligned}
Pole_Vault(X) \quad \rightarrow \quad & Jumping_Event(X), \\
& hasPart(X, A), Foam_Mat(A), \\
& hasPart(X, B), Athlete(B), \\
& hasPart(X, C), Horizontal_Bar(C), \\
& near(B, C), \\
& hasPart(X, D), Pole(D). \\
& touches(B, D).
\end{aligned}$$

Figure 7: Additional restrictions for *Pole_Vault* in the form of rules.

2.1.5 Multimedia Content Ontology

This ontological module represents the structure of the content of the multimedia documents. The top-level hierarchy consists of the following five concepts: *Image, Video, Audio, Audiovisual* and *Multimedia*. Each of these types has its own segment subclasses. A number of specialized subclasses deriving from the generic MPEG-7 Segment Description Scheme for describing the structure of multimedia content in time and space have been represented here. These subclasses describe the specific types of multimedia segments, such as video segments, moving regions, still regions and mosaics, which result from spatial, temporal, and spatiotemporal segmentation of the different multimedia content types. These classes are going to be used for the representation of the segments extracted from multimedia documents during the semantic extraction process. For example, a region that depicts a pole, extracted from an image during the image analysis will instantiate the concept *Still_Region* which is related with the concept *Image* through the property *has_Spatial_Decomposition* and will also be related to the concept *Pole_Visual* of the Modality Element Ontology through the relation *depicts*.

$$\begin{array}{rcl} Image & \sqsubseteq & \exists_{\geq 1} has\_Spatial\_Decomposition.Still\_Region \\ Still\_Region & \sqsubseteq & \exists_{\geq 1} depicts.Visual\_Representation \end{array}$$

Figure 8: A tiny example of the TBox of the Multimedia Content Ontology

2.1.6 Multimedia Descriptor Ontology

The Multimedia Descriptor Ontology includes the descriptor definitions with respect to the different multimedia modalities. The linking between a representation and the corresponding descriptors is performed though appropriately defined relations. The defined relations are organized in a hierarchical way; for example, the *has_DominantColorDescriptor* relation is subsumed by the *has_ColorDescriptor* relation which in turn is subsumed by the *has_VisualDescriptor* one. In the following Figure 9, an example of how the still region that depicts the visual representation of the pole (aforementioned *Pole_Visual*), is related to the low-level descriptors, such as the Color Descriptor and Texture Descriptor through the relation has_Visual_Descriptor is shown.

$$\begin{array}{rcl} Color\_Descriptor & \sqsubseteq & Visual\_Descriptor \\ Texture\_Descriptor & \sqsubseteq & Visual\_Descriptor \\ Still\_Region & \sqsubseteq & \exists_{\geq 1} has\_Visual\_Descriptor.Visual\_Descriptor \end{array}$$

Figure 9: A tiny example of the TBox of the Multimedia Descriptor Ontology

2.1.7 Core ontology

The use of a core ontology in still under investigation. Using a core ontology to harmonize the other ontology modules has certain advantages in terms of ensuring extensibility and easy adaptation to different domains, while cleaner ontology engineering can be achieved, e.g., when more than one domain-specific ontologies are needed to fully capture the required knowledge. However, the evolution process described in the following will remain the same also when a core ontology is employed [1].

2.2 The BOEMIE bootstrapping process

The BOEMIE project employs a bootstrapping approach, aiming to improve the overall performance of the system when operating in a constantly changing environment. The bootstrapping process is an iterative process, which uses an initial ontology to analyze and extract information from a corpus. The extracted information can be used to evolve the ontology, and through the evolved ontology the extraction of information can be improved. This cycle of using an ontology to extract information, which in return can be used to evolve the ontology is repeated until no more new information can be extracted from the corpus. The overall architecture of the bootstrapping process is shown in Figure 10. The bootstrapping process initiates by assuming that an initial

[1] For a detailed description of pros and cons of using a core ontology, the reader can refer to the project deliverables "D3.1, Multimedia Content and Descriptor Ontologies" and "D3.2, Domain Ontologies".

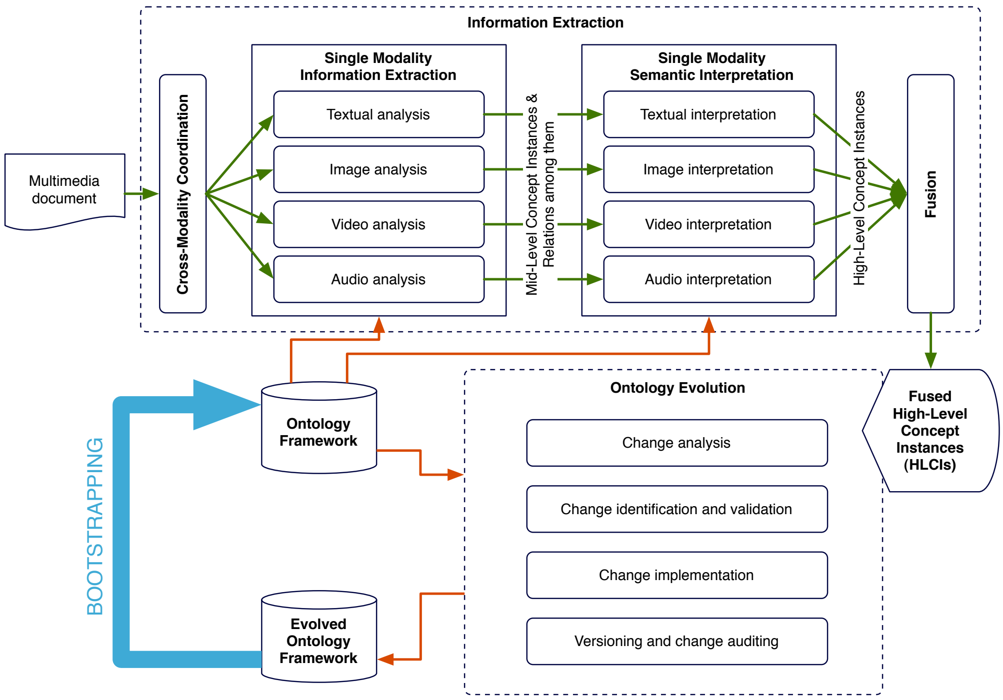

Figure 10: The BOEMIE Bootstrapping process

ontology exists, along with an information extraction system that is able to extract information related to mid-level concepts (MLCs) in the ontology. When a multimedia document is available to be processed, it goes through the first processing stage, the cross-modality coordination stage. The cross-modality coordinator is responsible for separating the multimedia document (possibly comprising more than one modality, e.g., text, images) into single-modality document elements. Each single-modality document element is processed by the corresponding single-modality information extraction system, in order to extract a set of mid-level concept instances (MLCis), along with a set of relations among these MLCis. The next processing stage, the single-modality semantic interpretation employs reasoning to construct a set of high-level concept instances, that explain every single-modality document element. Finally, fusion, the last processing stage of the information extraction phase, merges the explanations of single-modality document elements in order to produce a set of explanations for the original multimedia document. Each explanation is a high-level concept instance (HLCi), involving several MLCis and relations among them. These HLCis constitute the input to the ontology evolution phase. The ontology evolution phase receives as input the HLCis (explanations) produced by the information extraction phase and aims at the construction of an enhanced version of the ontology. The main phases of evolution are change analysis, change identification and validation, change implementation, and versioning. In the rest of the document, we provide a detailed description of the evolution process and of the methodology that is followed.

3 METHODOLOGY FOR ONTOLOGY EVOLUTION

In this section, the BOEMIE evolution methodology is described according to the input and the output, the methodological phases, the activities and tasks, and the supporting services enabling the execution of the different activities. The organization of the activities in the different phases depends on the different situations that can occur during the evolution process and on the connotations that feature the input data. In order to capture these different situations, we introduce in the methodology four *evolution patterns* that are described in this section together with their corresponding activities.

3.1 Design principles

Ontology evolution is a complex activity that aims to ensure the consistency of the ontology and possible dependent artifacts after that any modification has been applied. Ontology evolution becomes even more complex in the context of BOEMIE, due to the management of multimedia resources and ontologies and to the multi-modal semantic information extraction to be enabled. To this end, based on the state of the art in the field of ontology evolution methodologies and techniques, we define an evolution methodology that takes into account the new requirements posed by multimedia information extraction and management.

The design choices of the overall methodology have been driven by the ontology-oriented architecture the BOEMIE: the domain of application across all media is modeled through an ontology whereas the output of semantics extraction is used to trigger the appropriate evolution patterns resulting either in ontology population or ontology enrichment followed by ontology coordination.

The main design choices that have driven the definition of the Multimedia Ontology Evolution methodology are summarized as follows.

- *Evolution support in the presence of multiple ontological modules.* In order to provide the necessary background knowledge for the automated processes of semantics extraction from multimedia content and ontology evolution, the Multimedia Semantic Model consists of several interlinked ontological modules, each of them serving a different purpose in the knowledge space structure. Maintaining such a division of functionalities enables the independent use of each ontology knowledge space (i.e., an ontology module), and furthermore, allows for their linking, combination and extension. This requires to develop an evolution methodology capable of managing the coordinated evolution of such interlinked ontology modules, by which the modification applied to a part of the ontology will be automatically and consistently propagated to other interlinked ontological modules, to keep the them aligned and consistent each other.

- *Pattern-driven evolution.* Ontology evolution is affected by the background knowledge actually available in the ontology and by the kind and number of media objects produced by the extraction process. The different combinations of this available information correspond to typical situations that trigger ontology evolution in BOEMIE, each one requiring an appropriate combination of ontology modification activities. We design a pattern-driven evolution methodology, where different evolution patterns are identified, each one dealing with a different evolution scenario that can occur, automatically identified on the basis of the results of the multimedia extraction activity against the background knowledge.

- *Open system approach.* When background knowledge is not sufficient to interpret new incoming instances, new knowledge, such as a new concept, has to be added to the ontology. Concept definition is usually a manual, human-intensive activity which requires the ontology expert to find the most appropriate specification of the missing concept. In BOEMIE, we support a knowledge discovery approach typical of open networked systems for enriching the local knowledge of a given node. Thus, we design the evolution methodology to provide a concept discovery functionality which acquires new knowledge by probing external knowledge sources for the same domain, looking for candidate concepts semantically matching the incoming instance. This way, the ontology knowledge space is open towards external knowledge sources, which can provide useful knowledge for evolving the ontology adequately.

- *Minimization of human involvement.* A completely automated evolution process is not feasible, due to the intrinsically interactive nature of the evolution process, which eventually requires a qualified supervision of the ontology expert. However, it is commonly recognized that a key issue for an evolution methodology is the minimization of human involvement, especially for the labor-intensive manual activities related to new concept identification and definition. To this end, we design a methodology providing a set of learning, matching, and reasoning techniques that offer support in the various evolution activities, to allow the ontology expert to refine the proposed working knowledge (e.g., discovered candidate concepts) and/or to validate/choose among proposed alternative choices, thus limiting the human involvement as much as possible to expected interactions with support tools.

3.2 Overview of the methodology

The ontology evolution methodology of BOEMIE is shown in Figure 11.

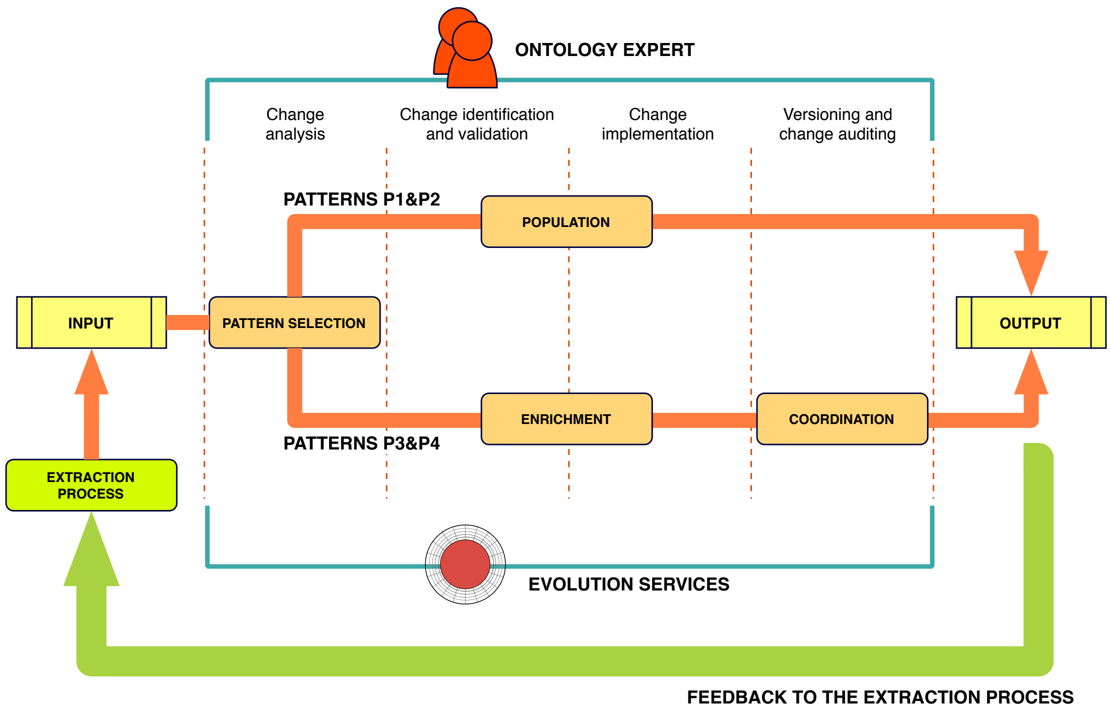

Figure 11: The BOEMIE evolution methodology

The main elements of the methodology are the following:

- **Input**
  - Instances extracted from multimedia resources
    1. A set of high-level concept instances (HLCis)
    2. A set of mid-level concept instances (MLCis)
    3. A set of modality descriptors without and associated MLC (Unknown MLCis)
  - The BOEMIE semantic model
  - A set of external knowledge sources
- **Output**
  - Evolved BOEMIE semantic model

  - Mapping knowledge
  - Log of ontology changes

- **Phases**
  - Change analysis
  - Change identification and validation
  - Change implementation
  - Versioning and change auditing

- **Evolution patterns**
  - Single concept explanation (P1)
  - Multiple concept explanation (P2)
  - Missing concept with metadata explanation (P3)
  - Missing concept without metadata explanation (P4)

- **Activities/Tasks/Operations**
  - Pattern selection
  - Population
    1. Instance matching
    2. Instance grouping
    3. ABox refinement
    4. ABox validation
    5. ABox assimilation
  - Enrichment
    1. Concept learning
       (a) Clustering
       (b) Concept formation
    2. Concept enhancement
    3. Concept definition
    4. Concept validation
    5. Concept assimilation
  - Coordination
    1. Ontology versioning and management
    2. Ontology alignment
       (a) Mapping definition
       (b) Mapping maintenance

- **Supporting services**
  - Reasoning
  - Learning matching

- **Roles**
  - Ontology expert

3.3 Input

The input of the evolution methodology is composed of:

- A set of concept instances extracted from the multimedia documents in the information extraction process. Given a multimedia resource, the extraction process has the aim of identifying in the resource a set of multimedia objects and to give an interpretation of the whole resource by associating (if possible) the media objects with a high-level concept (HLC) denoting a real object or an event. The input of the ontology evolution is the result of the extraction process, that can be:
  - A single HLCi or a set of HLCis describing the resources. This input is available when the extraction process is able to produce an explanation for the multimedia resources.
  - A set of MLCis describing the media objects in the resources without a HLCi. In this case, resource interpretation failed, but the media objects retrieved in the resources have been identified and, thus, associated with the corresponding MLC.
  - A set of modality descriptors associated with an unknown MLC. In this case, background knowledge is not enough to even assign an MLC to all extracted elements of a multimedia document, thus leaving extracted modality descriptors not associated with a MLC.
- The BOEMIE semantic model that contains the domain ontology together with related multimedia knowledge modules. In the initial cycle of the bootstrapping process, the domain ontology is manually defined and it could contain a low level of detail in the description of the domain. Through evolution a new model is produced which is used as input in the subsequent cycle.
- External knowledge sources (e.g., other domain ontology, taxonomies of web directories, lexical systems) that are aligned with the BOEMIE ontology during the versioning and change auditing phase of the evolution. In the initial cycle of the bootstrapping process, these resources are manually identified and provided to the system. In the subsequent cycles of the bootstrapping, new external sources are automatically considered in the coordination activity. Together with the external knowledge sources, the evolution methodology input is constituted also by the mapping knowledge that provides the mappings stored between the domain ontology and the external knowledge sources in the previous cycles of the bootstrapping process.

3.4 Output

The output of the evolution process is:

- The new evolved version of the semantic model that can contain new instances (population) and/or new concepts (enrichment).
- The mapping knowledge, which contains information about the mappings defined between the concepts in the BOEMIE domain ontology and the concepts in the external knowledge sources.
- A change log, which provides information about the operations that have been executed on the previous version of the ontology in order to define the new evolved version.

3.5 Evolution patterns

The evolution methodology aims to address the typical situations that can take place in evolving the ontology knowledge in BOEMIE, based on new information extracted from multimedia content. This activity is called *interpretation*. In particular, interpretation associates the contents of the resources with the concepts contained in the ontology. In other terms, the ontology is the background knowledge that we use to explain the resources. The expected result of the interpretation is to find a single explanation for a resource, but, if the background knowledge is not sufficient to explain a resource, it could happen that more than one possible explanation is found, or that no explanations at all are produced. Finally, it is also possible that neither the explanation of the identified media objects can be produced. These different results of the interpretation process lead to different situations in ontology evolution. These situations are managed by means of four *evolution patterns*. The interpretation outcomes and the patterns are summarized in Table 2.

Number of HLC explaining the multimedia resource	**1**	**>1**	**0**
MLC explanation available	P1	P2	P3
MLC explanation not available	P4	P4	P4

Table 2: Typical situations that occur in ontology evolution

Patterns P1 and P2 describe the situation where the interpretation has found one or more HLCs explaining the resources and, thus, the ontology population activity is performed. Patterns P3 and P4 describe the situation where no HLC explaining the resources is retrieved, triggering enrichment. The difference between P3 and P4 is that, in P4, we have also to deal with missing MLCis for some extracted media object. In case of missing explanations of MLCis, pattern P4 is always enforced to evolve the ontology in order to provide an interpretation of unknown media objects. Then, in the subsequent cycle of the bootstrapping, the most suitable pattern will be chosen for handling the new situation.

P1: Single concept explanation. In pattern P1, a single HLC explains the resource. As an example, consider an image of a pole vault, where all the required information to conclude that it is a pole vault (e.g., the athlete jumping over a horizontal bar, together with a pole) is specified in the ontology. In populating the ontology with the new pole vault scene, possible inconsistencies with the instances already contained in the ontology must be checked.

P2: Multiple concept explanation. In pattern P2, more than one HLC could explain the resource. As an example, consider an image of an athlete jumping over a horizontal bar. Such an image could describe a pole vault or high jump event. In this case, we find a HLCi in the ontology for each possible explanation of the resource. Moreover, we analyze the instances already contained in the ontology in order to find other information about the depicted event. The idea is that, if we find other assertions about the same fact, already classified with respect to a given HLC, we can exploit this information to select the right HLCi out of all HLCis.

P3: Missing concept with metadata explanation. In pattern P3, the background knowledge does not contain any HLC suitable for explaining the resources, but all the multimedia objects described in the resources have been extracted and interpreted. As an example, suppose to have an image of a triple jump event but a HLC Triple Jump is not defined in the ontology. In this case, the multimedia objects extracted from the resources (i.e., the resources metadata) are used to support learning and definition of a new HLC to be used to explain the incoming resources in subsequent bootstrapping cycles.

P4: Missing concept without metadata explanation. In pattern P4, HLCs explaining the resources are not available in the ontology and the resources contain one or more media objects that have not been associated with any MLC during the extraction process. As an example, suppose to have again a pole vault image, where the pole is not recognized as a pole because the MLC Pole is not present in the ontology. In this situation, an important component of the resource is missing and we have to enrich the ontology by introducing this new MLC concept in order to correctly explain the resource in the subsequent bootstrapping cycle. In other words, we consider the detection of the missing MLC to have priority over the identification of the missing HLC, because knowing about the new MLC could lead to different interpretation results about the same resources. The new interpretation will thus be performed in the next bootstrapping cycle.

3.6 Activities and tasks

The activities and the tasks involved in the ontology evolution methodology are defined as follows.

Pattern selection. The goal of the pattern selection activity is to analyze the input information in order to understand which pattern has to be triggered. This is done by considering the results of the interpretation process and by verifying how many explanations are available for the incoming resource as well as the presence of unexplained media objects. The ontology evolution process works on top of the results of the interpretation activity performed on multimedia documents during the extraction process. For the purposes of pattern selection, only on the ABox produced as a result of the interpretation is considered, while the documents from which the ABox description has been derived are no more considered. An example could clarify this situation. If in the same HTML document describes both a High_Jump and a Pole_Vault event, the interpretation activity will recognize two different events (i.e., HLC) in the document and the evolution process will receive an ABox containing two instances of the two different HLCs, thus leading to the activation of pattern P1 for each of them.

Population. Ontology population is the activity of adding new concept instances into the ontology. Ontology population is triggered by P1 and P2 evolution patterns, meaning that either a single (P1) or more (P2) explanations (HLCis) have been found during the interpretation activity to explain the extraction results for a single multimedia resource. The tasks of the population activity are the following:

- *Instance matching*: the goal of instance matching is the identification of similar instances in the domain ontology for each HLCi in the set of explanations. Having a single HLCi as input, instance matching returns a set of instances (ABoxes) that are similar to the incoming ABox, together with a measure of their similarity. The idea behind instance matching is to identify instances that represent the same real object or event and possibly help in disambiguating multiple explanations in the case of evolution pattern P2 (ABox refinement).
- *Instance grouping*: instance grouping has the goal of defining a group of instances that contains all the instances that represent the same real object or event (exploiting the results of the instance matching task).
- *ABox refinement* (evolution pattern P2 only): in case of multiple explanations, the most suitable explanation is selected by exploiting the results of the instance grouping task. Assertions related to the rest of the explanations are removed from the ABox, thus leading to a refined version of it.
- *ABox validation*: the goal of ABox validation is to detect possible inconsistencies among the instances that are assimilated in the ontology. ABox validation is performed through standard inference services (see Section 4).
- *ABox assimilation*: ABox assimilation has the goal of adding the ABox into the ontology, by creating all other needed instances/relations with reference to the other ontological modules of the semantic model.

Enrichment. Ontology enrichment is the activity of extending an ontology, through the addition of new concepts and relations. Ontology enrichment is performed every time the background knowledge is not sufficient to explain the extracted information from the processed multimedia documents. Thus, the ontology enrichment activity is expected to extend this background knowledge through the addition of new concepts/relations, in order to better explain extracted information in subsequent cycles of the bootstrapping process.

The ontology enrichment activity is triggered by either P3 or P4 and is articulated in the following tasks:

- *Concept learning*: concept learning has the goal of identifying possible new elements (either HLCs, relations, or MLCs) by exploiting similarities found through clustering, either in unexplained documents (evolution pattern P3) or in unknown media objects recognized by the information extraction toolkit (evolution pattern P4). It can be decomposed into two main tasks:

- *Clustering*: the goal of clustering is to support the creation of new concepts or relations.
- *Concept formation*: Exploiting the results of clustering, concept formation examines the ABoxes of the clustered elements in order to extract common information (e.g., properties, concepts, relations) and use it to form the new concept.

- *Concept enhancement*: the goal of concept enhancement is to extend the knowledge about a learned concept, with knowledge about concept name and possible definition acquired from external sources, such as other domain ontologies or taxonomies.

- *Concept definition*: in this task, information about the new concept that has been produced by the previous tasks is shown to the ontology expert. The ontology expert is required to fix the definition of the new concept in order to assimilate it into the ontology. The ontology expert can interactively revise the new concept definition before its assimilation.

- *Concept validation*: this task performs consistency checking, by trying to detect possible inconsistencies due to the addition of the new concept in the ontology.

- *Concept assimilation*: finally, the needed changes in the ontology are performed in order to incorporate the new concept into the domain ontology.

Unlike the population activity, ontology enrichment changes the TBox of the ontology and thus the background knowledge contained in the domain ontology. BOEMIE subsystems that rely on this background knowledge (such as the information extraction toolkit) must be notified about TBox changes, as they need to adapt to the modified background knowledge (e.g., a re-training of the extractors could be required).

Coordination. Coordination is the activity of producing a log of the changes introduced into the new evolved version of the ontology with respect to the previous version and of defining and updating mappings between the domain ontology and other external knowledge sources. Since this activity is affected by the changes of the background knowledge, it is executed after patterns P3 and P4. Ontology versioning has the aim of maintaining the new version of the ontology with respect to the previous version by providing a log of the changes that have been implemented. Finally, the ontology is aligned with the external knowledge sources by defining appropriate mappings. When a concept is modified during the enrichment activity, its existing mappings have to be updated accordingly. Coordination is executed also upon modifications that can occur in the external knowledge sources over time. The coordination activity is articulated according to the following tasks:

- *Ontology versioning and management*: the goal of the versioning task is to maintain a log of the changes between the new version of the ontology and the previous version, by providing details about the change operations that have been performed as well as the ontology expert annotations on the new version.

- *Ontology alignment*, comprising the following operations:
  - *Mapping definition*: the goal of mapping definition is to describe explicitly the level of similarity between a (new) concept in the domain ontology and other concepts in the external knowledge sources.
  - *Mapping maintenance*: the goal of mapping maintenance is to update the existing mapping knowledge according to the changes over time either in the domain ontology (enrichment) or in the external knowledge sources (changes).

3.7 Phases

Ontology evolution is a complex process that requires to ensure the consistency of the ontology and possible dependent artifacts after that any modification has been applied. To this end, in the state of the art, a six-phase evolution process has been proposed in [Stojanovic et al., 2002], which is generally recognized as a comprehensive reference methodology capable of handling the evolution of multiple ontologies. With respect to this reference, the BOEMIE evolution methodology is articulated in four phases, namely *change analysis*, *change identification and validation*, *change implementation*, and *versioning and change auditing*. In the change analysis phase, the input instances are analyzed with respect to the background knowledge, with the goal of determining which evolution situation occurs, that is which evolution pattern has to be chosen. In the change identification and validation phase, according to the selected pattern, the changes that are required on the ontology are determined and, before their implementation, the possible resulting ontology is validated in order to check its consistency with respect to the modifications. This phase is iterated until a valid solution is detected. If it is not possible to modify the ontology by preserving its consistency, the ontology remains unchanged. In the change implementation phase, the updating of the ontology is enforced and an intermediate version is released. Finally, in the versioning and change auditing phase, a log of the changes is produced and the mappings between the domain ontology and the external knowledge sources are defined. The evolution methodology activities that are executed in a cycle of the bootstrapping process depend on the evolution pattern which is activated and on the current evolution phase, as shown in Table 3

Phases/Patterns	**Pattern 1**	**Pattern 2**	**Pattern 3**	**Pattern 4**
Change analysis	Pattern selection			
Change identification and validation	Population (Instance matching, Instance grouping, ABox validation)	Population (Instance matching, Instance grouping, ABox refinement, ABox validation)	Enrichment (Concept learning, Concept enhancement, Concept definition)	Enrichment (Concept definition)
Change implementation	Population (ABox assimilation)		Enrichment (Concept assimilation)	
Versioning and change auditing	-	-	Coordination (Ontology versioning and management, Ontology alignment)	

Table 3: Organization of the activities and tasks with respect to the evolution patterns and phases

3.8 Supporting services

The evolution methodology is based on a *reasoning service* and on a *matching service*.

The reasoning service is used to perform the ABox validation task during the population activity, and to support the concept learning and concept validation task during enrichment.

The matching service is used to perform the instance matching task during the population activity, the concept enhancement task during the enrichment activity, and the mapping definition and mapping maintenance tasks during coordination.

3.9 Roles: the ontology expert

In BOEMIE, the ontology evolution process is interactive. One of the goals of the proposed methodology is to minimize the burden on the ontology expert by providing as much automated support as possible to the various activities and tasks. In particular, two kinds of interaction with the expert are expected in the evolution methodology:

- Parameter settings: the designer is required to properly set all the involved parameters (e.g., the matching threshold in coordination). A default value is provided and the designer can modify it if required.
- Result inspection/validation, definition/revision: the ontology expert is asked to validate the results of the automated population, enrichment, and coordination activities. However, the proposed solutions can be always modified to manually enforce a specific design choice if required.

In particular, the role of the expert per activity is as follows:

- **Population**
  - *Selection of HLCi*: In the case of evolution pattern P2, if the ABox refinement activity is not successful in choosing only one HLCi that correspond to the incoming instance, the ontology expert intervention is needed to choose the correct HLCi or even decide that the current instance is still unexplained, meaning that probably a new concept is needed.
- **Enrichment**
  - *Concept (HLC) definition*: The ontology expert is responsible for defining a new concept when the evolution pattern selected is P3 or P4. Although he is supported by the results obtained automatically from the concept enhancement activity, the ontology expert is required to actually define the concepts.
  - *Concept (MLC) definition:* In the case of pattern P4, when the definition of a new MLC is needed, the ontology expert is responsible for this task.
- **Coordination**
  - *Parameter setting*: the ontology expert is requested, when necessary, to set one or more parameters to configure the matching service to perform the various ontology matchings required in the methodology.
  - *Mapping validation*: the ontology expert is responsible for validating and choosing the mappings that need to be updated as a consequence of ontology enrichment activity; this is done according to the measure of change determined in the ontology alignment task.

4 LOGICAL FOUNDATIONS AND REASONING SERVICES

Ontology languages of the OWL family, which provide the skeleton for research on ontology evolution based on media interpretation, are based on description logics (DLs). In this chapter we introduce the logical basis of several ontology languages of the OWL family, define their semantics, and specify corresponding reasoning services. In the following subsections, we start with so-called expressive description logics (approximately corresponding to but slightly more expressive than OWL Lite), introduce additional constructs afterwards (corresponding to OWL DL, OWL 1.1), and also specify other fragments of first-order logic, some of which have also been standardized by W3C acitivities.

4.1 Expressive Description Logics: Syntax and Semantics

The DL $\mathcal{ALCQHI}_{R^+}(\mathcal{D})^-$ which is also known as $\mathcal{SHIQ}$ is briefly introduced as follows. We assume five disjoint sets: a set of concept names C, a set of role names R, a set of feature names F, a set of individual names O and a set of names for (concrete) objects O_C. The mutually disjoint subsets P and T of R denote non-transitive and transitive roles, respectively ($R = P \cup T$). $\mathcal{ALCQHI}_{R^+}(\mathcal{D})^-$ is introduced in Figure 12 using a standard Tarski-style semantics with an interpretation $\mathcal{I}_\mathcal{D} = (\Delta^\mathcal{I}, \Delta^\mathcal{D}, \cdot^\mathcal{I})$ where $\Delta^\mathcal{I} \cap \Delta^\mathcal{D} = \emptyset$ holds. A variable assignment α maps concrete objects to values in $\Delta^\mathcal{D}$.

In accordance with [Baader and Hanschke, 1991] we also define the notion of a concrete domain. A *concrete domain* $\mathcal{D}$ is a pair $(\Delta_\mathcal{D}, \Phi_\mathcal{D})$, where $\Delta_\mathcal{D}$ is a set called the domain, and $\Phi_\mathcal{D}$ is a set of predicate names. The interpretation function maps each predicate name P from $\Phi_\mathcal{D}$ with arity n to a subset $\mathsf{P}^\mathcal{I}$ of $\Delta^n_\mathcal{D}$. Concrete objects from O_C are mapped to an element of $\Delta^\mathcal{D}$. We assume that $\bot_\mathcal{D}$ is the negation of the predicate $\top_\mathcal{D}$. A concrete domain $\mathcal{D}$ is called *admissible* iff the set of predicate names $\Phi_\mathcal{D}$ is closed under negation and $\Phi_\mathcal{D}$ contains a name $\top_\mathcal{D}$ for $\Delta_\mathcal{D}$, and the satisfiability problem $\mathsf{P}_1^{\mathsf{n}_1}(\mathsf{x}_{11}, \ldots, \mathsf{x}_{1\mathsf{n}_1}) \wedge \ldots \wedge \mathsf{P}_\mathsf{m}^{\mathsf{n}_\mathsf{m}}(\mathsf{x}_{\mathsf{m}1}, \ldots, \mathsf{x}_{\mathsf{m}\mathsf{n}_\mathsf{m}})$ is decidable (m is finite, $\mathsf{P}_\mathsf{i}^{\mathsf{n}_\mathsf{i}} \in \Phi_\mathcal{D}$, n_i is the arity of P_i, and x_{jk} is a concrete object).

If $\mathsf{R}, \mathsf{S} \in R$ are role names, then $\mathsf{R} \sqsubseteq \mathsf{S}$ is called a *role inclusion axiom*. A *role hierarchy* $\mathcal{R}$ is a finite set of role inclusion axioms. Then, we define $\sqsubseteq^*$ as the reflexive transitive closure of $\sqsubseteq$ over such a role hierarchy $\mathcal{R}$. Given $\sqsubseteq^*$, the set of roles $\mathsf{R}^\downarrow = \{\mathsf{S} \in R \,|\, \mathsf{S} \sqsubseteq^* \mathsf{R}\}$ defines the *sub-roles* of a role R. R is called a super-role of S if $\mathsf{S} \in \mathsf{R}^\downarrow$. We also define the set $S := \{\mathsf{R} \in P \,|\, \mathsf{R}^\downarrow \cap T = \emptyset\}$ of *simple* roles that are neither transitive nor have a transitive role as sub-role. Due to undecidability issues number restrictions are only allowed for simple roles (see [Horrocks et al., 2000b]). In concepts, inverse roles R^{-1} (or S^{-1}) may be used instead of role names R (or S). If C and D are concepts, then $\mathsf{C} \sqsubseteq \mathsf{D}$ is a terminological axiom (*generalized concept inclusion* or *GCI*). A finite set of terminological axioms $\mathcal{T}_\mathcal{R}$ is called a *terminology* or *TBox* w.r.t. to a given role hierarchy $\mathcal{R}$.[2] An *ABox* $\mathcal{A}$ is a finite set of assertional axioms as defined in Figure 12c.

An interpretation $\mathcal{I}$ is a *model* of a concept C (or *satisfies* a concept C) iff $\mathsf{C}^\mathcal{I} \neq \emptyset$ and for all $\mathsf{R} \in R$ it holds that iff $(x, y) \in \mathsf{R}^\mathcal{I}$ then $(y, x) \in (\mathsf{R}^{-1})^\mathcal{I}$. An interpretation $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ iff it satisfies all axioms in $\mathcal{T}$ (see Figure 12b). An interpretation $\mathcal{I}$ is a model of an ABox $\mathcal{A}$ w.r.t. a TBox $\mathcal{T}$ iff it is a model of $\mathcal{T}$ and satisfies all assertions in $\mathcal{A}$ (see Figure 12c). Different individuals are mapped to different domain objects (unique name assumption). Note that features are interpreted differently from features in [Baader and Hanschke, 1991].

Reasoning about objects from other domains (so-called concrete domains, e.g. for real numbers) is very important for practical applications, in particular, in the context of the Semantic Web. For instance, one might want to express intervals for integer values ("the price range is between 200 and 300 Euro"), state the relationship between the Fahrenheit and Celsius scales, or describe linear inequalities ("the total price for the three goods must be below 60 Euro"). In [Baader and Hanschke, 1991] the description logic $\mathcal{ALC}(\mathcal{D})$ is investigated and it is shown that, provided a decision procedure for the concrete domain $\mathcal{D}$ exists, the logic $\mathcal{ALC}(\mathcal{D})$ is decidable. Unfortunately, adding concrete domains to expressive description logics (DLs) such as $\mathcal{ALCNH}_{R^+}$ [Haarslev and Möller, 2000b] might lead to undecidable inference problems. In [Haarslev et al., 2001b] it has

[2] The reference to $\mathcal{R}$ is omitted in the following if we use $\mathcal{T}$.

(a)

Syntax	Semantics
Concepts ($\mathsf{R} \in R$, $\mathsf{S} \in S$, and $\mathsf{f}, \mathsf{f_i} \in F$)	
A	$\mathsf{A}^\mathcal{I} \subseteq \Delta^\mathcal{I}$ (A *is a concept name*)
$\neg \mathsf{C}$	$\Delta^\mathcal{I} \setminus \mathsf{C}^\mathcal{I}$
$\mathsf{C} \sqcap \mathsf{D}$	$\mathsf{C}^\mathcal{I} \cap \mathsf{D}^\mathcal{I}$
$\mathsf{C} \sqcup \mathsf{D}$	$\mathsf{C}^\mathcal{I} \cup \mathsf{D}^\mathcal{I}$
$\exists\, \mathsf{R} \,.\, \mathsf{C}$	$\{\mathsf{a} \in \Delta^\mathcal{I} \mid \exists\, \mathsf{b} \in \Delta^\mathcal{I} : (\mathsf{a}, \mathsf{b}) \in \mathsf{R}^\mathcal{I} \wedge \mathsf{b} \in \mathsf{C}^\mathcal{I}\}$
$\forall\, \mathsf{R} \,.\, \mathsf{C}$	$\{\mathsf{a} \in \Delta^\mathcal{I} \mid \forall\, \mathsf{b} \in \Delta^\mathcal{I} : (\mathsf{a}, \mathsf{b}) \in \mathsf{R}^\mathcal{I} \Rightarrow \mathsf{b} \in \mathsf{C}^\mathcal{I}\}$
$\exists_{\geq n}\, \mathsf{S} \,.\, \mathsf{C}$	$\{\mathsf{a} \in \Delta^\mathcal{I} \mid \lVert\{\mathsf{y} \mid (\mathsf{x}, \mathsf{y}) \in \mathsf{S}^\mathcal{I}, \mathsf{y} \in \mathsf{C}^\mathcal{I}\}\rVert \geq \mathsf{n}\}$
$\exists_{\leq m}\, \mathsf{S} \,.\, \mathsf{C}$	$\{\mathsf{a} \in \Delta^\mathcal{I} \mid \lVert\{\mathsf{y} \mid (\mathsf{x}, \mathsf{y}) \in \mathsf{S}^\mathcal{I}, \mathsf{y} \in \mathsf{C}^\mathcal{I}\}\rVert \leq \mathsf{m}\}$
$\exists\, \mathsf{f}_1, \ldots, \mathsf{f_n} \,.\, \mathsf{P}$	$\{\mathsf{a} \in \Delta^\mathcal{I} \mid \exists\, \mathsf{x}_1, \ldots, \mathsf{x_n} \in \Delta^\mathcal{D} : (\mathsf{a}, \mathsf{x}_1) \in \mathsf{f}_1{}^\mathcal{I} \wedge \ldots \wedge (\mathsf{a}, \mathsf{x_n}) \in \mathsf{f_n}{}^\mathcal{I} \wedge$ $(\mathsf{x}_1, \ldots, \mathsf{x_n}) \in \mathsf{P}^\mathcal{I}\}$
$\forall\, \mathsf{f}_1, \ldots, \mathsf{f_n} \,.\, \mathsf{P}$	$\{\mathsf{a} \in \Delta^\mathcal{I} \mid \forall\, \mathsf{x}_1, \ldots, \mathsf{x_n} \in \Delta^\mathcal{D} : (\mathsf{a}, \mathsf{x}_1) \in \mathsf{f}_1{}^\mathcal{I} \wedge \ldots \wedge (\mathsf{a}, \mathsf{x_n}) \in \mathsf{f_n}{}^\mathcal{I} \Rightarrow$ $(\mathsf{x}_1, \ldots, \mathsf{x_n}) \in \mathsf{P}^\mathcal{I}\}$
Roles and Features	
R	$\mathsf{R}^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$
f	$\mathsf{f}^\mathcal{I} : \Delta^\mathcal{I} \to \Delta^\mathcal{D}$ (features are partial functions)

$\lVert \cdot \rVert$ denotes the cardinality of a set, and $n, m \in \mathbb{N}$ with $n > 1$, $m > 0$.

(b)

Axioms	
Syntax	Satisfied if
$\mathsf{R} \in \mathcal{T}$	$\mathsf{R}^\mathcal{I} = (\mathsf{R}^\mathcal{I})^+$
$\mathsf{R} \sqsubseteq \mathsf{S}$	$\mathsf{R}^\mathcal{I} \subseteq \mathsf{S}^\mathcal{I}$
$\mathsf{C} \sqsubseteq \mathsf{D}$	$\mathsf{C}^\mathcal{I} \subseteq \mathsf{D}^\mathcal{I}$

(c)

Assertions ($\mathsf{a}, \mathsf{b} \in O$, $\mathsf{x}, \mathsf{x_i} \in O_C$)	
Syntax	Satisfied if
$\mathsf{a} : \mathsf{C}$	$\mathsf{a}^\mathcal{I} \in \mathsf{C}^\mathcal{I}$
$(\mathsf{a}, \mathsf{b}) : \mathsf{R}$	$(\mathsf{a}^\mathcal{I}, \mathsf{b}^\mathcal{I}) \in \mathsf{R}^\mathcal{I}$
$(\mathsf{a}, \mathsf{x}) : \mathsf{f}$	$(\mathsf{a}^\mathcal{I}, \alpha(\mathsf{x})) \in \mathsf{f}^\mathcal{I}$
$(\mathsf{x}_1, \ldots, \mathsf{x_n}) : \mathsf{P}$	$(\alpha(\mathsf{x}_1), \ldots, \alpha(\mathsf{x_n})) \in \mathsf{P}^\mathcal{I}$
$\mathsf{a} = \mathsf{b}$	$\mathsf{a}^\mathcal{I} = \mathsf{b}^\mathcal{I}$
$\mathsf{a} \neq \mathsf{b}$	$\mathsf{a}^\mathcal{I} \neq \mathsf{b}^\mathcal{I}$

Figure 12: Syntax and Semantics of $\mathcal{ALCQHI}_{R^+}(\mathcal{D})^-$.

been shown that $\mathcal{ALCNH}_{R^+}$ extended by a limited form of concrete domains leads to decidable inference problems. This is achieved by disallowing so-called feature chains in $\mathcal{ALCNH}_{R^+}(\mathcal{D})^-$. It is easy to see that the same pragmatic approach can also be applied to very expressive DLs. By analogy to $\mathcal{ALCNH}_{R^+}(\mathcal{D})^-$ the description logic (DL) $\mathcal{ALCQHI}_{R^+}(\mathcal{D})^-$ extends the logic $\mathcal{ALCQHI}_{R^+}$ or $\mathcal{SHIQ}$ [Horrocks et al., 2000b] with concrete domains.

An important property of the language $\mathcal{SHIQ}$ (see above) is that the subsumption hierarchy of the *TBox* part of a *knowledge base* ($\mathcal{T}$,$\mathcal{A}$) is stable w.r.t. additions to the *ABox* part. Such, that multiple *knowledge bases* ($\mathcal{T}$,$\mathcal{A}_1$),...,($\mathcal{T}$,$\mathcal{A}_n$) can reuse computations done so far for a *TBox*, for query answering on any of the *ABoxes* $\mathcal{A}_i$. This is due to stability on the subsumption relationships between concepts, since they depend only on axioms in the TBox $\mathcal{T}$. This property is lost when introducing *nominals*, which are described in the next subsection.

4.1.1 Very Expressive Description Logics

A *nominal* (the letter $\mathcal{O}$ in a language name indicates the presence of nominals) is a singleton concept, syntactically represented as $\{o\}$ and semantically interpreted as $\{\mathrm{o}\}^\mathcal{I} = \{o^\mathcal{I}\}$. Thus, *nominals* stand for concepts with exactly one individual in contrast to a concept which stands for a set of individuals. This allows the use of individuals in concept definitions such as names for specific persons, countries, etc., leading to the situation in which there is no more difference between *TBox* and *ABox*. OWL DL is a language that supports nominals.

$\mathcal{SROIQ}$ [Horrocks et al., 2006] is the most expressive language w.r.t to role statements whose decidability has been proved. On top of $\mathcal{SHIQ}$, $\mathcal{SROIQ}$ allows for more expressivity concerning roles, where besides *TBox* and *ABox*, an *RBox* is introduced to include role statements, allowing for:

1. *Complex role inclusion axioms* of the form $R \circ S \sqsubseteq R$ and $S \circ R \sqsubseteq R$ where R is a role and S is a simple role.

2. *Disjoint roles*

3. *Reflexive, irreflexive and antisymmetric roles*
4. *Negated role assertions*
5. *Universal role*
6. *Local expressivity* to allow concepts of the form $\exists R$.Self

$\mathcal{SROIQ}$ represents the logical basis of OWL 1.1 plus datatypes and datatypes restrictions $\mathcal{SROIQ}(D^+)$. In [Horrocks et al., 2006] a tableaux algorithm is presented, proving that $\mathcal{SROIQ}$ is decidable, still there is no system implementation that supports all features of this language.

4.1.2 Other Decidable Fragments of First-Order Logic

The DL language $\mathcal{EL}$ is a tractable minimal language of special interest for existing learning approaches (see Section 4.4). Being the counterpart of the DL language $\mathcal{FL}_0$, $\mathcal{EL}$ allows full existential quantification ($\exists R.C$) and overcomes the intractability of $\mathcal{FL}_0$ in the presence of acyclic TBoxes. It is known since the 90s [Nebel, 1990], that even for a minimal language like $\mathcal{FL}_0$, considering TBoxes turns intractable, therefore any extension on $\mathcal{FL}_0$ is also intractable. For this reason $\mathcal{EL}$ becomes interesting, and relevant work [Baader et al., 2005] has been done in investigating extensions of it that do not sacrifice its tractability, specially in the presence of general concept inclusion axioms (GCI). A system for the fragment discussed above is CEL [Baader et al., 2006]. Another tractable fragment related to $\mathcal{EL}$ is the DL-Lite fragement, which is implemented in the QuOnto system [Calvanese et al., 2005].

4.1.3 Rules

As observed in previous sections, decidability is a characteristic that should be preserved by ontology languages and which has caused expressivity restrictions. This is one of the reasons why rules are gaining interest as an option to overcome expressivity limitations in DLs. A relevant proposal to extend DL languages (more specifically, the syntactic variant OWL-DL) with Horn-like rules, is the rule language called SWRL (Semantic Web Rule Language). SWRL uses OWL DL or OWL Lite as the underlying DL language to specify a KB.

But the extension of OWL DL with rules is known to be undecidable [Motik et al., 2005], this is due to the fact that decidability in OWL DL restricts the language to axioms that expresses only quasi tree-like structures (we disregard transitive relations). Such a property is lost when adding rules, therefore in order to add rules and still preserve decidability, a subset of SWRL can be used, the so called *DL-safe* rules [Motik et al., 2005].

DL-safe rules are applicable only to explicitly introduced individuals and are formally defined as follows:

Given a set of concept names N_c, a set of abstract and concrete role names $N_{R_a} \cup N_{R_c}$. A DL-atom is of the form C(x) or R(x,y), where $C \subseteq N_C$ and $R \in N_{R_a} \cup N_{R_c}$. A rule r is safe if each of its variables occurs in a non-DL-atom in the rule body. In practice this means that rules are applied to ABox individuals only.

4.2 Standard Inference Services

In the following we define standard inference services for description logics.

A *concept* is called *consistent* (w.r.t. a TBox $\mathcal{T}$) if there exists a model of C (that is also a model of $\mathcal{T}$ and $\mathcal{R}$). An *ABox* $\mathcal{A}$ is *consistent* (w.r.t. a TBox $\mathcal{T}$) if $\mathcal{A}$ has model $\mathcal{I}$ (which is also a model of $\mathcal{T}$). A *knowledge base* ($\mathcal{T}$,$\mathcal{A}$) is called *consistent* if there exists a model for $\mathcal{A}$ which is also a model for $\mathcal{T}$. A concept, ABox, or knowledge base that is not consistent is called *inconsistent*.

A concept D *subsumes* a concept C (w.r.t. a TBox $\mathcal{T}$) if $\mathsf{C}^\mathcal{I} \subseteq \mathsf{D}^\mathcal{I}$ for all interpretations $\mathcal{I}$ (that are models of $\mathcal{T}$). If D subsumes C, then C is said to be *subsumed by* D.

Besides these basic problems, some additional inference services are provided by description logic systems. A basic reasoning service is to compute the subsumption relationship between concept names (i.e. elements from C). This inference is needed to build a hierarchy of concept names w.r.t. specificity. The problem of computing the most-specific concept names mentioned in $\mathcal{T}$ that subsume a certain concept is known as computing the *parents* of a concept. The *children* are the most-general concept names mentioned in $\mathcal{T}$ that are subsumed by a certain concept. We use the name *concept ancestors* (*concept descendants*) for the transitive closure of the parents (children) relation. The computation of the parents and children of every concept name is also called *classification* of the TBox. Another important inference service for practical knowledge representation is to check whether a certain concept name occcurring in a TBox is inconsistent. Usually, inconsistent concept names are the consequence of modeling errors. Checking the consistency of all concept names mentioned in a TBox without computing the parents and children is called a TBox *coherence check*.

If the description logic supports full negation, consistency and subsumption can be mutually reduced to each other since D subsumes C (w.r.t. a TBox $\mathcal{T}$) iff $\mathsf{C} \sqcap \neg\mathsf{D}$ is inconsistent (w.r.t. $\mathcal{T}$) and C is inconsistent (w.r.t. $\mathcal{T}$) iff C is subsumed by $\bot$ (w.r.t. $\mathcal{T}$). Consistency of concepts can be reduced to ABox consistency as follows: A concept C is consistent (w.r.t. a TBox $\mathcal{T}$) iff the ABox $\{\mathsf{a}:\mathsf{C}\}$ is consistent (w.r.t. $\mathcal{T}$).

An individual i is an *instance* of a concept C (w.r.t. a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$) iff $\mathsf{i}^\mathcal{I} \in \mathsf{C}^\mathcal{I}$ for all models $\mathcal{I}$ (of $\mathcal{T}$ and $\mathcal{A}$). For description logics that support full negation for concepts, the instance problem can be reduced to the problem of deciding if the ABox $\mathcal{A} \cup \{\mathsf{i}:\neg\mathsf{C}\}$ is inconsistent (w.r.t. $\mathcal{T}$). This test is also called *instance checking*. The most-specific concept names mentioned in a TBox $\mathcal{T}$ that an individual is an instance of are called the *direct types* of the individual w.r.t. a knowledge base $(\mathcal{T}, \mathcal{A})$. The direct type inference problem can be reduced to subsequent instance problems (see e.g. [Baader et al., 1994] for details).

An ABox $\mathcal{A}'$ is *entailed* by a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ if all models of $\mathcal{T}$ and $\mathcal{A}$ are also models of $\mathcal{A}'$. For ABox entailment we write $\mathcal{T} \cup \mathcal{A} \models \mathcal{A}'$.

TBox inference services are provided by the systems CEL [Baader et al., 2006], Fact++ [Tsarkov and Horrocks, 2006], KAON2 [Hustadt et al., 2004], Pellet [Sirin and Parsia, 2006], QuOnto [Calvanese et al., 2005], and RacerPro [Haarslev and Möller, 2001c]. The latter four systems also support ABox inferences services.

4.3 Retrieval Inference Services

For practical application another set of inference services deals with finding individuals in ABoxes that satisfy certain conditions.

The *retrieval* inference problem is to find all individuals mentioned in an ABox that are instances of a certain concept C. The set of *fillers* of a role R for an individual i w.r.t. a knowledge base $(\mathcal{T}, \mathcal{A})$ is defined as $\{\mathsf{x} \mid (\mathcal{T}, \mathcal{A}) \models (\mathsf{i}, \mathsf{x}):\mathsf{R}\}$ where $(\mathcal{T}, \mathcal{A}) \models ax$ means that all models of $\mathcal{T}$ and $\mathcal{A}$ also satisfy ax. The set of *roles* between two individuals i and j w.r.t. a knowledge base $(\mathcal{T}, \mathcal{A})$ is defined as $\{\mathsf{R} \mid (\mathcal{T}, \mathcal{A}) \models (\mathsf{i}, \mathsf{j}):\mathsf{R}\}$.

In practical system such as RACER, there are some auxiliary queries supported: retrieval of the concept names or individuals mentioned in a knowledge base, retrieval of the set of roles, retrieval of the role parents and children (defined analogously to the concept parents and children, see above), retrieval of the set of individuals in the domain and in the range of a role, etc. As a distinguishing feature to other systems, which is important for many applications, we would like to emphasize that RACER supports multiple TBoxes and ABoxes. Assertions can be added to ABoxes after queries have been answered. In addition, RACER also provides support for retraction of assertions in particular ABoxes.

In addition to the basic inference service *concept-based instance retrieval* in practical applications more expressive query languages are required.

A query consists of a *head* and a *body*. The head lists variables for which the user would like to compute bindings. The body consists of query atoms (see below) in which all variables from the head must be mentioned. If the body contains additional variables, they are seen as existentially quantified. A query answer is a set of tuples representing bindings for variables mentioned in the head. A query is written $\{(X_1, \ldots, X_n) \mid atom_1, \ldots, atom_2\}$.

Query atoms can be *concept* query atoms ($C(X)$), *role* query atoms ($R(X, Y)$), *same-as* query atoms ($X = Y$) as well as so-called *concrete domain* query atoms. The latter are introduced to provide support for querying the concrete domain part of a knowledge base and will not be covered in detail here. Complex queries are built from query atoms using boolean constructs for conjunction (indicated with comma), union ($\vee$) and negation ($\neg$) (note that the latter refer to atom negation not concept negation and, for instance, negation as failure semantics is assumed in [Wessel and Möller, 2005]). In addition, a *projection* operator is supported in order to reduce the dimensionality of an intermediate tuple set. This operator is particularly important in combination with negation (complement). For details see [Wessel and Möller, 2005].

Answering queries in DL systems goes beyond query answering in relational databases. In databases, query answering amounts to model checking (a database instance is seen as a model of the conceptual schema). Query answering w.r.t. TBoxes and ABoxes must take all models into account, and thus requires deduction. The aim is to define expressive but decidable query languages. Well known classes of queries such as *conjunctive queries* and *unions of conjunctive queries* are topics of current investigations in this context.

In the literature (e.g., [Horrocks et al., 2000a; Glimm et al., 2007; Wessel and Möller, 2006]), two different semantics for these kinds of queries are discussed. In *standard* conjunctive queries, variables are bound to (possibly anonymous) domain objects. In so-called *grounded* conjunctive queries, variables are bound to named domain objects (object constants). However, in grounded conjunctive queries the standard semantics can be obtained for so-called tree-shape queries by using existential restrictions in query atoms.

ABox entailment can be reduced to query answering. An ABox $\mathcal{A}'$ is entailed by a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ if for all assertions α in $\mathcal{A}'$ it holds that the boolean query $\{() \mid \alpha\}$ returns *true*.

4.4 Nonstandard Inference Services

In addition to standard inference services and retrieval inference services, recently another set of inference problems have been defined, decision problems have been shown to be decidable, and practical inference algorithms as well as system implementations have been developed (cf., [Turhan and Kissig, 2004]). These services are known as non-standard inference services. Non-standard inference services are useful for building ontologies from examples (learning or bottom-up approach for constructing ontologies). Note that depending on the ontology language, a solution for the problems mentioned below need not necessarily exist. Furthermore, algorithms for the problems are known only for non-expressive description logics in most cases.

- Least-common subsumers (LCS) [Baader and Küsters, 1998; Baader et al., 1999b, 2004]
  A least-common subsumer represents the commonalities between a set of concepts. For a given set of concepts C_1,...,C_n their least-common subsumer E is defined as follows:
  1. $C_i \sqsubseteq E$ for all $i =$ 1,...,n and
  2. If E' is a concept satisfying $C_i \sqsubseteq E'$ for all $i =$ 1,...,n then $E \sqsubseteq E'$.
- Most-specific concept (MSC) [Borgida and Küsters, 1999]
  Given a finite set of assertions in *ABox* $\mathcal{A}$ of the form $C(a)$ or $r(a, b)$, where C is a concept and r is a role name, the concept E represents the most-specific concept of the individual a in *ABox* $\mathcal{A}$ if it satisfies:
  1. $\mathcal{A} \models E(a)$, and
  2. If E' is a concept satisfying $\mathcal{A} \models E'(a)$ then $E \sqsubseteq E'$.
- Rewriting w.r.t. terminologies [Baader et al., 2000]
  The aim is to rewrite a given concept definition C w.r.t. to a terminology T into an equivalent concept C', such that redundancies are removed and its length is reduced. The resulting concept definition C' is easier to understand by humans than the previous definition C. In an ideal world the objective is a *minimal rewriting*, such that given an ordering $\preceq$ on concept definitions, a rewriting C' is *minimal* iff there does not exist a rewriting C'' such that $C'' \preceq C'$. Obtaining the *minimal* concept rewriting is computationally non-trivial, therefore it is sufficient to compute a *small* but not *minimal* definition.
- Approximation [Brandt et al., 2001, 2002]
  Given a concept description written in a more expressive DL, the aim is to translate the concept into a less expressive DL with minimal loss of information. Thus, given two DL

languages, e.g. $\mathcal{L}_1 = \mathcal{ALC}$ and $\mathcal{L}_2 = \mathcal{ALE}$ an *approximation* of a concept description C in $\mathcal{L}_1$ to a concept description in $\mathcal{L}_2$ is a concept description D, such that $C \sqsubseteq D$ and D is minimal w.r.t. subsumption.

- Difference [Brandt et al., 2002]
  The difference E of two concepts C and D is defined as follows: $E \sqcap D \equiv C$.

- Matching [Baader et al., 1999a; Baader and Küsters, 1999a,b, 2000, 2001; Baader et al., 2001]
  The aim is to locate similar concepts in one terminology to find redundancies, or similar concepts in different ontologies to help in their integration. Matching is based on concept patterns. A pattern substitutes the concept terms that appear in a concept description with *concept variables*, such *concept variables* can represent any arbitrary concept description. Thus, matching is based on finding concepts that have similar patterns. This is formally defined as follows: given a concept pattern C and a concept term D, find a substitution σ of the concept variables in C with concept terms such that $\sigma(C) \equiv$ D.

- Explanation (axiom pinpointing) [Kalyanpur et al., 2006b,a]
  To support ontology engineers in finding the causes of errors or inconsistencies in an ontology, axiom pinpointing aims to identify those parts of the ontology causing an unintended ramification (e.g., an ABox inconsistency). In this context, the explanation service should provide arguments that prove why the previously pinpointed axioms hold for the unintended ramification.

- Similarity [Lutz et al., 2003; Borgida et al., 2005]
  Similarity of concepts and individuals has been the subject of recent investigations, in particular for learning applications. The ideas is that individuals are similar if they have similar direct types and similar role fillers. The problem is, however, that similarity measures must be based on the models of concepts or ABoxes rather than on the syntactic structure of concept terms or ABox assertions. For instance, the TBox and ABox shown below enforces some constrains that will have to be considered for similarity assessment, but which are not syntactically apparent.

$$\begin{array}{rcl} X & \equiv & (\forall r.\neg C) \sqcup (\exists_{\leq 1} r) \\ (i,j) & : & r \\ j & : & C \\ i & : & X \\ (i,k) & : & r \\ (k,l) & : & r \\ k & : & D \\ l & : & E \end{array}$$

  It can be seen that j is not only an instance of C but also an instance of D, and j has two role fillers, namely k and l.

  Different notions of similarity are defined in the literature and different measures have been proposed [Linckels and Meinel, 2005; Janowicz, 2005; d'Amato et al., 2006].

- Abduction [Elsenbroich et al., 2006]
  Abduction aims to derive a set of explanations Δ for a given set of assertions Γ such that Δ is consistent w.r.t. to the ontology $(\mathcal{T}, \mathcal{A})$ and satisfies:

  1. $\mathcal{T} \cup \mathcal{A} \cup \Delta \models \Gamma$ and
  2. If Δ' is an ABox satisfying $\mathcal{T} \cup \mathcal{A} \cup \Delta' \models \Gamma$, then $\Delta' \models \Delta$ (Δ is least specific)

4.5 Examples of interpretation in BOEMIE

Ontologies in a description logic framework are seen as a tuple consisting of a TBox and an ABox. In order to populate an ontology, in technical terms the ABox part of the ontology is extended with some new assertions describing individuals and their relations. These descriptions are derived by media interpretation processes using the ontology (we assume the ontology axioms are denoted in a set Σ and we do not distinguish between TBox and ABox).

Interpretation processes are set up for different modalities, still images, videos, audio data, and texts. Using an example interpretation for still images, in this section we discuss the assumed output of the interpretation process. The output is a symbolic description represented as an ABox. This ABox is the result of an abduction process (see [Shanahan, 2005] for a general introduction). In this process a solution for the following equation is computed: $\Sigma \cup \Delta \models \Gamma$. The solution Δ must satisfy certain side conditions (see Section 4.4).

Figure 13: Still image displaying a pole vault event.

In Figure 13 an example from the athletic domain is presented. Assuming it is possible to detect a horizontal bar bar_1, a jumper $jumper_1$, and a pole $pole_1$ by image analysis processes the output of the analysis phase is represented as an ABox Γ. Assertions for the individuals and (some of) their relations detected by analyzing Figure 13 are shown in Figure 14.

$$
\begin{array}{rcl}
pole_1 & : & Pole \\
jumper_1 & : & Jumper \\
bar_1 & : & Horizontal\_Bar \\
(bar_1, jumper_1) & : & near \\
(jumper_1, pole_1) & : & touches
\end{array}
$$

Figure 14: An ABox Γ representing the result of the image analysis phase.

In order to continue the interpretation example, we assume that the ontology contains the axioms shown in Figure 15 (the ABox of the ontology is assumed to be empty). For some purposes, description logics are not expressive enough. Thus, we assume the ontology is extended with so-called DL-safe rules (rules that are applied to ABox individuals only, see Section 4.1.3). In Figure 16 a set of rules for the athletics example is specified. Note that the spatial constraints *touches* and *near* for the parts of a $Pole\_Vault$ event (or a $High\_Jump$ event) are not imposed by the TBox in Figure 15. Thus, rules are used to represent this additional knowledge. Since spatial relations depend on the specific "subphases" of the events, corresponding clauses are included on the right-hand sides of the rules (being aware of the possibility to model phases explicitly as individuals, here we use phase predicates for brevity). In the following we assume that the rules are part of the TBox Σ.

We assume that the agent interpreting the image decides that spatial relations between certain objects are not arbitrary, i.e., the agent would like to have an explanation about why is it the case

$$
\begin{array}{rcl}
Man & \sqsubseteq & Person \\
Woman & \sqsubseteq & Person \\
Man & \sqsubseteq & \neg Woman \\
Athlete & \equiv & Person \sqcap \exists hasProfession.Sport \\
Foam\_Mat & \sqsubseteq & SportEquipment \\
Pole & \sqsubseteq & SportEquipment \\
Javelin & \sqsubseteq & SportEquipment \\
Horizontal\_bar & \sqsubseteq & SportEquipment \\
Jumping\_Event & \sqsubseteq & Event \sqcap \exists hasPart.Jumper \\
Pole\_Vault & \sqsubseteq & Jumping\_Event \sqcap \\
 & & \exists hasPart.Pole \sqcap \\
 & & \exists hasPart.Horizontal\_Bar \sqcap \\
 & & \exists hasPart.Foam\_Mat \\
High\_Jump & \sqsubseteq & Jumping\_Event \sqcap \\
 & & \exists hasPart.Horizontal\_Bar \sqcap \\
 & & \exists hasPart.Foam\_Mat \\
PVStartPhase & \sqsubseteq & \top \\
PVEndStartPhase & \sqsubseteq & \top \\
HJJumpPhase & \sqsubseteq & \top \\
 & \ldots &
\end{array}
$$

Figure 15: An tiny example TBox Σ for the athletics domain

$$
\begin{array}{rcl}
touches(Y, Z) & \leftarrow & Pole\_Vault(X), \\
 & & PVStartPhase(X), \\
 & & hasPart(X, Y), Jumper(Y), \\
 & & hasPart(X, Z), Pole(Z). \\
near(Y, Z) & \leftarrow & Pole\_Vault(X), \\
 & & PVEndStartPhase(X), \\
 & & hasPart(X, Y), Horizontal\_Bar(Y), \\
 & & hasPart(X, Z), Jumper(Z). \\
near(Y, Z) & \leftarrow & High\_Jump(X), \\
 & & HJJumpPhase(X), \\
 & & hasPart(X, Y), Horizontal\_Bar(Y), \\
 & & hasPart(X, Z), Jumper(Z). \\
 & \ldots &
\end{array}
$$

Figure 16: Additional restrictions for $Pole\_Vault$ and $High\_Jump$ in the form of rules.

that a jumper touches a pole and is near a horizontal bar. Explanations can be seen as the result of the image interpretation process. As mentioned above the idea is to use the abduction inference service for deriving these kinds of explanations (in the sense of interpretations). In our scenario we slightly modify the abduction equation by taking into consideration that initially the ABox does not need to be empty. Thus, we divide Γ (see Figure 14) into a part Γ_2 that the agent would like to have explained, and a part that the interpretation agent takes for granted (Γ_1). In our case Γ_2 is $\{(bar_1, jumper_1) : near, \quad (jumper_1, pole_1) : touches\}$ and Γ_1 is $\{pole_1 : Pole, \quad jumper_1 : Jumper, \quad bar_1 : Horizontal\_Bar\}$.

Coming back to the abduction problem specified above, we need solution(s) for the equation $\Sigma \cup \Delta \cup \Gamma_1 \models \Gamma_2$. In other words, given the background ontology Σ from Figures 15 and 16, the query as derived from Γ_2 should return *true*:

$$
\begin{array}{rl}
Q_1 := \{() \mid & near(bar_1, jumper_1), \\
 & touches(jumper_1, pole_1)\}
\end{array}
$$

Obviously, this is not the case if Δ is empty (note that the ABox part of the ontology Σ is also empty). In order to see how an appropriate Δ could be derived, let us have a look at the rules in

Figure 16. In particular, let us focus on the rules for $Pole\_Vault$ first. If we apply the rules to the query in a backward chaining way (i.e., from left to right) and unify corresponding terms we get corresponding variable bindings for Y and Z. The "unbound" variable X of the corresponding rules are instantiated with fresh individuals (e.g., pv_1 and pv_2). For some reason the agent might assume, only one event is involved (i.e., only pv_1 is used). Then, a possible solution Δ for the abduction equation can be derived. Δ is shown in Figure 17.

$$
\begin{array}{rcl}
pv_1 & : & Pole\_Vault \\
pv_1 & : & PVStartPhase \\
pv_1 & : & PVEndStartPhase \\
(pv_1, jumper_1) & : & hasPart \\
(pv_1, bar_1) & : & hasPart \\
(pv_1, pole_1) & : & hasPart
\end{array}
$$

Figure 17: One possible solution of the abduction equation.

Considering the GCIs involving $Pole\_Vault$ in the TBox shown in Figure 15 it becomes apparent that for a pole vault there also exists a foam mat which is not found by the image analysis module: Maybe it is not visible or the analysis just could not detect it. In the latter situation, one could somehow adapt the image analysis processes and start a feedback loop. This feedback from the image interpretation module (high level) to the image analysis module (low-level) is subject to ongoing research and will be covered elsewhere. The assertions concerning the relation $hasPart$ and the phases derived by the rule are included in the interpretation result. Thus, the output of the interpretation phase in our example is the ABox shown in Figure 18.

$$
\begin{array}{rcl}
pole_1 & : & Pole \\
jumper_1 & : & Jumper \\
bar_1 & : & Horizontal\_Bar \\
(bar_1, jumper_1) & : & near \\
(jumper_1, pole_1) & : & touches \\
pv_1 & : & Pole\_Vault \\
pv_1 & : & PVStartPhase \\
pv_1 & : & PVEndStartPhase \\
(pv_1, jumper_1) & : & hasPart \\
(pv_1, bar_1) & : & hasPart \\
(pv_1, pole_1) & : & hasPart
\end{array}
$$

Figure 18: An ABox representing the result of the image interpretation phase.

The example discussed in this document covers the interpretation of still images. It is necessary, however, to keep in mind that each media object might consist of multiple modalities, each of which will be the basis of modality-specific interpretation results (ABoxes). In order to provide for an integrated representation of the interpretation of media objects as a whole, these modality-specific interpretation results must be appropriately integrated. A cornerstone of this integration process will be to determine which modality-specific names refer to the same domain object. This will be discussed in the following sections. In the following, both modality-specific and media-specific ABoxes will be called interpretation ABoxes. Due to the context, ambiguities should not arise.

So far we have discussed an example where there is one unique explanation (and, hence, one unique interpretation). However, this need not necessarily be the case. In Figure 19 an example is presented that might lead to two different interpretations. For the example we assume the ABox in Figure 20 is produced by the image analysis component.

For the interpretation process we assume the same ontology as above. It is easy to see that we get two explanations by the abduction process (see Figures 21 and 22). Note that each interpretation process will generate new names which might refer to the same domain object. For query answering we assume that each individual is somehow associated with its ABox and each ABox is associated with its media objects (the techniques used to implement these associations are described elsewhere). A query $\{x \mid Pole\_Vault(x)\}$ applied to the ABoxes in Figures 21 and 22

Ukraine's Andrey Sokolovskiy clears 2.38m in Rome (Getty Images)

Figure 19: Image displaying a snapshot of a high jump or pole vault (where the pole it outside the image).

$$\begin{array}{rcl} bar_2 & : & Horizontal\_Bar \\ jumper_2 & : & Jumper \\ (bar_2, jumper_2) & : & near \end{array}$$

Figure 20: An ABox Γ representing the result of the analysis of the image in Figure 19.

returns $\{(pv_2)\}$. From this solution the corresponding ABox and, in turn, the media object is derived. Similar results are obtained for $High\_Jump$ queries.

Continuing the example it might be the case that for some images the ontology does not contain relevant axioms or rules. In this case, the interpretation result, i.e., the result of solving the abduction problem $\Sigma \cup \Delta \cup \Gamma_1 \models \Gamma_2$ will be degenerated because, due to missing axioms or rules in Σ, Δ must necessarily be equal to Γ_2 in order to solve the equation. As an example of such a situation we can discuss an interpretation of Figures 13 or 19 without the rules from Figure 16 and the GCIs for $Pole\_Vault$ and $High\_Jump$ in Figure 15. The degenerated interpretation result is shown (as Γ) in Figure 14. If a query such as $\{(x) \mid Pole\_Vault(x) \vee High\_Jump(x)\}$ is specified, the corresponding image will not be found.

$$\begin{array}{rcl} jumper_2 & : & Jumper \\ bar_2 & : & Horizontal\_Bar \\ (bar_2, jumper_2) & : & near \\ hj_2 & : & High\_jump \\ hj_2 & : & HJJumpPhase \\ (hj_2, jumper_2) & : & hasPart \\ (hj_2, bar_2) & : & hasPart \end{array}$$

Figure 21: An ABox representing the first result of the image interpretation process.

In the following chapter we will discuss how, for instance, the GCIs in Figure 15 can be derived. Then we will investigate how an abstraction of the rules in form of GCIs are computed. Note that rules are applied to ABox individuals only (which no problem for the abduction process). For a more general use of the ontologies it is beneficial to approximate the rules also as GCIs which impose restrictions on *all* domain objects.

$$
\begin{array}{rcl}
jumper_2 & : & Jumper \\
bar_2 & : & Horizontal\_Bar \\
(bar_2, jumper_2) & : & near \\
pv_2 & : & Pole\_Vault \\
pv_2 & : & PVEndStartPhase \\
(pv_2, jumper_2) & : & hasPart \\
(pv_2, bar_2) & : & hasPart
\end{array}
$$

Figure 22: An ABox representing the second result of the image interpretation process.

5 ONTOLOGY POPULATION

Ontology population is the process of inserting concept instances into an existing ontology. The first activity of ontology evolution, which leads to either population or enrichment, is evolution pattern selection. Evolution pattern selection is responsible for selecting which evolution pattern (one of P1 – P4) will be followed, based mainly on the number of explanations (or their lack of) produced for the incoming ABox. Ontology population is triggered through the selection of either P1 or P2 evolution patterns, meaning that either a single (evolution pattern P1) or more (evolution pattern P2) explanations (HLCis) have been found during the interpretation activity to explain the extraction results for a single multimedia document. Having as input a single HLCi or a set of HLCis, the population activity is responsible for performing the following tasks:

- *Instance matching*: The first task of the population activity is the identification of similar instances from the domain ontology for each HLCi in the set of explanations. Having a single HLCi as input, instance matching is expected to return a set of instances (ABoxes) that populate the HLC and are similar to the incoming ABox. Each returned matching instance is also expected to have a similarity figure, which measures the similarity of the two ABoxes (HLCis). The results of instance matching can be used to group instances that represent the same real object or event (as ABoxes that are similar are assumed to represent the same real object or event) and possibly help in disambiguating multiple explanations in the case of evolution pattern P2 (ABox refinement).
- *Instance grouping*: This task is responsible for grouping all the instances that represent the same real object or event (exploiting the results of the instance matching task). Every incoming HLCi has been matched with a set of other instances that populate this HLC through instance matching. Instance grouping is responsible to decide which of these matching instances will be kept and grouped together with the original incoming ABox to form a group that represents the same real object or event.
- *ABox refinement* (evolution pattern P2 only): In case of multiple explanations (evolution pattern P2), the most suitable explanation is selected by exploiting the results of the instance grouping task. Assertions related to the rest of the explanations are removed from the ABox, thus leading to a refined version of it.
- *ABox validation*: This task performs consistency checking, to detect possible inconsistencies due to the additions that will be performed to the ontology. Two tasks must be validated: the addition of the incoming ABox (i.e., the one that originally triggered the population activity) into the ontology and the addition of a new instance of a grouping concept or the modification of an existing one. Both type of validation can be performed through standard inference services (see Section 4).
- *ABox assimilation*: The final task is responsible for performing the needed changes in the ontology (by creating all instances/relations in all ontological modules), in order to incorporate the information in the new ABox into the ontology.

The input to the population activity for a single multimedia document is:

- The ABox that the information extraction toolkit produced for the multimedia document. This ABox is expected to contain:

  - A set of mid-level concept instances (MLCis).
  - A set of relations between the various MLCis.
  - A set of HLCis.

- The evolution pattern (P1 or P2), selected by the interpretation activity.

In the following sections, the various tasks of the population activity will be described, along with a complete example showing the various tasks performed in the evolution patterns P1 and P2, respectively.

5.1 Single concept explanation

This section explains the tasks involved when evolution pattern P1 has been selected during the evolution pattern selection activity. The selection of evolution pattern P1 occurs when the ABox of a multimedia document contains a unique HLCi that explains the document. This explanation HLCi will not be the only HLCi in the ABox but will be the only HLCi that explains the document as a whole, possibly involving other HLCis in this explanation. Having identified this HLCi in the ABox, the following tasks must be performed.

5.1.1 Instance matching

The goal of instance matching (or individual matching) is to evaluate the degree of similarity among different individuals for which assertions provide restrictions in one or more ABoxes defined with respect to the same TBox. For matching, the similarity of role filler values as well as the similarity of their direct types (or MSCs) will have to be considered. In Section 4.4 similarity has been introduced as a non-standard inference service, and with some examples we have demonstrated the role of reasoning for determining role fillers and, e.g., direct types of individuals. In order to simplify the discussion, here we only consider "structural" matching, i.e., we consider cases where similarity can be defined with respect to the structure of Abox assertions. The measures defined will have to be extended to semantics-based matching.

In order to perform the matching we distinguish among instances of high-level aggregate concepts (HLCi) and instances of mid-level simple concepts (MLCi). A HLCi description is a portion of an ABox that contains all the assertions involving at least one individual of an aggregate concept (HLC), while MLCi is a portion of an ABox that contains all the assertions involving only individuals of simple concepts (MLC). With respect to the example of Figure 18, we have three mid-level concepts (i.e., $Pole$, $Horizontal\_Bar$, and $Jumper$) and one high-level concept (i.e., $Pole\_Vault$) [3]. According to this, we distinguish three MLCis and one HLCi in the ABox of Figure 18, as shown in Figure 23.

$MLCi_{pole_1}$		
$pole_1$	:	$Pole$
$(jumper_1, pole_1)$	:	$touches$

$MLCi_{jumper_1}$		
$jumper_1$	:	$Jumper$
$(bar_1, jumper_1)$	:	$near$
$(jumper_1, pole_1)$	:	$touches$

$MLCi_{bar_1}$		
bar_1	:	$Horizontal\_Bar$
$(bar_1, jumper_1)$	:	$near$

$HLCi_{pv_1}$		
pv_1	:	$Pole\_Vault$
$(pv_1, jumper_1)$	:	$hasPart$
(pv_1, bar_1)	:	$hasPart$
$(pv_1, pole_1)$	:	$hasPart$

Figure 23: Example of mid-level concept instances and a high-level concept instance.

[3] Note that, in the example, $Jumper$ is considered as a simple component of the image (i.e., only low-level features are provided for it). If the extraction provides more information about the jumper as an athlete (e.g., name, nationality), it is considered an HLCi in the instance matching process

In order to illustrate the instance matching procedure, we introduce a second example ABox which describes a similar image where the jumper does not touch the pole. The corresponding instances are shown in Figure 24. The idea behind instance matching is that the similarity between

$MLCi_{pole_2}$

$pole_2$	:	$Pole$

$MLCi_{jumper_2}$

$jumper_2$	:	$Jumper$
$(bar_2, jumper_2)$	:	$near$

$MLCi_{bar_2}$

bar_2	:	$Horizontal\_Bar$
$(bar_2, jumper_2)$	:	$near$

$HLCi_{pv_2}$

pv_2	:	$Pole\_Vault$
$(pv_2, jumper_2)$	:	$hasPart$
(pv_2, bar_2)	:	$hasPart$
$(pv_2, pole_2)$	:	$hasPart$

Figure 24: Example of instances extracted from another ABox.

the two ABoxes depends on the similarity among the instances that comprise them. Instance similarity is evaluated by taking into account the objects that are mentioned in the assertions that compose each instance.

Matching of MLCi. Given two MLC instances $MLCi_1$ and $MLCi_2$, we evaluate their degree of similarity on the basis of the number of matching assertions that compose the two instances. Given two assertions of the form $a : A$ and $a' : A'$, they match if:

1. $A \equiv A'$
2. $d(a, a') \geq t$

where, $d(a, a')$ denotes the measure of data similarity between a and a', while t denotes a threshold that is used in order to discard data with a low level of similarity. In case of role assertions of the form $(a, b) : A$ and $(a', b') : A'$, they match if:

1. $A \equiv A'$
2. $d(a, a') \geq t \wedge d(b, b') \geq t$

The data similarity value $d(a, a')$ is a value in the range $[0, 1]$, and is evaluated by exploiting specific matching functionalities depending on the datatype associated with a and a'. When data represent low level features (e.g., image sections, audio stream), specific matching functions [Farag and Abdel-Wahab, 2003; Ding et al., 2004] have been proposed and can be used in order to evaluate $d(a, a')$. When data are strings, string matching facilities [Stoilos et al., 2005; Cohen et al., 2003] are used. For other basic datatypes (e.g., numbers, dates), specific operators [Qiang et al., 2003] can be defined. Moreover, instance matching relies on lookup tables containing pre-defined rules for data compatibility, that can be defined and maintained by the ontology expert, in order to control the kind of data values that should be considered equivalent in the instance matching process.

Finally, the comprehensive instance similarity $\mathcal{I}(MLCi_1, MLCi_2)$ between two mid-level instances $MLCi_1$ and $MLCi_2$ featuring the same mid-level concept is evaluated by means of a comprehensive matching measure. An example [4] of such a measure is given by the following formula [Castano et al., 1998].

$$\mathcal{I}(MLCi_1, MLCi_2) = \frac{\mid a \in \{A \cap A'\} \mid}{\mid a \in \{A \cup A'\} \mid} \tag{1}$$

where $\mid a \in \{A \cap A'\} \mid$ denotes the number of matching assertions in A and A', while $\mid a \in \{A \cup A'\} \mid$ denotes the total number of assertions in A and A'.

[4] The matching service in BOEMIE provides also other metrics that can be used to evaluate the comprehensive instance matching value.

With respect to the example, we assume that:

$$d(pole_1, pole_2) \geq t$$

$$d(jumper_1, jumper_2) \geq t$$

$$d(bar_1, bar_2) \geq t$$

Thus, by applying (1), we obtain the following values of matching:

$$\mathcal{I}(MLCi_{pole_1}, MLCi_{pole_2}) = 0.67$$

$$\mathcal{I}(MLCi_{jumper_1}, MLCi_{jumper_2}) = 0.8$$

$$\mathcal{I}(MLCi_{bar_1}, MLCi_{bar_2}) = 1.0$$

Note the different MLCis of the example, even if they refers to matching objects, state different facts about these objects. For example, $MLCi_{pole_1}$ states that there is a pole which is touched by a jumper, while $MLCi_{pole_2}$ states only that there is a pole in the image. For this reason, even if the two poles are considered to be similar, the two MLCis are not.

Matching of HLCi. When two aggregate concept instances $HLCi_1$ and $HLCi_2$ are matched, we evaluate their similarity on the bases of the similarity among their components. In other terms, the matching procedure is the same as in the case of MLCi but we do not consider the objects that are classified as HLCis. With respect to the example, we do not have a matching value between pv_1 and pv_2, because they are instances of the high-level concept $Pole\_Vault$. When we match role assertions of the form $(h, b) : A$ and $(h', b') : A'$, where h and h' are objects classified as high level concept instances, they match if:

1. $A \equiv A'$
2. $d(b, b') \geq t$

Finally, we use (1) in order to evaluate the comprehensive value of matching between $HLCi_1$ and $HLCi_2$, that is:

$$\mathcal{I}(HLCi_{pv_1}, HLCi_{pv_2}) = 1.0$$

Coming back to the ABoxes of the example, we can conclude that the two resources describe the same (or very similar) pole vault event, that is a pole vault event composed by the same (or very similar) objects (i.e., $pole_1/pole_2$, $jumper_1/jumper_2$, bar_1/bar_2). Moreover, also the horizontal bars in the resources are the same one (or very similar). In contrast, even if the poles and the jumpers in the two resources are similar, the assertions about them are different, because $jumper_1$ touches $pole_1$, while there is no relation between $jumper_2$ and $pole_2$. A degree of similarity between the two ABoxes can be evaluated by setting a threshold t and by applying (1) to the two ABoxes, as follows:

$$\mathcal{I}(ABox_1, ABox_2) = \frac{\mid x \in \{ABox_1 \cap ABox_2\} \mid}{\mid x \in \{ABox_1 \cup ABox_2\} \mid}$$

where $\mid x \in \{ABox_1 \cap ABox_2\} \mid$ denotes the number of matching instances in the two ABoxes, while $\mid x \in \{ABox_1 \cup ABox_2\} \mid$ denotes the total number of instances in the two ABoxes. In the case of the example, by setting a threshold $t = 0.8$, we obtain:

$$\mathcal{I}(ABox_1, ABox_2) = \frac{6}{8} = 0.75$$

The similarity value calculated between the two ABoxes is due to the fact that the two images represent the same (or similar) objects but two different situations. For instance it could be the case of two images of the same jumping event, but taken of two different moments. In fact, in the first image, the jumper touches the pole, while in the second there is no relation between the same (or similar) jumper and the same (or similar) pole.

5.1.2 Instance grouping

The instance matching task receives a single HLCi explaining the document, and returns a set of ABoxes (already in the ontology as instances of the HLCi). If this returned set is empty (i.e., no matching instances have been found), then instance grouping creates a new group, that contains the ABox description of the HLCi. In the case that the returned set has one or more elements, then instance grouping must decide which of these instances will form a group with the ABox instance, meaning that they refer to the same real object or event.

The instance among similar instances returned by the matching process can exploit either intrinsic information (such as the similarity measure, which must exceed a threshold) or contextual evidence (such as the presence of other unexplained HLCis/MLCis/relations in the two involved documents). All instances that have been selected as representing the same object or event will populate the ontology by remaining as separate instances (i.e., they are not merged into a single instance). In addition, an instance of a concept that can group/index all these instances that represent the same object or event is added to the domain ontology, which allows the easy retrieval of all instances in the group.

5.1.3 ABox validation

This task performs consistency checking, by trying to detect possible inconsistencies among instances that represent the same real objects or events. In order to perform consistency checking, an ABox corresponding to all instances of a formed cluster is constructed and its consistency with respect to the domain ontology is examined through the standard inference services (Section 4).

5.1.4 ABox assimilation

This task is responsible for integrating the ABox and the grouping information in the ontology. All elements that comprise the ABox are inserted to the corresponding ontological modules. For example, MLCis/low-level features populate the multimedia content and descriptor ontology, relations involving only MLCis populate the spatiotemporal ontology, modality-specific HLCis populate the modality elements ontology and finally the remaining HLCis populate the domain ontology. For HLCis that populate the domain ontology, if grouping information has been generated and the group contains two or more elements, then an instance of a grouping concept is additionally inserted into the ontology.

5.2 Multiple concept explanation

This section explains the tasks involved when evolution pattern P2 has been selected during the evolution pattern selection activity. The selection of P2 occurs when two or more HLCis that explain the document have been found through interpretation. The tasks performed during evolution pattern P2 are the same as in evolution pattern P1, with the addition of the ABox refinement task after the instance grouping task.

For each HLCi explaining the document, instance matching and instance grouping are applied, as in P1. Once instance matching and instances grouping have been performed on all HLCis explaining the instance, then ABox refinement is invoked in order to select the HLCi that better explains the instance. If ABox refinement succeeds in selecting a single HLCi, then all other HLCis are rejected. In case ABox refinement is unable to select a single HLCi, then a single HLCi is selected with the help of the ontology expert. Once a single HLCi has been selected (either automatically or with the help of the ontology expert) then population proceeds as as in evolution pattern P1: the other HLCis are rejected (leading to a refined version of the ABox) and the ABox validation and assimilation tasks are invoked to process the selected HLCi.

5.2.1 Instance matching and ABox refinement

As described above, in the evolution pattern P2, we have multiple explanations of the same resource. This means that we have more than one ABox describing a resource, as seen in the example. The ABox refinement task is responsible for selecting a single HLCi that better describes the incoming instance out of the set of possible explanations. The input to the ABox refinement task is a set of HLCis, each of them associated with a group of similar instances, acquired through the instance matching and instances grouping tasks. In order to select a single HLCi, the instances group of an HLCi is compared with all other groups of all remaining HLCis. Intrinsic information of a group is primarily used for comparisons, such as the number of similar instances that participate in a group or its coherence expressed through the similarity values among the group instances. If these criteria can lead to the selection of a single HLCi, then non-selected HLCis are automatically discarded, otherwise, the incoming instances are presented to the ontology expert. The ontology expert can select a single HLCi as explaining better the document, or discard all HLCis, leaving the document unexplained. If the ontology expert selects a single HLCi, then all remaining HLCis are discarded. If no HLCi is selected by the ontology expert, then all explanations are discarded, making the document an unexplained one. The population activity stops processing the document and this unexplained document is returned to the evolution pattern selection activity, which in turn will lead the unexplained document to the ontology enrichment activity, through the selection of evolution pattern P3.

In other words, the idea behind ABox refinement is that, it is possible to exploit the ABoxes already in the ontology with are unambiguously annotated with a single aggregate concept. In particular, we exploit instance matching to evaluate the degree of affinity between the new ABoxes and the ABoxes in the ontology. In particular, given multimedia resource r and a set $A = \{a_1, a_2, \ldots, a_n\}$ of ABoxes that potentially describe r, we match each $a_i \in A$ against the ABoxes in the ontology, by selecting the ABox a_k in the ontology that best matches a_i, that is, the most similar one. The idea is to find resources similar to r already described in the ontology in order to exploit them for choosing the best description of r among the ones provided in A. As an example of this task, consider the ABoxes described in Figures 21 (referred as a_1) and 22 (referred as a_2), and suppose that the ontology ABox constrains the set of instances a_3 described in Figure 25.

$$\begin{array}{rcl} jumper_2 & : & Jumper \\ bar_3 & : & Horizontal\_Bar \\ (bar_3, jumper_2) & : & near \\ hj_3 & : & High\_jump \\ (hj_3, jumper_2) & : & hasPart \\ (hj_3, bar_3) & : & hasPart \end{array}$$

Figure 25: An Abox already stored in the ontology.

We execute instance matching (with a threshold of 0.5) between a_1 and a_3 and between a_2 and a_3, respectively. Since the jumper is the same in the three ABoxes, but only a_1 and a_3 describe high jump events, we obtain:

$$\mathcal{I}(a_1, a_3) = 0.67$$

$$\mathcal{I}(a_2, a_3) = 0.34$$

where the second result is discarded because it is lower than the threshold. The instance similarity between a_1 and a_3 is then submitted to the ontology expert, since it suggest that the resource could be a description of an high jump event more than a pole vault event. On the basis of the suggestion, the ontology expert can chose to refine the ABoxes produced by the interpretation activity. In particular, a_2 is eliminated while a_1 in maintained as the correct description of the resource r.

5.2.2 Example: pole vault or high jump

For the purposes of this example, let's assume that the information extraction toolkit has processed a multimedia document, containing text and still images, originating from the URL http://www.iaaf.org/-news/Kind=2/newsId=33486.html. An overview of this multimedia document is presented in Figure 26:

Let's also assume that the information extraction toolkit was able to analyze the first image at the top of the news item, and the textual passage of the main column. The still image analysis toolkit managed to identify a jumper, an horizontal bar and a pole. In addition, the fact that the

Figure 26: Example of a multimedia document

jumper touches the pole and that she is near the horizontal bar was identified and stated through proper relations, shown in Figures 27. and 28.

MLCs
jumper
pole
horizontal_bar
foam_mat
. . .
Relations
near(horizontal_bar, jumper)
touches(jumper, pole)
. . .

Figure 27: Information extracted from an image

Surekha edges closer to 4m - Indian Grand Prix, 1st leg
Thursday 16 February 2006
New Delhi, India - Vazhipali Suresh Surekha regained the national mark for womens pole vault from Chetna Solani scaling 3.90m braving the afternoon temperature in the first leg of Indian GP circuit meet at Jawaharlal Nehru Stadium here. Surekha, with roots from Kerala but opting to participate for her domicile state of Tamil Nadu, however failed to accomplish her dream of sailing over 4m on Wednesday (15 Feb).
Women:
Pole Vault: 1. V.S. Surekha (TN) 3.90 [National Record]; 2. Geetanjali Bora (Asm) 3.75; 3. Chetna Solanki (UP) 3.60;

Athlete name	Vazhipali Suresh Sureka Geetanjali Bora Chetna Solani
Nationality	-
Age	-
Sport	women's pole vault
Ranking	1, 2, 3
Performance	3.90m, 3.75m, 3.60m
Location	New Delhi, India
Event	Indian Grand Prix, 1st leg
Date	Thursday 16 February 2006

Figure 28: Information extracted from a text

Provided that concepts similar to the TBox in Figure 15 exist in the ontology, the interpretation activity of the information extraction toolkit may produce the following explanation (ABox):

$$
\begin{array}{rcl}
\mathrm{pole}_1 & : & \mathrm{Pole} \\
\mathrm{jumper}_1 & : & \mathrm{Jumper} \\
\mathrm{horizontal\_bar}_1 & : & \mathrm{HorizontalBar} \\
(\mathrm{horizontal\_bar}_1, \mathrm{jumper}_1) & : & \mathrm{near} \\
(\mathrm{jumper}_1, \mathrm{pole}_1) & : & \mathrm{touches} \\
\mathrm{name}_1 & : & \mathrm{Name} \\
\mathrm{name}_2 & : & \mathrm{Name} \\
\mathrm{name}_3 & : & \mathrm{Name} \\
\mathrm{ranking}_1 & : & \mathrm{Ranking} \\
\mathrm{ranking}_2 & : & \mathrm{Ranking} \\
\mathrm{ranking}_3 & : & \mathrm{Ranking} \\
\mathrm{performance}_1 & : & \mathrm{Performance} \\
\mathrm{performance}_2 & : & \mathrm{Performance} \\
\mathrm{performance}_3 & : & \mathrm{Performance}
\end{array}
$$

$jumper_2$	:	Jumper
$jumper_3$	:	Jumper
$sport_1$	:	Sport
$location_1$	:	Location
$athletic\_event_1$	:	AthleticEvent
$date_1$	:	Date
($jumper_1$, $name_1$)	:	hasName
($jumper_1$, $ranking_1$)	:	hasRanking
($jumper_1$, $performance_1$)	:	hasPerformance
($jumper_1$, $sport_1$)	:	hasSport
($jumper_2$, $name_2$)	:	hasName
($jumper_2$, $ranking_2$)	:	hasRanking
($jumper_2$, $performance_2$)	:	hasPerformance
($jumper_2$, $sport_2$)	:	hasSport
($jumper_3$, $name_3$)	:	hasName
($jumper_3$, $ranking_3$)	:	hasRanking
($jumper_3$, $performance_3$)	:	hasPerformance
($jumper_3$, $sport_3$)	:	hasSport
($sport_1$, $location_1$)	:	hasLocation
($sport_1$, $date_1$)	:	hasDate
($sport_1$, $jumper_1$)	:	hasParticipant
($sport_1$, $jumper_2$)	:	hasParticipant
($sport_1$, $jumper_3$)	:	hasParticipant
($athletic\_event_1$, $sport_1$)	:	hasSport
$\mathbf{pole\_vault}_1$	**:**	**PoleVault**
($\mathbf{pole\_vault}_1$, $\mathbf{jumper}_1$)	**:**	**hasPart**
($\mathbf{pole\_vault}_2$, $\mathbf{jumper}_2$)	**:**	**hasPart**
($\mathbf{pole\_vault}_1$, $\mathbf{jumper}_1$)	**:**	**hasPart**
($\mathbf{pole\_vault}_1$, $\mathbf{horizontal\_bar}_1$)	**:**	**hasPart**
($\mathbf{pole\_vault}_1$, $\mathbf{pole}_1$)	**:**	**hasPart**
($\mathbf{athletic\_event}_1$, $\mathbf{pole\_vault}_1$)	**:**	**hasPart**

The above ABox constitutes the input to the ontology evolution process and according to it, the multimedia document describes a pole vault event (i.e., the HLC *Pole_Vault* has been found through interpretation) performed in the context of an athletic event, in which details about three jumpers (their ranking/performance) are presented. As this is a case of a single concept explanation, the evolution pattern P1 is selected leading to a population activity. In order to perform the first task (instance matching), the HLC of the incoming ABox must be identified. Since we are interested only in HLCis at this stage, we use the retrieval inference services offered by the ontology reasoning services (Section 4) to find all individuals of the ABox that are instances of HLCi concepts like *Athlete*, *Sport* and *Athletic_Event*. The results of instance retrieval can be:

$jumper_1$	:	Athlete
$jumper_2$	:	Athlete
$jumper_3$	:	Athlete
$sport_1$	:	Sport
$athletic\_event_1$	:	AthleticEvent

Once the HLCis contained in the ABox have been identified, they are passed to the instance matching task, with the goal of identifying other instances contained in the ontology that are similar to the HLCis retrieved from the ABox. For the purposes of this example, we suppose that the instance matching task has returned 2 similar instances for instance $jumper_1$ that are additionally grouped together representing the same real object or event by the instance grouping task, 1 similar instance for $jumper_2$ and no similar instances for $jumper_3$. In addition, several instances similar to the sport and athletic event instances were found, all grouped as representing the same real entities. However, for reasons related to the simplicity and ease of understanding of this example, we will focus primarily on the returned athlete instances, as the same process can be repeated for instances of every other concept, such as sport and athletic event. The athlete instances from the ABox as well as their similar instances from the ontology are shown in Table 6 and in Figure 29.

ABox instances	Similar instances from the ontology	
jumper$_1$ name: name$_1$ ranking: ranking$_1$ performance: performance$_1$ round: sport$_1$ age: - gender: - nationality: - ...	**jumper**$_{1520}$ name: name$_{1520}$ ranking: ranking$_{1520}$ performance: perfromance$_{1520}$ round: sport$_{1520}$ age: 24 gender: woman nationality: Indian ... **jumper**$_{1580}$ name: name$_{1580}$ ranking: ranking$_{1580}$ performance: perfromance$_{1580}$ round: sport$_{1580}$ age: 23 gender: woman nationality: - ...	**AthleteParticipations**$_2$ group: jumper$_{1520}$,jumper$_{1580}$
jumper$_2$ name: name$_2$ ranking: ranking$_2$ performance: perfromance$_2$ round: sport$_1$ age: - gender: - nationality: - ...	**jumper**$_{2650}$ name: name$_{2650}$ ranking: ranking$_{2650}$ performance: perfromance$_{2650}$ round: sport$_{2650}$ age: - gender: woman nationality: - ...	
jumper$_3$ name: name$_3$ ranking: ranking$_3$ performance: perfromance$_3$ round: sport$_1$ age: - gender: - nationality: - ...		

Table 6: Similar instances retrieved in the ontology

Once groups of similar instances have been identified, these groups must be validated. An ABox corresponding to all instances of an instance group is constructed (by refining the original ABox) and its consistency with respect to the domain ontology is checked through the standard inference services. It is worth noting that consistency checking verifies compatibility of instances with respect to the ontology (which is a standard ABox validation process), but can additionally check compatibility with respect to the values of the various instance properties. For example, two instances of the same athlete can have different values for the nationality property. This could be an error (e.g., because the information extraction toolkit has made a mistake or the original multimedia document was wrong) or could not be correct (e.g., the same athlete chose to participate in a specific athletic event under a different nationality). A quick analysis of the athletics domain reveals several properties that can have incompatible values, such as age (which is different according to the event date), sport (as an athlete can participate in two or even more sports). Deciding whether a different property value is an error or not is almost impossible in an evolving infrastructure such as the BOEMIE approach, which assumes an open world where new information is used to expand its knowledge base. As a result, property value compatibility checks are performed only in some properties that cannot change dramatically, such as an athlete's surname. Based on the results of each validation stage, instances that fail the ABox validation are discarded and are not considered for ontology population, while instances that fail the property value validation are excluded from the group and are placed in a new group. Continuing our example, we assume that all considered instances in all groups passed the validation checks. Thus, population continues by performing ABox assimilation, considering the groups shown in Figure 30.

During the ABox assimilation task, all the required changes are performed in the ontology

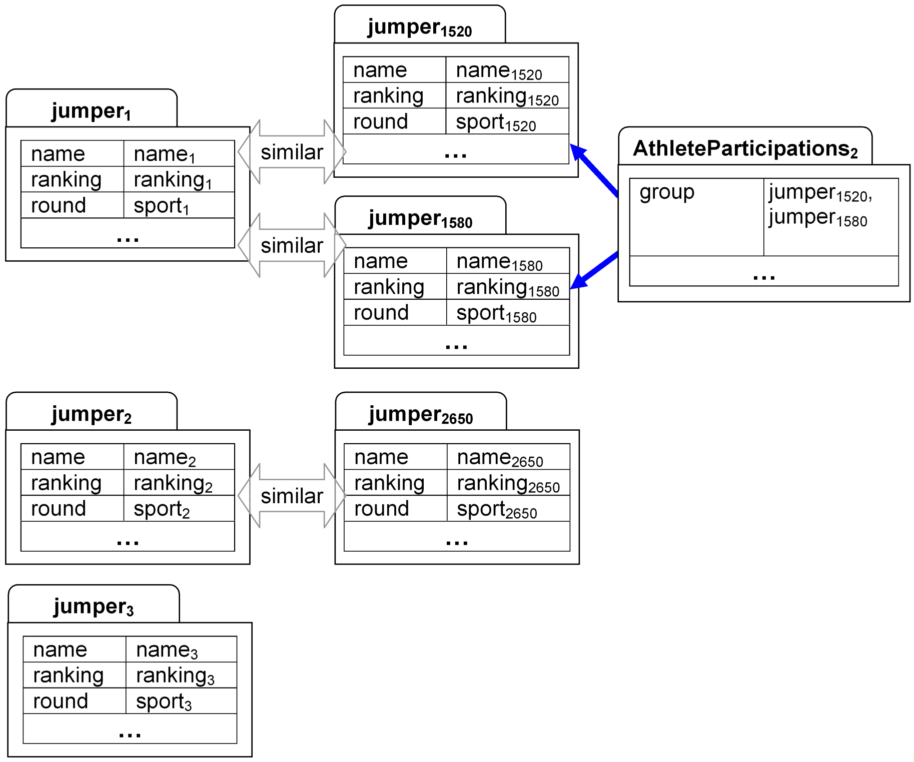

Figure 29: Graphical representation of similar instances of the example

(by creating all instances/relations in all involved ontological modules), in order to incorporate the information from the ABox into the ontology. ABox assimilation is responsible for inserting the extracted MLCis, their relations and the HLCis that explain these MLCis into the ontology. The ABox assimilation process follows a bottom-up approach, by choosing to populate first the multimedia content ontological module. The modality element that describes an MLCi in the still images modality is a *still region*, while a textual MLCi is represented by a *phrase*. As a result, all needed instances that represent all MLCis in the ABox populate the TBox, as shown in Figure 31:

After populating the multimedia content ontology module, the population proceeds with the modality elements ontology module (Figure 32).

The third ontology module populated is the relation ontological module, where spatiotemporal (for audio, video and still image modalities) and other relations (for the text modality) are stored. An example of such a population is provided in Figure 33

Once the multimedia content, the modality elements, and the relation ontology modules have been populated, the domain ontology is finally populated. In the Figure 34, an example of population of the domain ontology with an instance of a jumper (jumper$_1$) is presented. The process for populating with instances of any other concept is the same.

5.3 Measurable objectives

Measurable objectives for the population activity are the performance of the instance matching and instance grouping tasks. There are two scenarios for evaluating these two tasks.

The first evaluation scenario uses as input the manually annotated corpus (this is the ideal input expected from the information extraction toolkit; no ambiguous interpretations – evolution pattern P1 case is only covered) to measure the performance of instance matching and instance grouping tasks. A target of 90% or more may be set for this evaluation scenario.

The second scenario takes as input the system annotated corpus (covers both P1 and P2 evolution patterns). The performance of the information extraction toolkit is known from WP2 evaluation, meaning that it is known which of the interpretations are correct, erroneous, etc. The measuring objective of this scenario is how well the population activity manages to populate

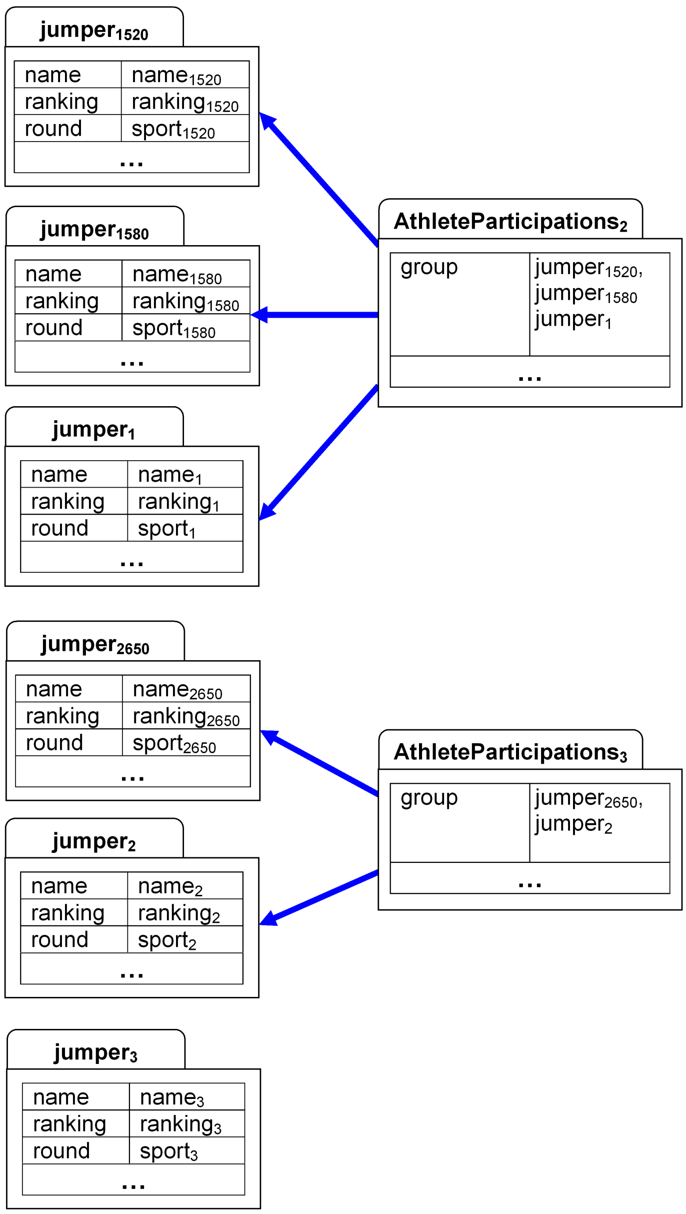

Figure 30: Groups selected for population

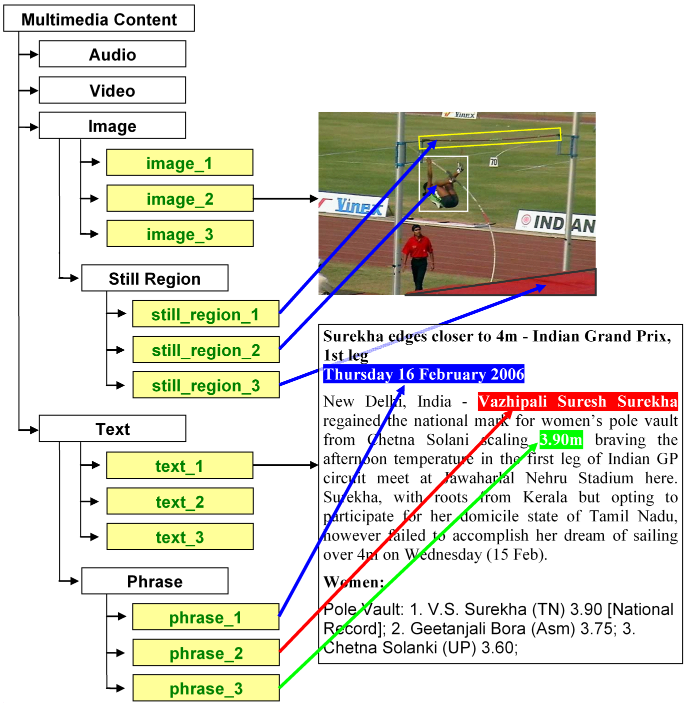

Figure 31: Population of the multimedia content ontology module

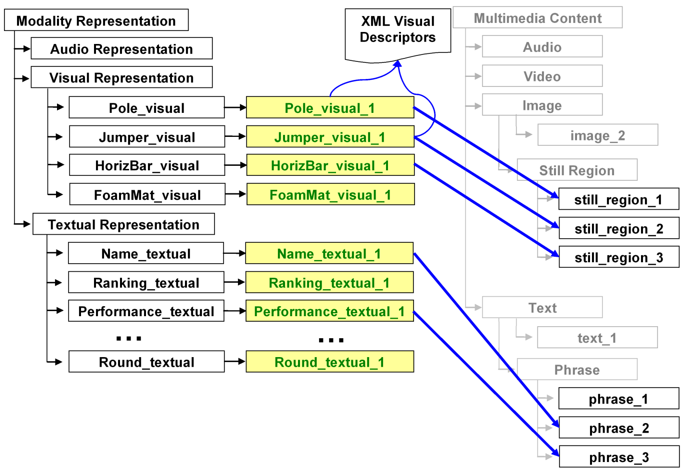

Figure 32: Population of modality elements

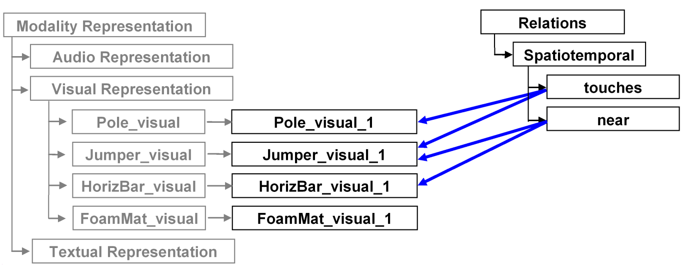

Figure 33: Population of relation ontology module

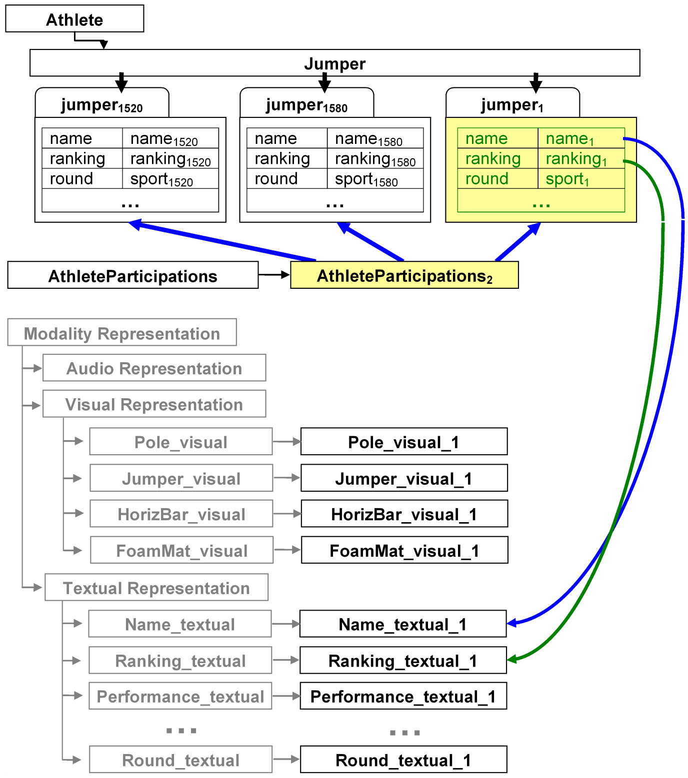

Figure 34: Population of the domain ontology

the ontology without loss of extracted information and at the same time solving the ambiguous interpretations (evolution pattern P2).

6 ONTOLOGY ENRICHMENT

Ontology enrichment is the activity of extending an ontology, through the addition of new concepts and relations. Ontology enrichment is performed every time the background knowledge is not sufficient to explain the extracted information from the processed multimedia documents. Thus, the ontology enrichment activity is expected to extend this background knowledge through the addition of new concepts/relations, in order to better explain extracted information in subsequent cycles of the bootstrapping process.

The ontology enrichment activity is triggered by either P3 or P4 evolution patterns. Evolution pattern P3 is selected when no explanation (i.e., an HLC) has be found for a given ABox, and can lead to the insertion of a new HLC or a new relation into the ontology, or in the accumulation of the ABox in a "waiting" queue if available evidence cannot justify the addition of a new concept. On the other hand, evolution pattern P4 is selected when the background knowledge is not sufficient to even assign MLCs to all of the extracted elements of a multimedia resource, thus leaving extracted modality descriptors not associated with a MLC in the ABox. In this case, pattern P4 can result in the addition of a new MLC in the ontology. In fact, we consider the detection of the missing MLC to have priority over the identification of the missing HLC, because knowing about the new MLC could lead to different interpretation results about the same resources. The new interpretation will thus be performed in the next bootstrapping cycle. Ontology enrichment is decomposed into the following tasks:

- *Concept learning*: The goal of this task is to propose new concepts (either HLCs or MLCs) and relation by exploiting similarities found through clustering, either in unexplained documents (evolution pattern P3) or in unknown objects recognized by the information extraction toolkit (evolution pattern P4). It can be decomposed into two main tasks, clustering and concept formation.
  - *Clustering*: The main objective of the clustering task is to provide evidence that can support the creation of new concepts or relations.
  - *Concept formation*: This task is applicable only if a new HLC has been proposed by clustering. Exploiting the results of clustering, concept formation examines the ABoxes of the clustered elements in order to extract common information (such as concepts and relations) and use this common information to form the new concept through a new ABox, which is the result of this task.
- *Concept enhancement*: This task is responsible for improving a concept identified by concept learning, through knowledge acquired from external sources, such as external domain ontologies or taxonomies.
- *Concept definition*: This task receives the new concept (either a new MLC or HLC) and relation as defined through the previous tasks, and shows the concept definition to the ontology expert. The ontology expert must approve the new concept in order to be assimilated into the ontology and additionally can revise the definition of the new concept.
- *Concept validation*: This task performs consistency checking, by trying to detect possible inconsistencies due to the addition of the new concept relation to the ontology.
- *Concept assimilation*: The last task of ontology enrichment is responsible for performing the needed changes in the ontology in order to incorporate the newly formed concept into the domain ontology TBox.

Unlike the population activity, ontology enrichment changes the TBox of the ontology and thus the background knowledge contained in the ontology. BOEMIE subsystems that rely on this background knowledge (such as the information extraction toolkit) must be notified about TBox changes, as they need to adapt to the modified background knowledge. For example, MLC recognizers of the information extraction toolkit must be adapted to recognize new MLC concepts that have been added into the TBox. As a result, when an enrichment activity is applied to the ontology (through actions taken during the P3 and P4 evolution patterns), the bootstrapper must be notified about all the changes occurred, in order to provide the correct feedback to the BOEMIE components affected, such as the information extraction toolkit.

In the following two subsections, the tasks performed for both P3 and P4 evolution patterns will be described in detail, along with a complete example showing the various tasks performed in the cases of evolution patterns P3 and P4 trigger an enrichment activity.

6.1 Missing concept with metadata explanation

This section describes the tasks performed when evolution pattern P3 has been selected by the evolution pattern selection activity. Evolution pattern P3 is selected when a multimedia document cannot be explained by the semantic interpretation. When a document triggers pattern P3 no tasks are performed: the ABox of the document is simply placed in a "waiting" queue, where ABoxes related to unexplained documents are accumulated. This waiting queue can have unexplained ABoxes from other documents of this bootstrapping cycle, but also from earlier bootstrapping invocations.

Thus, the selection of evolution pattern P3 simply leads to the addition of the document's ABox in a set of unexplained ABoxes. When all multimedia documents given as input to the evolution toolkit for this bootstrapping cycle have been examined, then the enrichment process that corresponds to evolution pattern P3 is triggered. As a result, enrichment tasks related to pattern P3 are activated only after all documents associated with pattern P3 have been accumulated in the "waiting" area.

The goal of the enrichment task associated with evolution pattern P3 is to define new HLCs and relations, and present these (along with the evidence supporting their creation) to the ontology expert. In order to support this goal, enrichment tries to exploit commonalities that may exist among the unexplained ABoxes (for creating new HLCs) or among ABoxes with unexplained MLCs through clustering. The tasks that support this goal, are shown in the following table:

HLC Learning	Relation Learning

• Concept learning
  – Clustering
  – Concept formation
• Concept enhancement
• Concept definition
• Concept validation
• Concept assimilation
	• Concept learning
	  – Clustering
	• Concept definition
	• Concept validation
	• Concept assimilation

6.1.1 Concept learning

The concept learning task in the context of evolution pattern P3 is responsible for proposing either a new HLC or a new relation to be added in the ontology TBox, based on similarity evidence among unexplained ABoxes (for learning new HLCs) or unexplained MLCs (for learning relations) through clustering. As a result, concept learning involves two types of clustering that have different goals and operate upon different input.

Clustering for HLC learning. Clustering for HLC learning aims to propose new HLCs by exploiting similarities in the ABoxes that have been accumulated in the "waiting" queue. Clustering will try to cluster the ABoxes found by using information relative to the MLCIs, HLCIs, relations among them, but also the *terms identified by the text modality* of the extraction toolkit. Despite of the fact that terms are in essence unknown objects and also modality specific objects of the text modality, we believe that the clustering task for HLC learning can exploit them not only because they can provide important evidence for clustering, but also because it is easy to use them: they can be used directly as symbolic values, without the need to resort to any other low level features – a unique property of the unknown objects of the text modality.

The result of this clustering task is a set of clusters, each cluster containing a set of ABoxes that are similar to each other, but unexplained. Such a cluster is assumed to contain evidence about a real object or event that cannot be explained by the current TBox and thus a new HLC is required to explain all ABoxes in the cluster.

Clustering for relation learning. Clustering for relation learning tries to propose new relations among concepts (either HLCs or MLCs), by examining the unexplained concepts in ABoxes that have been explained and thus have populated the ontology. When the ABox of a multimedia document contains an explanation, this does not necessarily mean that all instances (either MLCIs or HLCIs) are related to this explanation. It is possible that some (known to the ontology) MLCIs/HLCIs may still be unexplained (unrelated to the explanation) despite the fact that an explanation exists for the ABox. The idea behind relation learning is that the frequent existence of an unrelated instance in ABoxes that share the same explanation may be an indication that this instance can be part of the explanation, but was left unexplained due to a missing relation. Thus, clustering for relation learning tries to identify these situations in order to propose a new relation among an unexplained concept and concepts that participate in the explanation.

The input to the clustering for relation learning is a set of instances (ABoxes) that populate a specific HLC, and the ABox of the document containing a set of unexplained instances. The result of this clustering task is a set of clusters, where each cluster contains a set of ABoxes sharing the same unexplained concept instance. Each cluster is assumed to contain evidence about a possible relation between the unexplained concept and the explained ones. However, no attempt is made to explicitly define the relation. Instead, the evidence is shown to the ontology expert (during concept definition), which is responsible to define the relation if he decides that there is enough evidence to justify the addition of a new relation.

Concept formation. If for many interpretation tasks the abduction process described in Section 5 results in degenerated results, the Γ ABoxes provide the basis of a concept definition process. For this purpose, a bottom-up construction approach through the use of non-standard reasoning services (see Section 4.4) represents a good option to learn new concept definitions (see [Baader et al., 2003, P. 250 ff] for an overview). Figure 35 illustrates a general approach in which from the discovery of most specific concepts (MSC), their least common subsumer (LCS) is computed and finally minimal rewriting is applied on the LCS to produce a new concept C'. Afterwards the new concept C' can be used in order to discover new concepts to be added in the Tbox part of the KB.

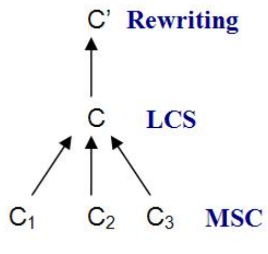

Figure 35: Bottom-up approach to learn new concept definitions

To illustrate such a process in the context of BOEMIE, let us consider the ABoxes in Figure 36 as Γs that after abduction represent only degenerated interpretation results. Each Γ corresponds to a different media object (note that for this example no spatial or topological relations are considered). These Γ_is represent the basis to learn new concept definitions, thus following the bottom-up approach, each Γ is considered for the creation of a MSC. For this purpose we consider the ABox individuals as parts of some anonymous aggregate instance named with a fresh individual, e.g, $x_i : C11Y$ and finally create *has_Part* relations among the individuals and the new aggregate individual (see Figure 36) in order to obtain the augmented ABoxes Γ'_i ($C11Y$ is some arbitrary aggregate concept).

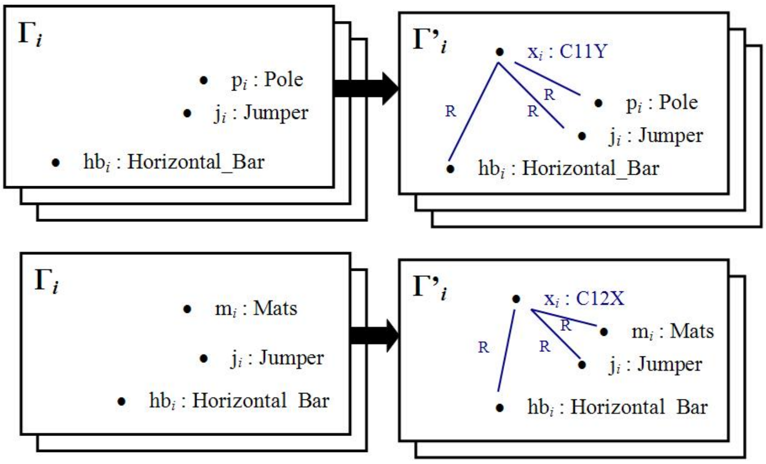

Figure 36: Γ ABoxes as basis for learning new concept definitions

From the Γ_i's the specification of MSCs is observed in Figure 37 from which the LCS is obtained.

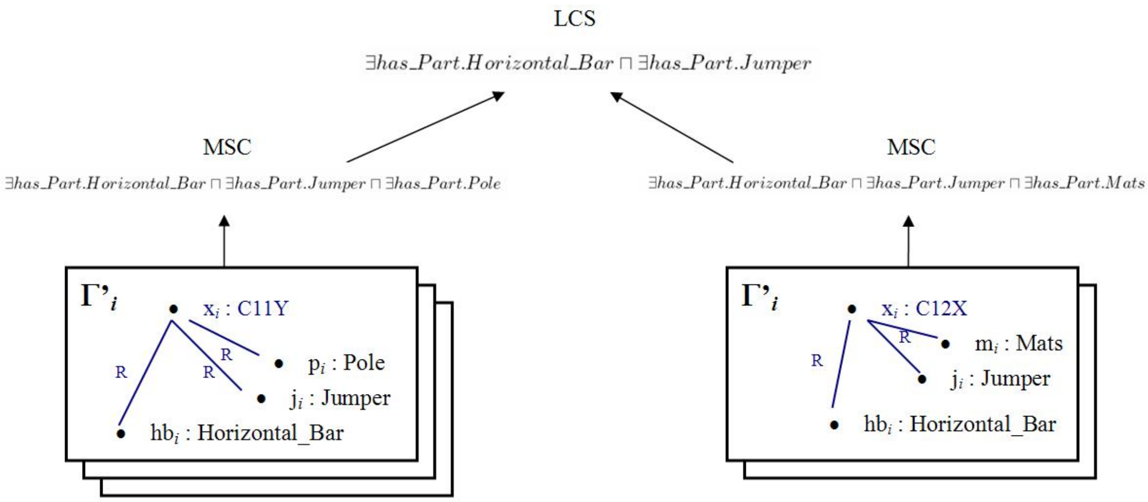

Figure 37: Obtaining the LCS from MSC

The LCS concept can be possibly rewritten using the terminology to make it more succinct. If no rewriting is possible then the new concept definition (GCI), e.g.,

$$C13X \sqsubseteq \exists has\_Part.Horizontal\_Bar \sqcap \exists has\_Part.Jumper$$

could be added to the TBox part of the KB. However, in order to frame the new concept in the ontology, it is desirable to discover a meaningful name and other possible features of it.

6.1.2 Concept enhancement

The idea of concept enhancement is that, after learning the LCS C', we have a definition of the components required in the new concept, but not a definition of a specific event or object to be used in order to explain the ABoxes. More information to support the ontology expert in enriching the ontology is needed. To this end, a concept enhancement activity is triggered. The goal of concept enhancement is to find elsewhere, by probing other external knowledge sources, definitions of concepts similar to the anonymous concept whose existence has been learned in the previous phase. Such discovered concept definitions are then used as the starting basis by the ontology expert for interactively defining the new concept to be inserted into the domain ontology. Concept enhancement is performed through probe queries. A probe query is a description of the properties, the constraints, and the semantic relations that should be exhibited by the (actually anonymous) new concept in order to consider it as a suitable candidate for ontology enrichment. The probe query is defined by taking into account both the properties and relations of the anonymous concept and the current version of the ontology. Furthermore, in order to reduce the involvement of the ontology expert as much as possible, in probe query formulation, an internal concept discovery process can be triggered which explores the background ontology knowledge with the goal of collecting all the knowledge available in the ontology regarding C'. In particular, we look for one or more concepts already present in the ontology that are featured by the same or similar properties of C'. This *internal concept discovery* is performed by means of ontology matching techniques. In BOEMIE, we adopt a matching service [Castano et al., 2006] capable of determining the level of matching of two different concepts by evaluating, separately or in combination, their *linguistic affinity* and *contextual affinity*, respectively. Linguistic affinity provides a measure of similarity between two ontology concepts C and C' computed on the basis of their linguistic features (i.e., concept names). Contextual affinity provides a measure of similarity by taking into account the contextual features of the ontology concepts C and C'. The context of a concept can include properties, semantic relations with other concepts as well as property restrictions. Given the features of the anonymous concept C', ontology matching is performed by taking into account only the contextual affinity and by using a threshold t in order to discard the concepts with a low similarity with C'. This matching process produces as a result a set MC of matching concepts with C', such that, $MC = \{C_i \in \mathcal{O} \mid SA(C_i, C') = CA(C_i, C') \geq t\}$, where $SA(C_i, C')$ denotes the semantic affinity between C_i and C', and $CA(mc_i, C')$ denotes the contextual affinity between C_i and C'. With respect to the example, for internal discovery we match the concept

$$C13X \sqsubseteq \exists hasPart.Horizontal\_Bar \sqcap \exists hasPart.Jumper$$

against the TBox shown in Figure 15, by supposing that the concepts $High\_Jump$ and $Pole\_Vault$ are missing. We use the matching service default threshold (i.e., 0.5). Since the ontology of the example is very poor, we obtain only a matching concept for $C13X$, that is:

$$CA(C13X, Jumping\_Event) = 0.75.$$

The result of the internal discovery step, is then used as starting basis for formulating probe queries to enable concept enhancement over external knowledge sources. For each retrieved matching concept C_i in the ontology a *probe query* is created. A probe query is basically a set of axioms that describes the target of concept enhancement, that is a description of the properties, the constraints, and the semantic relations that should be exhibited by the (actually missing) new concept in order to consider it as a candidate for ontology enrichment. In particular, we define a probe query containing the definition of C' and a probe query PQ_i for each matching concept $C_i \in MC$ retrieved through the internal discovery process. Each PQ_i contains all the properties, restrictions and relations featuring the corresponding concept C_i. Referring to the example, we define two probe queries, as shown in Figure 38.

$$PQ_1 : \quad [Jumping\_Event \sqsubseteq Event \sqcap \exists hasPart.Jumper]$$

$$PQ_2 : \quad [C13X \sqsubseteq \exists hasPart.Horizontal\_Bar \sqcap \exists hasPart.Jumper]$$

Figure 38: Example of probe queries

Each probe query is then matched against one or more external knowledge sources. The goal of this phase is to discover some external concepts that could be useful to define the new candidate concept to be inserted in the ontology. The different probe queries can be matched by exploiting different configurations of the matching service, depending on the probe query contents. The probe query containing the definition of C' is always matched by evaluating contextual affinity, since we do not have a meaningful name for C', while probe queries composed by using ontology concepts are executed evaluating a comprehensive measure of semantic affinity SA, which takes into account also the linguistic affinity features. In particular, the semantic affinity value is evaluated as a linear combination of linguistic affinity and contextual affinity, by using a threshold-based mechanism for matching concept selection, such that, the lower the threshold t, the higher the number of concepts that are retrieved as candidates for ontology enrichment. The execution of a probe query PQ_i against an external knowledge source KS produces a (possibly empty) set of mappings m_i, of the form $m_i\langle C_i, C', SA(C_i, C')\rangle$, where C_i denotes a probe query concept in PQ_i, C' denotes a concept in the external source KS, and $SA(C_i, C')$ denotes the semantic affinity between C_i and C', respectively [5]. Finally, for each mapping m_i, the corresponding external concept C' is returned to the ontology expert as a candidate for ontology enrichment.

With respect to the example, we match the probe queries shown in Figure 38 against a taxonomy of concepts in the athletics domain extracted from Wikipedia [6], which is shown in Figure 39. In

$$
\begin{array}{rcl}
Events\_in\_Athletics & \sqsubseteq & Athletics \\
All\_round\_athletics & \sqsubseteq & Events\_in\_Athletics \\
Decathlon & \sqsubseteq & Events\_in\_Athletics \\
Discus\_Throw & \sqsubseteq & Events\_in\_Athletics \\
Half\_Marathon & \sqsubseteq & Events\_in\_Athletics \\
Hammer\_Throw & \sqsubseteq & Events\_in\_Athletics \\
Heptathlon & \sqsubseteq & Events\_in\_Athletics \\
High\_Jump & \sqsubseteq & Events\_in\_Athletics \\
Hurdling & \sqsubseteq & Events\_in\_Athletics \\
Javelin\_Throw & \sqsubseteq & Events\_in\_Athletics \\
Long\_Distance\_Track\_Event & \sqsubseteq & Events\_in\_Athletics \\
Long\_Jump & \sqsubseteq & Events\_in\_Athletics \\
Marathon & \sqsubseteq & Events\_in\_Athletics \\
Middle\_Distance\_Track\_Event & \sqsubseteq & Events\_in\_Athletics \\
Penthatlon & \sqsubseteq & Events\_in\_Athletics \\
Pole\_Vault & \sqsubseteq & Events\_in\_Athletics \\
Race\_Walking & \sqsubseteq & Events\_in\_Athletics \\
Running & \sqsubseteq & Events\_in\_Athletics \\
Shot\_Put & \sqsubseteq & Events\_in\_Athletics \\
Spring & \sqsubseteq & Events\_in\_Athletics \\
Standing\_High\_Jump & \sqsubseteq & Events\_in\_Athletics \\
Standing\_Long\_Jump & \sqsubseteq & Events\_in\_Athletics \\
Standing\_Triple\_Jump & \sqsubseteq & Events\_in\_Athletics \\
Steplechase & \sqsubseteq & Events\_in\_Athletics \\
Triple\_Jump & \sqsubseteq & Events\_in\_Athletics \\
Ultramarathon & \sqsubseteq & Events\_in\_Athletics \\
Womens\_Penthatlon & \sqsubseteq & Events\_in\_Athletics
\end{array}
$$

Figure 39: Example of an external knowledge source (OWL representation of a Wikipedia portion)

the example, we use a low threshold value of 0.3, in order to obtain a high number of candidate concepts for ontology enrichment. The evaluation of PQ_1 and PQ_2 of Figure 38 against the source taxonomy of Figure 39, returns the mappings shown in Figure 40. According to the results shown in Figure 40, the following concept names are returned to the ontology expert as candidates for ontology enrichment.

> $Athletics$, $Events\_in\_athletics$, $High\_Jump$, $Long\_Jump$, $Pole\_Vault$, $Standing\_High\_Jump$, $Standing\_Long\_Jump$, $Standing\_Triple\_Jump$, $Triple\_Jump$

The ontology expert evaluates the candidates in order to define the most appropriate concept definition for the anonymous concept to be added in the ontology.

[5] In case of contextual affinity evaluation, $SA(C_i, C')$ is equal to $CA(C_i, C')$

[6] http://en.wikipedia.org/wiki/Category:Events_in_athletics

		PQ_1
m_1	=	$\langle Jumping\_Event, High\_Jump, 0.4\rangle$
m_2	=	$\langle Jumping\_Event, Long\_Jump, 0.4\rangle$
m_3	=	$\langle Jumping\_Event, Triple\_Jump, 0.4\rangle$
m_4	=	$\langle Jumping\_Event, Athletics, 0.32\rangle$
m_5	=	$\langle Jumping\_Event, Pole\_Vault, 0.32\rangle$
m_6	=	$\langle Jumping\_Event, Standing\_High\_Jump, 0.32\rangle$
m_7	=	$\langle Jumping\_Event, Standing\_Long\_Jump, 0.32\rangle$
m_8	=	$\langle Jumping\_Event, Standing\_Triple\_Jump, 0.32\rangle$

		PQ_2
m_9	=	$\langle C13X, Events\_in\_athletics, 0.45\rangle$

Figure 40: Result of the evaluation of PQ_1 and PQ_2

6.1.3 Concept definition and validation

The concept definition task aims to present to the ontology expert the definition of a new concept along with the evidence supporting its creation, or the evidence supporting the creation of a new relation, in order to finally decide the addition of a concept/relation to the ontology. In the case of a new concept, the formation of the new concept (as formed by the concept learning, concept definition and concept enhancement tasks) is presented to the ontology expert. In addition, the cluster of ABoxes that lead to the initial definition of the concept is presented to the ontology expert as the evidence justifying the addition of the new concept. The ontology expert must decide if the concept should be inserted into the ontology and, if necessary, he can revise the concept definition. If the decision to add the new concept into the ontology is taken, the ontology expert is expected to select which ABoxes/documents from the ones presented can be explained by the new concept (i.e., which ones become instances of the new concept in the ontology).

When the ontology expert is required to analyze a set of possible candidates for ontology enrichment, he chooses the concept names that provide the best description of the new resources, according to his understanding of the domain. Note that, since the candidates are extracted from external knowledge sources (e.g., web directory taxonomies), the expert can also examine the external source in order to increase his understanding of the domain, if necessary. However, the name selection and the features of the anonymous concept C' are not sufficient for concept definition. In fact, the ontology expert is required also to define other axioms concerning the new concept, in order to use it for population. In the case of the example shown in Figure 37, we have two arbitrary aggregate concepts $C11Y$ and $C12X$, that are defined as follows:

$$C11Y \sqsubseteq \exists hasPart.Horizontal\_Bar \sqcap \exists hasPart.Jumper \sqcap \exists hasPart.Pole$$

$$C12X \sqsubseteq \exists hasPart.Horizontal\_Bar \sqcap \exists hasPart.Jumper \sqcap \exists hasPart.Mats$$

respectively. The ontology expert choses, among the candidate concept names, the candidates $Pole\_Vault$ for $C11Y$ and $High\_Jump$ for $C12X$, and obtains the following definitions:

$$Pole\_Vault \sqsubseteq \exists hasPart.Horizontal\_Bar \cap \exists hasPart.Jumper \cap \exists hasPart.Pole$$

$$High\_Jump \sqsubseteq \exists hasPart.Horizontal\_Bar \cap \exists hasPart.Jumper \cap \exists hasPart.Mats$$

The two concept definitions have then to be framed into the current version of the ontology, that is shown in Figure 41.

In order to find a suitable location for the new concepts in the ontology, the ontology expert can use again the matching service, by matching the new concept definitions against the current version of the ontology of Figure 41. In the example, the matching results are the following:

$$SA(High\_Jump, Jumping\_Event, 0.6)$$

$$SA(Pole\_Vault, Jumping\_Event, 0.56)$$

$$
\begin{array}{rcl}
Man & \sqsubseteq & Person \\
Woman & \sqsubseteq & Person \\
Man & \sqsubseteq & \neg Woman \\
Athlete & \equiv & Person \sqcap \exists hasProfession.Sport \\
Foam\_Mat & \sqsubseteq & SportEquipment \\
Pole & \sqsubseteq & SportEquipment \\
Javelin & \sqsubseteq & SportEquipment \\
Horizontal\_bar & \sqsubseteq & SportEquipment \\
Jumping\_Event & \sqsubseteq & Event \sqcap \exists hasPart.Jumper
\end{array}
$$

Figure 41: Current version of the ontology

by using both linguistic and contextual affinity evaluation. Moreover, the ontology expert can use the matching service also in order to evaluate the similarity between the components of the two aggregate concepts (i.e., $Horizontal\_Bar$, $Jumper$, $Pole$, $Mats$) and the concepts already present in the ontology. By using this facility, the expert retrieves the following results:

$$SA(Horizontal\_Bar, Horizontal\_Bar, 1.0)$$

$$SA(Jumper, Jumper, 1.0)$$

$$SA(Mats, Foam\_Mat, 0.8)$$

The semantic affinity values obtained are intended to support the ontology expert in refining the concept definitions and to frame them in the ontology. To this end, the interpretation of the affinity values in terms of relations among the ontology concepts is left to the expert. In the case of the example, the high affinity between $Foam\_Mat$ and $Mats$ can lead the expert to the decision of substituting the restriction $\exists hasPart.Mats$ with $\exists hasPart.Foam\_Mat$. Moreover, according to his understanding of the domain, the expert can also choose to add $\exists hasPart.Foam\_Mat$ also in the definition of $Pole\_Vault$. Finally, the ontology expert analyzes the affinity measured among $Pole\_Vault$, $High\_Jump$ and $Jumping\_Event$. A possible interpretation of this affinity is to consider $Pole\_Vault$ and $High\_Jump$ as to jumping events, since the definition of the concept $Jumping\_Event$ subsumes the definition of $Pole\_Vault$ and $High\_Jump$. According to this interpretation, the two concepts are refined by inserting the axioms $Pole\_Vault \sqsubseteq Jumping\_Event$ and $High\_Jump \sqsubseteq Jumping\_Event$ and by deleting $Pole\_Vault \sqsubseteq \exists hasPart.Jumper$ and $High\_Jump \sqsubseteq \exists hasPart.Jumper$, respectively. The final definition of the new concepts is shown in Figure 42.

$$
\begin{array}{rcl}
Pole\_Vault & \sqsubseteq & Jumping\_Event \sqcap \\
 & & \exists hasPart.Pole \sqcap \\
 & & \exists hasPart.Horizontal\_Bar \sqcap \\
 & & \exists hasPart.Foam\_Mat \\
High\_Jump & \sqsubseteq & Jumping\_Event \sqcap \\
 & & \exists hasPart.Horizontal\_Bar \sqcap \\
 & & \exists hasPart.Foam\_Mat
\end{array}
$$

Figure 42: The new concepts $Pole\_Vault$ and $High\_Jump$

When the two concepts are added to the Tbox, it makes sense to check whether some concept name becomes inconsistent (coherence check, see Section 4.2). For rules however, this kind of service can not be applied because rules must be DL-safe and, therefore, are applied to Abox individuals only. If a GCI is derived as an approximation of the rule and the GCI causes some concept names to become inconsistent when the GCI is added to the Tbox, then the rule will also cause this effect, of course. In general, the consistency of rules must be checked w.r.t. example ABoxes (see the ABox consistency check described in Section 4.2).

In the previous example no spatial or topological relations where considered for the learning process, since the previous bottom-up approach does not take into consideration the existence of rules. Rules are especially useful in this project for the modeling of such spatial/topological relations.

Approaches for learning rules are presented in [Muggleton, 1999] (Inductive Logic Programming, ILP).

6.1.4 Concept assimilation

This task is responsible for integrating the validated new concept/relation with the ontology, leading to an enriched version of the ontology, which, in the case of the example, is shown in Figure 43.

$$
\begin{array}{rcl}
Man & \sqsubseteq & Person \\
Woman & \sqsubseteq & Person \\
Man & \sqsubseteq & \neg Woman \\
Athlete & \equiv & Person \sqcap \exists hasProfession.Sport \\
Foam\_Mat & \sqsubseteq & SportEquipment \\
Pole & \sqsubseteq & SportEquipment \\
Javelin & \sqsubseteq & SportEquipment \\
Horizontal\_bar & \sqsubseteq & SportEquipment \\
Jumping\_Event & \sqsubseteq & Event \sqcap \exists hasPart.Jumper \\
Pole\_Vault & \sqsubseteq & Jumping\_Event \sqcap \\
 & & \exists hasPart.Pole \sqcap \\
 & & \exists hasPart.Horizontal\_Bar \sqcap \\
 & & \exists hasPart.Foam\_Mat \\
High\_Jump & \sqsubseteq & Jumping\_Event \sqcap \\
 & & \exists hasPart.Horizontal\_Bar \sqcap \\
 & & \exists hasPart.Foam\_Mat
\end{array}
$$

Figure 43: Example of the enriched intermediate version of the ontology

6.2 Missing concept without metadata explanation

This section describes the tasks performed when evolution pattern P4 has been selected by the evolution pattern selection activity. Evolution pattern P4 operates in a similar manner to evolution pattern P3, as multimedia documents triggering evolution pattern P4 are accumulated, and tasks related to pattern P4 are activated when all documents have been examined in the extraction phase. Evolution pattern P4 is selected by the evolution pattern selection in the presence of a document ABox that contains at least one unknown MLCI. Unknown MLCIs are instances of real objects that have been recognized by the information extraction toolkit, but no information about them exists in the ontology. Thus, the information extraction toolkit cannot associate a known MLC to them, classifying them as unknown. Every unknown MLCI encountered is stored as an instance of the "unknown" MLC of the ontology along with the low level features associated with it, effectively placing the unknown MLCI in the "waiting" area.

The goal of the enrichment tasks associated with evolution pattern P4 is to decide upon the correlation of these unknown MLCIs with the application domain and to define a new MLC to describe the significant ones. Both actions are performed with the aid of the ontology expert. The tasks related to evolution pattern P4 are the following:

- Concept learning
- Concept definition
- Concept validation
- Concept assimilation

6.2.1 Concept learning

The concept learning task in the context of evolution pattern P4 is responsible for proposing a new MLC to be added in the ontology TBox, based on similarity evidence found among instances of unknown MLCs through clustering. The input to this clustering action is all unknown MLCIs placed in a "waiting" stage by the P4 pattern, along with the low-level features associated with each unknown MLCI. Clustering is expected to decide which of these instances refer to the same real object or event by exploiting similarities found mainly in the low-level features. Thus, this clustering action is modality-specific and as such cannot be performed at the level of semantics. Thus, this clustering is performed through the exploitation of the clustering service offered by the information extraction. The set of unknown MLCIs and their low-level features are provided to the clustering service of the extraction toolkit. The result of this clustering service is a set of clusters, with each cluster containing instances that refer to the same real object or event. Instances contained in such a cluster constitute the supporting evidence that will be shown to the ontology expert as justification for the creation of a new MLC.

6.2.2 Concept definition

The concept definition task is responsible for showing each cluster formed during the concept learning task to the ontology expert. The low-level features associated with all instances in a cluster must be presented to the ontology expert in a suitable, easy understandable format that will help the visualization of the instances and will allow the expert to decide whether the instances describe a relevant MLC for the application domain. For example, low-level features from the text modality may be presented as colored text portions in the document's text, whether low-level features from the image modality can be presented as colored regions on an image. Having all this information as input, the ontology expert must decide whether the displayed MLC instances are important or not. In case the addition of a new MLC has been decided, the ontology expert must provide a name for it, and select which low-level features are associated with it.

6.2.3 Concept validation

This task performs consistency checking using standard inference services (Section 4), in order to ensure that the new MLC is consistent with the ontology.

6.2.4 Concept assimilation

This task is responsible for integrating the validated new MLC with the ontology, leading to an enriched version of the ontology. When a new concept is inserted in the ontology, the links among it and the other modules of the semantic model have to be defined. This activity may require to add new elements in the ontology modules corresponding to the new MLC.

As an example, a MLC has to be linked with a concept that generalizes its visual representation, i.e., a MLC in the domain ontology should have a corresponding concept in the modality elements ontology. If this concept does not exist it is created during the concept assimilation by exploiting functionalities of the coordination tool. In particular, given a new MLC C that represents for example a media object extracted from an image, a new concept C_visual featuring the visual representation of C is created in the modality elements ontology as a subclass of the concept $Visual\_Representation$. Then, C is linked to C_visual by means of the $hasVisualRepresentation$ relation. This link may be represented as shown in Figure 44.

6.3 Example: learning a new concept

As an example of the ontology enrichment activity, the case of learning a new HLC will be described, as it involves all tasks described in this section. For the purposes of this example, we assume to have processed a set of HTML pages from an image gallery of a web site related to the Athletics domain. Each HTML page is assumed to contain only an image and sometimes a legend. A second assumption is that our ontology miss concepts for pole vault or high jump, but the information extraction toolkit is able to recognize MLCs like athlete, pole and horizontal

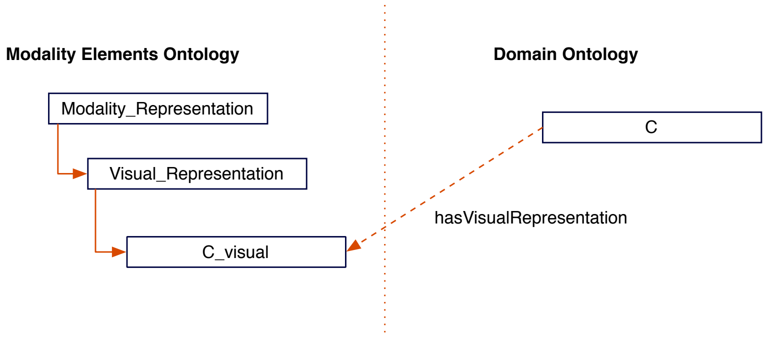

Figure 44: Definition and linking of a new visual element

bar. Under such assumptions, an explanation is not found for images involving pole vault events during the interpretation phase performed by the extraction toolkit. Thus, the evolution process will receive an input similar to the one shown in Table 8. None of the those ABoxes contains an explanation. All ABoxes contain some MLCIs and a few relations, mainly provided by the image analysis. The three incoming ABoxes are analyzed by the evolution pattern selection activity, and, due to the absence of an explanation, the evolution pattern P3 is selected. The ABoxes are then placed in a "waiting" area where they remain until all documents provided as input to the evolution process have been processed. Once all input documents have been processed, the three unexplained ABoxes in the "waiting" area are processed according to the following tasks of the evolution pattern P3:

- Concept learning
  - Clustering
  - Concept formation
- Concept enhancement
- Concept definition
- Concept validation
- Concept assimilation

The first task performed is concept learning, involving two tasks, namely clustering and concept formation. In our example, the clustering task produces a cluster containing the three ABoxes shown in Table 8, by exploiting the similarities among the ABoxes, such as the existence of the same MLCIs possibly related through the same relations. Out of this cluster, a concept is formed by calculating the least common subsumer of the three ABoxes (concept formation task) as follows:

$$C13 \sqsubseteq \exists has\_Part.HorizontalBar \sqcap \exists has\_Part.Jumper$$

Such a new formed concept (C13) has only two properties (i.e., horizontal bar and jumper), because other properties or relations (e.g., the athlete name, the pole) are not common to all the three considered ABoxes. Once the concept C13 has been formed, we perform the concept enhancement via external knowledge sources. The goal of this task is to probe external knowledge sources in order to find definitions of concepts similar to the unnamed concept just formed (C13). First, C13 is matched against the domain ontology TBox, in order to find similar concepts already included in the ontology, if any. As a result, the *JumpingEvent* concept is found, which is the closest matching concept due to absence of a more specific concept for pole vault or high jump. Subsequently, probe queries are specified using the knowledge in C13 and in the *JumpingEvent* concept, in order to collect back a set of names and, possibly definitions, of external concepts matching C13. In our

Document	**ABox**
 Vazhipali Suresh Surekha (India) in action (Rahul Pawar)	pole_1 : Pole jumper_1 : Jumper horizontal_bar_1 : HorizontalBar athlete_name_1 : AthleteName (horizontal_bar_1, jumper_1) : near (jumper_1, pole_1) : touches
 Svetlana Feofanova (RUS) (Serafim Tracharis)	jumper_2 : Jumper horizontal_bar_2 : HorizontalBar athlete_name_2 : AthleteName (horizontal_bar_2, jumper_2) : near
	pole_3 : Pole jumper_3 : Jumper horizontal_bar_3 : HorizontalBar (horizontal_bar_3, jumper_3) : near (jumper_3, pole_3) : touches

Table 8: Input of the enrichment activity

case, by probing the *Yahoo* and *or Wikipedia* directory taxonomies looking for Jumping Event matching concepts, the following candidate concept names are retrieved:

$$Athletics, High\_Jump, Long\_Jump, Pole\_Vault, Triple\_Jump$$

Passing to concept definition, the formed concept C13, the set of retrieved candidate concepts listed above as well as the three images and the ABoxes of Table 8 that triggered concept learning are displayed to the ontology expert. The ontology expert, through a provided graphical interface, is able to interactively define the new concept and revise it. The ontology expert is not only able to perform changes, but experiment with them. In our example, suppose that the ontology expert has decided (by looking at the three images) that the new concept must describe a pole vault event. Possible actions that the ontology expert can perform for defining this new concept are:

1. To add a relation to the C13 concept, stating that the pole is a component of the new pole vault concept under definition, by exploiting information in the considered ABoxes. This action will produce a new concept, C14 with the following definition:
   $C14 \sqsubseteq \exists has\_Part.HorizontalBar \sqcap \exists has\_Part.Jumper \sqcap \exists has\_Part.Pole$

2. To find matching concepts for C14 in the ontology.

3. To discovery other candidate concepts by probing external sources for C14, if required.

As *Jumping_Event* is still the most similar concept retrieved in the actual ontology for C14, no further probing of external sources is performed. The ontology expert decides to name the new concept *Pole_Vault*; moreover, he decides to specify that it as a subclass of the *Jumping_Event* concept in the ontology. Once the ontology expert has defined the new concept and has selected the location of the new concept in the ontology, validation is performed. If the new concept is consistent with the ontology, then it is assimilated in order to become part of the ontology TBox. When the new concept becomes part of the TBox, the corresponding ABoxes will populate the ontology. This population occurs by sending these ABoxes back to the interpretation phase for a new evolution pattern selection (i.e., P1 or P2).

6.4 Measurable objectives

Measurable objectives for the ontology enrichment activity concern the ability of the enrichment activity to learn a new MLCs, HLCs and relations. In order to perform such an evaluation, the following scenario is proposed: starting from a "complete" version of the domain ontology, 20% of the concepts are hidden, i.e., removed from the ontology. These hidden concepts must contain concepts that are HLCs, MLCs and relations (33% from each). Using as input the manually annotated corpus (which constitutes the ideal input expected from the information extraction toolkit), documents whose explanations involve hidden concepts are selected. Explanations are removed from these documents and the documents are processed by the evolution toolkit. Each of these documents will trigger the selection of evolution patterns P3 and P4, leading to an enrichment task. At the end, the performance will be measured according to the percentage of the concepts recovered by the evolution toolkit.

7 ONTOLOGY COORDINATION

Coordination is the activity of producing a log of the changes introduced into the new evolved version of the ontology with respect to the input ontology and of defining and updating mappings between the evolved ontology and other relevant external knowledge sources. The first goal is achieved by means of the ontology versioning and management task, which has the aim of producing both a new version of the ontology storing the results of the enrichment activity on the previous version and a log of the changes that have been implemented during the enrichment activity. The input of the ontology versioning is constituted by:

- the ontology used as input of the evolution process (in the following referred to as $\mathcal{O}_{input}$) together with its associated mapping knowledge;
- the intermediate ontology, i.e., the working copy of the ontology produced by the enrichment activity;
- the change log, i.e, a document containing the registration of all the operations executed on the ontology during the enrichment activity.

The output of the ontology versioning and management task is constituted by

- the new evolved ontology (in the following $\mathcal{O}_{output}$);
- an optimized copy of the change log.

Finally, the evolved ontology is aligned with the external knowledge sources by defining appropriate mappings. When a concept is modified during the enrichment activity, possibly existing mappings for it have to be updated appropriately to reflect the modifications applied. The coordination activity is articulated in the following tasks:

- *Ontology versioning and management.* Ontology versioning and management has the goal of producing the new evolved ontology $\mathcal{O}_{output}$ (starting from the intermediate ontology) together with the optimized *change log*, that is the log containing the minimal set of operations required to obtain $\mathcal{O}_{output}$ starting from $\mathcal{O}_{input}$.
- *Ontology alignment.* The goal of ontology alignment is to produce mapping knowledge referred to the concepts of $\mathcal{O}_{output}$. This requires the definition of new mappings for the new concepts inserted into $\mathcal{O}_{output}$ as well as the maintenance of already existing mappings involving modified concepts or other concepts that are either directly or indirectly related to a modified/inserted concept in the intermediate ontology.

7.1 Ontology versioning and management

Ontology versioning is the activity of releasing a new version V_1 of an ontology starting from a previous version V_0, by keeping track of the modifications of V_1 with respect to V_0. In BOEMIE, ontology versioning works on the input ontology $\mathcal{O}_{input}$ and releases the evolved ontology $\mathcal{O}_{output}$. The changes in $\mathcal{O}_{output}$ with respect to $\mathcal{O}_{input}$ are stored in a *change log* during the ontology enrichment activity. In particular, the change log is a sequential file that contains a record for every operation performed on the intermediate ontology during the enrichment activity and that constitutes a input for ontology versioning. The change log is optimized during the versioning task to obtain an optimized change log as a *transformation set.* As described in [Noy and Klein, 2004], a transformation set provides a minimal set of operations to transform a version of an ontology into a new version. A transformation set differs from a simple log of changes because i) it does not contains all the operations performed to transform the old into the new version of the ontology, but only a minimal subset; ii) it requires only a partial ordering in which all the operations of concept creation are performed before all the other operations. Starting from a change log is possible to extract an *optimized change log* as a transformation set, by identifying all basic changes and thus by finding and discarding all redundant changes. By storing an optimized version of the change log, the recovery of the new version of the ontology starting from the old one, when required, can be executed in an efficient way. The task of ontology management constitutes the "glue" for turning the activities of population, enrichment, and coordination, as well as the corresponding tools, into the desired evolution toolkit. In addition, ontology management control also ontology versioning, by maintaining the different versions of the ontology throughout the evolution process. The decision about which and how many versions of the ontology should be kept in the ontology repository has to be taken at the bootstrapper level, where a complete view of the whole bootstrapping process cycles is possible. For example, the previous version (i.e., $\mathcal{O}_{input}$) could be maintained as the reference version until re-annotation of the resources has been completely executed, and then it could be discarded. On the other hand, one could decide to keep a history of the ontology, by keeping the various versions in the repository. As noted before, during ontology evolution, changes may generate inconsistencies in the already annotated resources. This generally requires the re-annotation of certain resources at the meta-data level exploiting the changes as these are recorded in the ontology versioning task.

7.2 Ontology alignment

The goal of aligning the BOEMIE ontology with the external knowledge sources is to maintain over time a correspondence among the concepts in the BOEMIE domain ontology and other semantically related concepts in the external sources. These correspondences are called *mappings* and are used in order to support the process of concept discovery in the enrichment activity of ontology evolution. Moreover, mappings provide useful external information semantically related to the concepts of the BOEMIE ontology. As an example, if the concept $Pole\_Vault$ of the BOEMIE ontology has a mapping with the concept $Pole\_Vault$ of the Google web directory system [7], we can exploit the documents in the corresponding Google directory to retrieve new resources about pole vault events.

7.2.1 Definition of new mappings

In this section, we examine the case of new mappings definition, for new concepts that have been inserted in $\mathcal{O}_{output}$. Establishing mappings between two ontologies means that for each entity (e.g., concept, property) in one ontology we try to find a corresponding entity in the second ontology, with the same or the closest intended meaning; usually this correspondence is expressed by one to one functions. Mappings can be established after an analysis of the similarity of the entities in the compared ontologies, according to a certain metric. It is important to note that the mapping process does not modify the involved ontologies but produces, as the output, a set of correspondences on top of the involved ontologies [Interop, 2004; Ehrig, 2004]. The focus of this definition is on the notion of similarity between the meanings of different ontology elements expressed by a value in the range [0,1]. Another interesting note is that some systems provides one-to-one mappings between ontologies, while some other provide also one-to-many mappings. These latter approaches address the fact that an element in an ontology may have more than one similar element in another ontology.

More formally, given two ontologies $\mathcal{O}$ and $\mathcal{O}'$ and two concepts $C \in \mathcal{O}$ and $C' \in \mathcal{O}'$, the mapping $m_{C,C'}$ between C and C' is defined as a 4-tuple of the form:

$$m_{C,C'} = \langle C, C', R, V \rangle$$

where R denotes the semantic relation holding between C and C', while V denotes the confidence value associated with R. In most cases, the notion of similarity between two concepts is interpreted as a degree of confidence about the equivalence of the two concepts. Thus, it is often the case that mappings represent equivalence relations among the concepts (i.e., R is $\equiv$) with an associated degree of confidence V in the range [0,1].

The process of discovering mappings between two or more heterogeneous ontologies, by exploiting automatic or semi-automatic methods and techniques is referred to as *ontology matching*. The process of ontology matching can be summarized in three main steps, as shown in Figure 45.

The first step is the acquisition of the ontologies to be matched that are represented by means of an internal model for the matching purposes. The second step concerns the analysis of the

[7] http://www.google.com/Top/Sports/Track_and_Field/Pole_Vault/

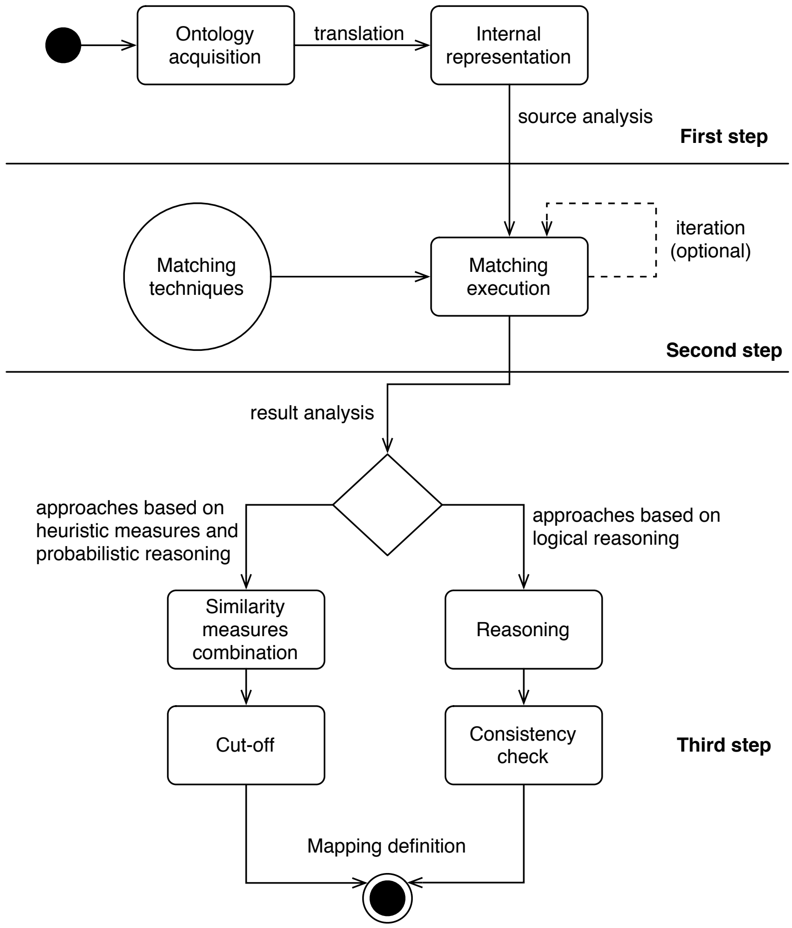

Figure 45: High-level representation of the matching process

ontologies and by the execution of the matching procedures. This step is different depending on the algorithm that is adopted. In general, several matching techniques can be adopted for ontology matching. For this reason, this step is often iterated several times in order to refine the results obtained in the previous executions. In the third step, the mappings among ontology elements are determined. Here we can have different tasks depending on the type of matching process that has been performed. In the case of the matching techniques that produce a similarity value between the ontology elements, such as the approaches based on heuristic rules or probabilistic reasoning [Kalfoglou and Schorlemmer, 2003; Doan et al., 2002], the different similarity measures are combined into a comprehensive value of similarity for the ontology elements and the results that are not considered to be relevant are cut off. In the case of the matching techniques that produce a semantic relation between the ontology elements, such as the approaches based on logical reasoning, a reasoning task is performed for inferring new relations out of the matching results and for checking the consistency of the mappings [Giunchiglia et al., 2004; Giunchiglia and Shvaiko, 2003]. Finally, a set mappings are determined between the input ontology elements.

In the state of the art, many ontology matching systems have been developed and there are complete and detailed surveys on ontology matching approaches and tools. For example, in [Shvaiko and Euzenat, 2005; Ferrara, 2005] a classification of different tools is proposed. In [Noy, 2004; Kalfoglou and Schorlemmer, 2003] the authors describe a large number of different tools that are presented by taking into account different sets of requirements and different points of view. The matching service in BOEMIE will provide a wide spectrum of functionalities and techniques for ontology matching that can be used for the purpose of aligning the BOEMIE ontology with external knowledge sources. In the examples of the present section, we describe the matching activities by referring to one specific set of techniques that can be used to address the alignment goal.

In BOEMIE, the new concepts inserted in $\mathcal{O}_{output}$ are matched against the external knowledge sources by means of the matching service. The configuration of the matching procedure depends on the features of the concept to be matched as well as on the features of the external sources. In particular, the matching parameters are affected by the semantic complexity of the concepts that are matched. The notion of *semantic complexity* has been introduced in [Ehrig and Sure, 2004], to describe different levels of complexity at which an ontology can be examined for matching purposes. Semantic complexity depends on the number and type of constructs that are used in a concept definition. If we match two concepts featured only by a name and a small set of taxonomic relations the semantic complexity of the matching process is lower than the case of concepts described by properties and a high number of property constraints. The matching service of BOEMIE, which support four different matching models to cope with increasing levels of semantic complexity in concept description. The configuration of the matching service is usually a manual activity, but there are several heuristics that have been proposed to automatically configure the matcher that can be used to reduce the manual effort required in the coordination process [Mochol et al., 2006].

Given a set $N = \{C_1, C_2, \ldots, C_n\}$ of new concepts inserted in $\mathcal{O}_{output}$ during the ontology enrichment activity, we build a target ontology $\mathcal{O}_{target}$ that contains each concept $C_i \in N$ and all the concepts of $\mathcal{O}_{output}$ that are have a semantic relation with a concept in N. By concept context we define the set of properties, constraints, and semantic relations featuring a concept. Once $\mathcal{O}_{target}$ is defined, it is matched against each external knowledge source K of interest. To this end, wrapping techniques can be used to acquire the external sources metadata and translate them into the internal representation format required by the matching service. This acquisition process corresponds to the first step of the matching process shown in Figure 45. Passing from first to second step, the two ontologies are analyzed in order to determine their respective level of semantic complexity and, thus, configuring accordingly the matching service. In the second step of the matching process, the matching activity is executed to find a set of candidate mappings between the concepts in $\mathcal{O}_{target}$ and the concepts in the external sources. Then, in the third step, mappings are validated, in order to cut-off those mappings that are not considered to be correct and/or relevant. In order to perform such an activity, it is possible to use an heuristic approach, such as a threshold-based mechanism [Castano et al., 2006; Doan et al., 2002], or a reasoning-based mechanism [Meilicke et al., 2006]. Mappings that pass the validation check are finally inserted in the mapping knowledge repository.

7.2.2 Maintenance of existing mappings

In this section, we discuss the maintenance of existing mappings to reflect the modifications applied to existing concepts. In particular, a mapping maintenance activity is required on existing mappings that involve concepts included in the context of a modified/inserted concept. To clarify this, consider the case of a new concept C inserted in the ontology during the enrichment activity. As a consequence of this, also new relations could have been inserted between C and other concepts of the ontology (e.g., suppose to add, for the new concept C, the axiom $C \sqsubseteq C'$, where C' is a concept already present in the ontology). The context of C' has changed, since we now know that there is a subclass C of C'. This change in the context configuration requires to update, as an indirect effect, also the mappings already stored for C'. A strategy that is adopted to reduce the number of mappings that need to be updated is to evaluate a *measure of change* of a concept C when passing from $\mathcal{O}_{input}$ to $\mathcal{O}_{output}$. After the enrichment activity, O_{output} may contain new or modified concepts, properties, constraints or semantic relations. If at least one of the modifications involves a concept for which mappings already exist in the mapping knowledge, these mappings have to be updated as well. However, the entity of the modification on the existing mappings could be considered not relevant in many cases. In general, a potentially high number of mappings could be involved in the maintenance, and a strategy is required to evaluate if all mappings really need to be updated. The idea is to update the mappings only in the case of "relevant modifications" produced by the enrichment activity. In order to evaluate the relevance of modifications, we introduce the notion of *measure of change* $M_{Ch}(C)$ of a concept C when passing from the definition of C in $\mathcal{O}_{input}$ to the modified definition $\overline{C}$ in $\mathcal{O}_{output}$. The measure of change of a concept C is defined in terms of the level of matching of the two definitions of C in $\mathcal{O}_{input}$ and $\mathcal{O}_{output}$, respectively. In particular, the measure of change of a concept C is calculated as $M_{Ch}(C) = 1 - SA(C, \overline{C})$, where $SA(C, \overline{C})$ denotes the semantic affinity of C and $\overline{C}$. A threshold mechanism can be used to automatically select those mappings that need to be updated. We note also that the information about the measure of change can be exploited also to choose when the concepts used by the annotation tools need to be updated.

7.3 Example: ontology versioning and alignment

In this section, we provide a comprehensive example of the ontology coordination activity, by considering the ontology versioning task as well as the alignment of the ontology with respect to the external knowledge sources. In our example, we suppose that two new concepts $Pole\_Vault$ and $High\_Jump$ have been inserted in the intermediate ontology (see Section 6).

In order to show an example of transformation set applied to the insertion of $Pole\_Vault$ and $High\_Jump$ in the ontology, suppose that the ontology expert actions during the concept definition task of ontology enrichment are the following. He first defines $High\_Jump$ as a subclass of $Jumping\_Event$ and $Pole\_Vault$ as a subclass of $High\_Jump$. Then, he realizes that this is wrong and he changes the definition of $Pole\_Vault$ by asserting that it is a subclass of $Jumping\_Event$. In such a case, the change log of this operations contains the following records:

```
1) Add (High_Jump subclass Jumping_Event)
2) Add (Pole_Vault subclass High_Jump)
3) Delete (Pole_Vault subclass High_Jump)
4) Add (Pole_Vault subclass Jumping_Event)
```

The role of transformation set is to identify all superfluous operations in the change log and to provide a minimal set of operations to be performed in order to obtain the final ontology version starting from the initial one. Back to the example, operation 2 and operation 3 are superfluous, since the same axiom is first added and then deleted in the ontology. Thus, by applying the transformation set, the optimized change log becomes as follows:

```
1) Add (High_Jump subclass Jumping_Event)
2) Add (Pole_Vault subclass Jumping_Event)
```

When the number of modifications to the ontology is high, the transformation set is useful to reduce the number of operations to be executed in order to recover $\mathcal{O}_{output}$ starting from $\mathcal{O}_{input}$, when required.

For the purpose of the alignment of the ontology with the external knowledge sources, the target ontology $\mathcal{O}_{target}$ is defined as shown in Figure 46. The target ontology contains the con-

$$
\begin{array}{rcl}
Pole & \sqsubseteq & SportEquipment \\
Foam\_Mat & \sqsubseteq & SportEquipment \\
Horizontal\_bar & \sqsubseteq & SportEquipment \\
Jumping\_Event & \sqsubseteq & Event \sqcap \exists hasPart.Jumper \\
Pole\_Vault & \sqsubseteq & Jumping\_Event \sqcap \\
& & \exists hasPart.Pole \sqcap \\
& & \exists hasPart.Horizontal\_Bar \sqcap \\
& & \exists hasPart.Foam\_Mat \\
High\_Jump & \sqsubseteq & Jumping\_Event \sqcap \\
& & \exists hasPart.Horizontal\_Bar \sqcap \\
& & \exists hasPart.Foam\_Mat
\end{array}
$$

Figure 46: The target ontology

cepts $High\_Jump$ and $Pole\_Vault$, together with all the concepts that are related to them. Each concept in the target ontology is provided with its context (i.e., the set of axioms in which it is involved) [8]. As an example of external knowledge sources, we use the OWL representation of two

[8]The context of a concept in the target ontology contains also the axioms that implicitly affects the concepts in the ontology (e.g., the axioms that are inherited from taxonomic relations). This information can be easily acquired by means of standard reasoning services.

web directory taxonomies and two web sites. The graphical representations of these resources is shown in Figures 47 and 48, respectively.

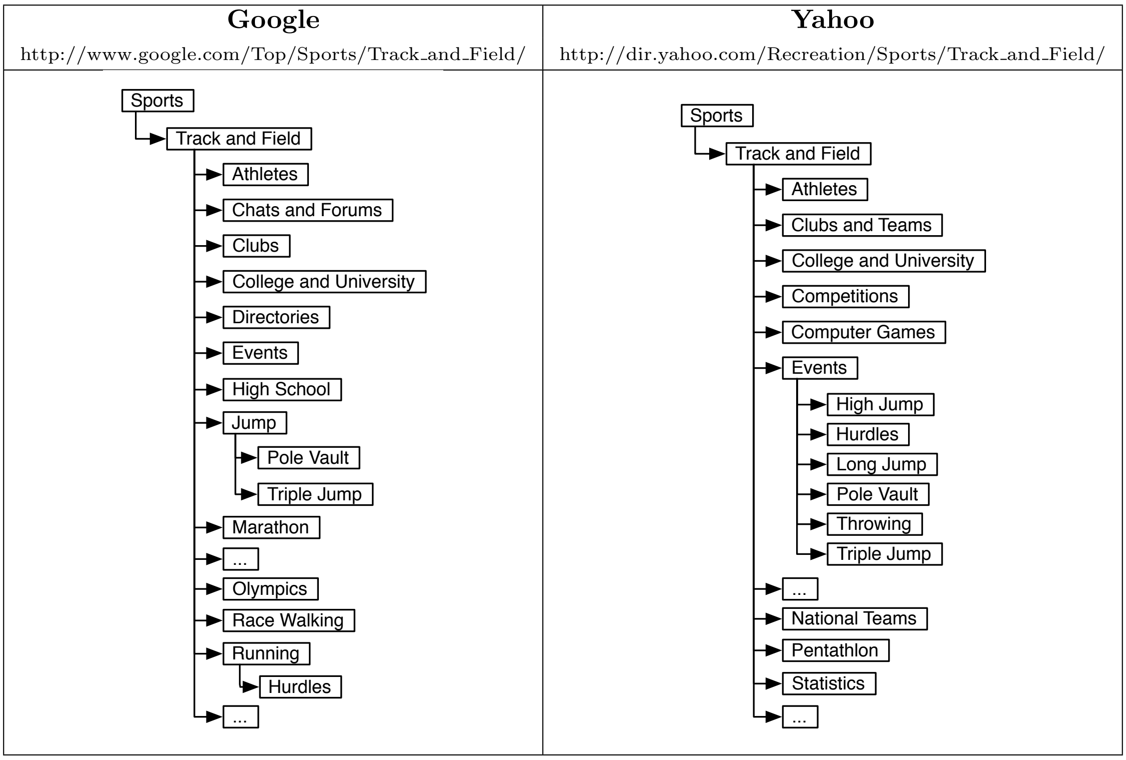

Figure 47: Graphical representation of the external knowledge sources (1)

The concepts provided by the external sources are featured only by taxonomic relations. According to this analysis, we configure the matching service for the example to consider only the affinity among the concept names and the taxonomic relations in the concept contexts. Then, we adopt a threshold-based mechanism (e.g., threshold equal to 0.5) to validate the mappings. The final results of the matching procedure are shown in Table 9

Target ontology	**Google**	**Semantic Affinity**
Pole_Vault	Pole_Vault	1.0
High_Jump	Jump	0.8
Jumping_Event	Jump	0.6
Target ontology	**Yahoo**	**Semantic Affinity**
Pole_Vault	Pole_Vault	1.0
High_Jump	High_Jump	1.0
Target ontology	**Wikipedia**	**Semantic Affinity**
Pole_Vault	Pole_Vault	1.0
High_Jump	High_Jump	1.0
Target ontology	**Encyclopedia Britannica**	**Semantic Affinity**
Pole_Vault	Pole_Vault	1.0
High_Jump	High_Jump	1.0
Jumping_Event	Jumping_Event	1.0

Table 9: Results of matching the target ontology against external sources

According to these results, the following mappings are defined, by considering the semantic affinity value as a measure of confidence associated with the equivalence relation of corresponding

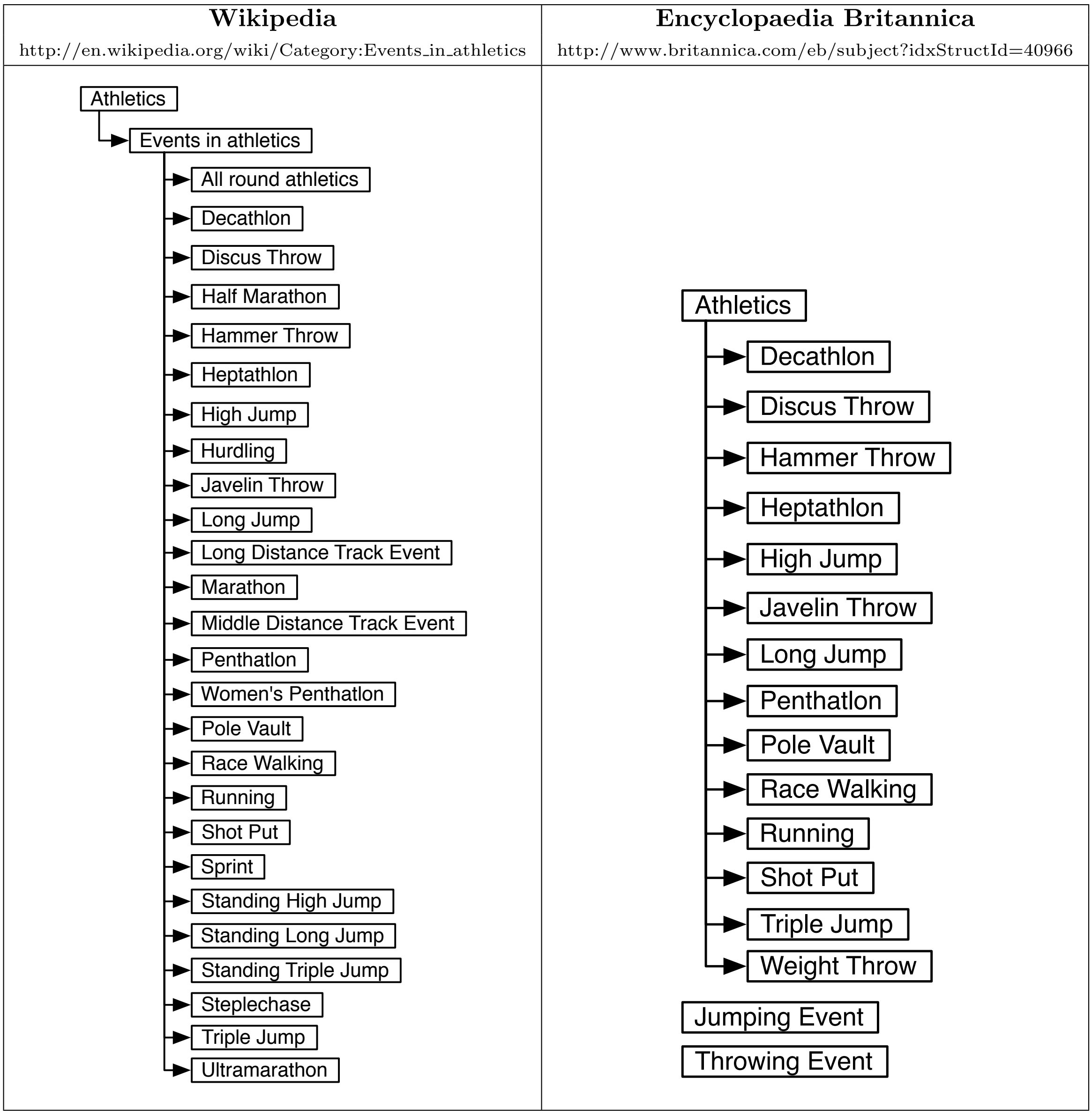

Figure 48: Graphical representation of the external knowledge sources (2)

concepts.

$$M01 = \langle Pole\_Vault, google\#Pole\_Vault, \equiv, 1.0\rangle$$

$$M02 = \langle High\_Jump, google\#Jump, \equiv, 0.8\rangle$$

$$M03 = \langle Jumping\_Event, google\#Jump, \equiv, 0.6\rangle$$

$$M04 = \langle Pole\_Vault, yahoo\#Pole\_Vault, \equiv, 1.0\rangle$$

$$M05 = \langle High\_Jump, yahoo\#High\_Jump, \equiv, 1.0\rangle$$

$$M06 = \langle Pole\_Vault, wikipedia\#Pole\_Vault, \equiv, 1.0\rangle$$

$$M07 = \langle High\_Jump, wikipedia\#High\_Jump, \equiv, 1.0\rangle$$

$$M08 = \langle Pole\_Vault, britannica\#Pole\_Vault, \equiv, 1.0\rangle$$

$$M09 = \langle High\_Jump, britannica\#High\_Jump, \equiv, 1.0\rangle$$

$$M10 = \langle Jumping\_Event, britannica\#Jumping\_Event, \equiv, 1.0\rangle$$

The concept $Jumping\_Event$, which was already present in the ontology, is involved in mappings $M03$ and $M10$. These new mappings are used to update the mappings eventually stored in the mapping knowledge repository between $Jumping\_Event$ and the concepts of Google ($M02$) and Encyclopedia Britannica $M10$.

As an example of a possible optimization of mapping maintenance, we consider the concept $Jumping\_Event$. This concept was already present in $\mathcal{O}_{input}$, but, in $\mathcal{O}_{output}$ we have inserted two more subclasses of it, namely $High\_Jump$ and $Pole\_Vault$. This modification affects the mapping maintenance as shown above. In this case, we are interested in evaluating the measure of the change of $Jumping\_Event$ over time, in oder to understand if the mappings in which it is involved have to be updated. The semantic affinity between the new and the old version of $Jumping\_Event$ is 0.84. Thus, the measure of change of the two concepts is evaluated as:

$$M_{Ch}(Jumping\_Event) = 1 - 0.84 = 0.16$$

Through a threshold-based mechanism, the system automatically can exploit the calculated measure of change in order to determine if the mappings associated with changed concepts (in our case, $Jumping\_Event$) need to be updated. In the case of the example, the change value is low and mappings will not be updated. In general, high values of the change measure suggest to update mappings, while for low values of change mappings are kept unaltered.

7.4 Measurable objectives

Measurable objectives for the ontology coordination activity provide an evaluation of the coordination techniques with respect to the quality of the ontology alignment and ontology versioning. For what concerns ontology alignment, several evaluation strategies have been proposed to measure the quality of ontology matching techniques [Euzenat et al., 2006]. These techniques are based on the idea of evaluating the precision and recall of the mappings automatically retrieved by a matching system with respect to a set of expected mappings. More in detail, precision is defined as the ratio of the number of matching concepts automatically found by the matching system considered as relevant to the total number of matching concepts found. Recall is defined as the ratio of the number of relevant matching concepts automatically found by the system to the total number of matching concepts (i.e., mappings) expected. In order to use these metrics in BOEMIE, we need to define a set of mappings that are expected. To this end, we hide more than 20% of the mappings retrieved automatically by the system. Then, we manually define mappings among the concepts involved in the hidden mappings and the external knowledge sources. These manually defined mappings will be the set of expected mappings. The target of the evaluation is to execute the alignment on the concepts involved in order to achieve more than 70% of precision and recall with respect to this set of expected mappings. Ontology versioning will be evaluated by recovering different versions of the ontology starting from the previous ones and the change logs, in controlled experiments.

8 ARCHITECTURE FOR ONTOLOGY EVOLUTION

The proposed methodology is realized as an open architecture and implemented through an evolution toolkit comprising two evolution tools, namely, the population and enrichment tool, and the coordination tool, as well as two support services, namely, the reasoning service and the matching service. The open architecture of the methodology evolution toolkit is shown in Figure 49.

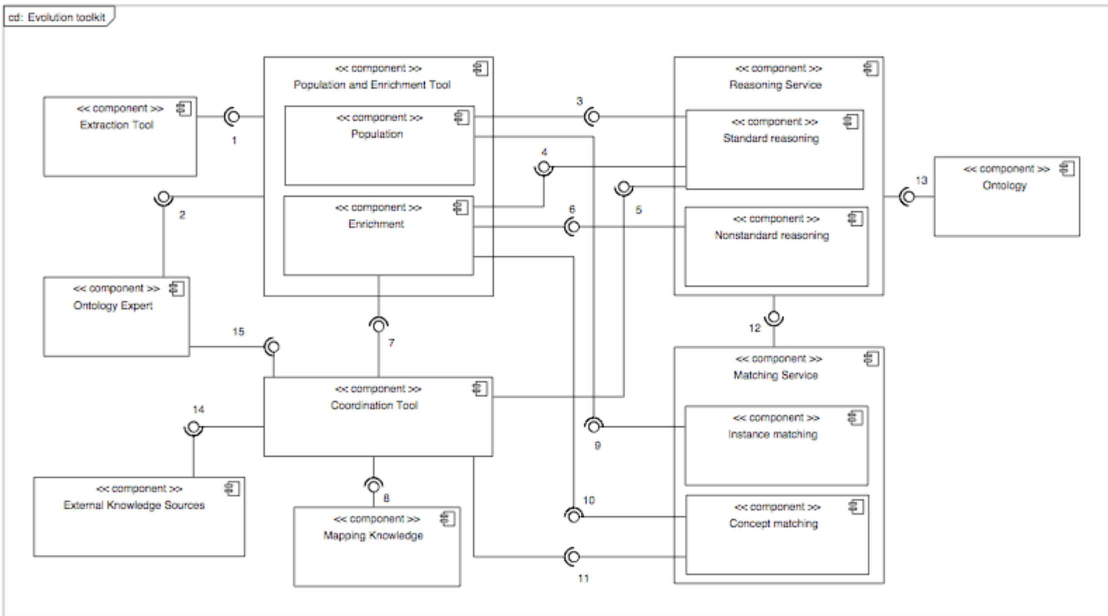

Figure 49: Evolution toolkit component and interactions diagram

8.1 Evolution tools and interactions

The evolution toolkit is conceived to support all the phases of the ontology evolution methodology. The ontology expert will be provided with immediate access to the domain ontology, through a user-friendly interface which provide him with the tools needed to correctly evolve the ontology. The ontology evolution toolkit is developed to provide an interface for managing the ontology. The interface of the tool is based on the existing ontology editors that are extended with new functionalities in order to not only visualize and edit the ontology elements, but also to enrich and coordinate independent and heterogeneous ontologies. The idea is to support the ontology expert in maintaining an ontology during the entire bootstrapping process in the context of an interactive environment. The designer will be able to acquire previously defined ontologies and align them with the current ontology. He will also be supported in performing the population based on the information extracted from the resources. The evolution toolkit constitutes the “glue” for turning the individual components into a comprehensive architecture. In this section, the BOEMIE evolution tools that compose the toolkit are described, together with their interactions.

8.1.1 Population and enrichment tool

The population and enrichment tool is responsible for the activities of population (patterns P1 and P2) and enrichment (patterns P3 and P4). The diagram of the tasks executed by the tool is shown in Figure 50. In order to perform the required tasks, the population and enrichment tool interacts with the extraction tool, the ontology expert and the evolution services. In particular, the population and enrichment tool interacts with the extraction tool (interaction 1) in order to acquire the input for the evolution process, i.e., the extracted instances and low-level features, and to provide to the extraction the feedback information concerning the enrichments of the ontology (patterns P3 and P4). Once the input is acquired, the population and enrichment tool performs

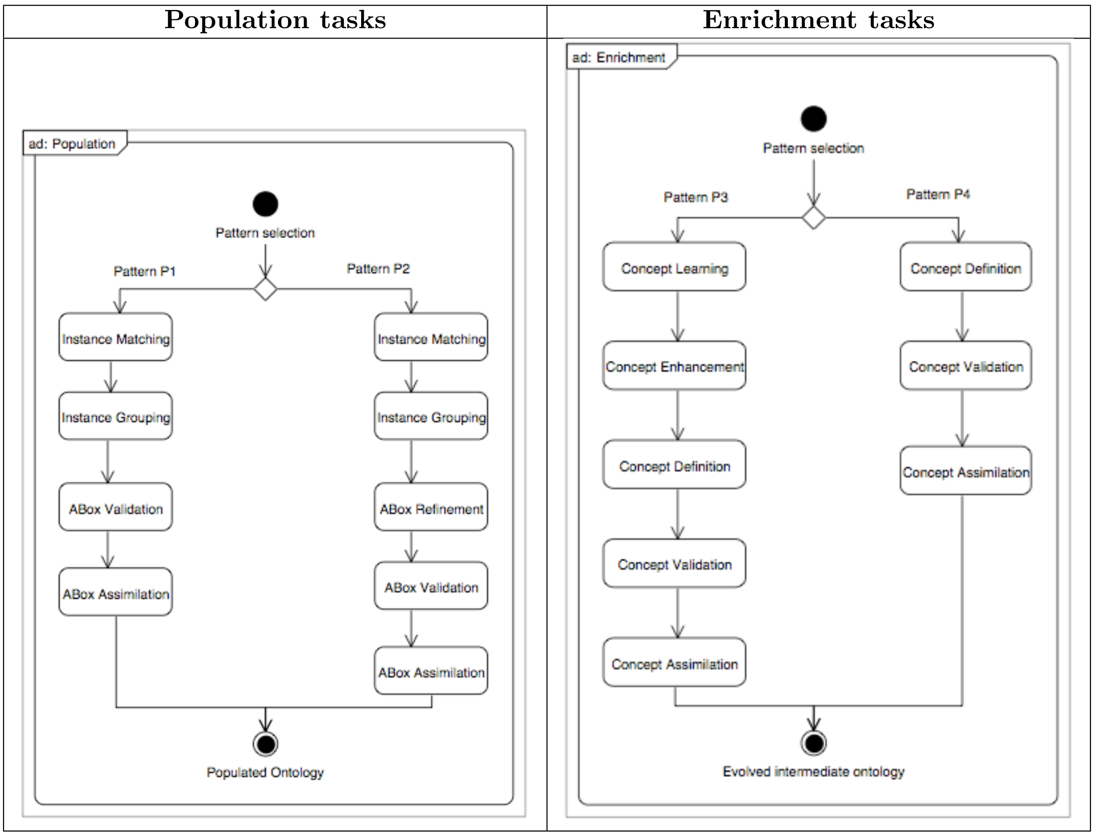

Figure 50: Population and enrichment tasks

pattern selection. According to the selected pattern, the population tasks (P1 and P2) or the enrichment tasks (P3 and P4) are activated. In case of population, the tool interacts with the instance matching component of the matching service (interaction 9) in order to perform instance matching. To this end, the matching service can exploit the reasoning service in order to support the instance matching by nonstandard reasoning (interaction 12). Given the results of instance matching, the population and enrichment tool executes instance grouping. In case of pattern P2 only, also ABox refinement is executed. During this activity, if the instance matching result is not sufficient to automatically refine the ABox, the tool interacts with the ontology expert in order to refine the ABox (interaction 2). Finally, the resulting ABox is validated and assimilated into the ontology by interacting with the standard reasoning service (interaction 3). In case of enrichment, the population and enrichment tool performs concept learning by means of the clustering and concept formation tasks. This last task is executed with the support of nonstandard reasoning (interaction 6). The subsequent activity is concept enhancement, which is executed by interacting with the concept matching component of the matching service (interaction 10). Again, the matching service interacts with the standard reasoning service in order to acquire the concept definitions to be matched. The new concept is then defined by interacting with the ontology expert (interaction 2), by inspecting and interactively working on the output produced by the previous tasks. The new concept is then validated and inserted into the domain ontology by interacting with the standard reasoning (interaction 4). Moreover, the support of the coordination tool (interaction 7) is exploited in order to align the new concept with the other modules of the semantic model.

8.1.2 Coordination tool

The coordination tool is responsible for the alignment of the new evolved ontology with the external knowledge sources and for ontology versioning. The diagram of the tasks executed by the tool is shown in Figure 51. The coordination activity is triggered by the results of the enrichment activity (interaction 7). The coordination tool interacts with the reasoning service in order to acquire the

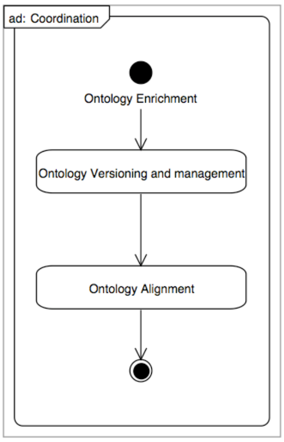

Figure 51: Coordination tasks

definition of the new concept inserted in the intermediate ontology (interaction 5). First, ontology versioning is executed in order to optimize the change log that describes the operations that have been executed during the enrichment activity in order to enrich the ontology. Then, ontology alignment is executed. Both mapping definition and update need an interaction with the ontology expert (interaction 15) in order to set the parameters used for ontology matching as well as for the validation and refinement of the matching process. A measure of change of the new version of the ontology with respect to the old one is calculated to choose which mappings already existing in the mapping knowledge need to be updated, to avoid a complete update of the mappings every time. The ontology alignment is executed in order to define new mappings and/or update existing ones, by exploiting the results provided by the matching service (interaction 11). In order to invoke the matching service, the coordination tool determines the target ontology and interacts with the external knowledge sources in order to choose the sources to be probed (interaction 14). Finally, the new version of the ontology and the optimized log of changes are released and the coordination tool interacts with the mapping knowledge in order to store the new mappings (interaction 8).

8.2 Evolution services

8.2.1 Reasoning service

The standard reasoning services required for the tasks mentioned in previous sections are provided by the description logic system RacerPro (Renamed ABox and Concept Expression Reasoner PROfessional). RacerPro implements a TBox and ABox reasoner for the logic $\mathcal{SHIQ}$ (see above) [Haarslev and Möller, 2003a,b]. RacerPro was the first full-fledged ABox description logic system for a very expressive logic and is based on optimized sound and complete algorithms. RacerPro also implements a decision procedure for modal logic satisfiability problems (possibly with global axioms).

The description logic (DL) $\mathcal{SHIQ}$ [Horrocks et al., 2000b] extends the logic $\mathcal{ALCNH}_{R^+}$ [Haarslev and Möller, 2000b] by additionally providing qualified number restrictions and inverse roles. Using the $\mathcal{ALCNH}_{R^+}$ naming scheme, $\mathcal{SHIQ}$ could be called $\mathcal{ALCQHI}_{R^+}$ (pronunciation: ALC-choir).

The ABox consistency algorithm implemented in the RacerPro system is based on the tableau calculus presented in [Haarslev and Möller, 2000b]. For dealing with qualified number restrictions and inverse roles, the techniques introduced in the tableau calculus for $\mathcal{SHIQ}$ [Horrocks et al., 2000b] are employed. Nominals are currently approximated using concept names.

For practical systems, optimized search techniques are required in order to guarantee good average-case performance. The RacerPro architecture incorporates the following standard optimization techniques: dependency-directed backtracking [Stallman and Sussman, 1977] and DPLL-style semantic branching (see [Freeman, 1995] for an overview of the literature). Among a set of new optimization techniques, the integration of these techniques into DL reasoners for concept consistency has been described in [Horrocks, 1997]. The implementation of these techniques in the ABox reasoner RacerPro differs from the implementation of predecessor systems such as FaCT, which provide TBox reasoning only. The latter systems have to consider only so-called "labels" (sets of concepts) whereas an ABox prover such as RacerPro has to explicitly deal with individuals. ABox optimizations are also explained in [Haarslev and Möller, 2004; Möller et al., 2006].

The integration of techniques for representing "concrete domains" (e.g. linear inequalities between real numbers) on the role fillers of an individual has been investigated in [Haarslev et al., 2001b]. In addition, optimization techniques for dealing with qualified number restrictions [Haarslev and Möller, 2001a] are integrated into RacerPro.

The techniques for TBox reasoning described in [Haarslev et al., 2001b] (marking and propagation as well as lazy unfolding) are also supported by RacerPro. As indicated in [Haarslev and Möller, 2004], the architecture of RacerPro is inspired by recent results on optimization techniques for TBox reasoning [Horrocks and Patel-Schneider, 1998], namely transformations of axioms (GCIs) [Horrocks and Tobies, 2000], model caching [Haarslev and Möller, 2000a] and model merging [Horrocks, 1997] (including so-called deep model merging and model merging for ABoxes [Haarslev et al., 2001a]). RacerPro also provides additional support for very large TBoxes (see [Haarslev and Möller, 2001b]).

RacerPro is implemented in Common Lisp and is available for research purposes as a server program which can be installed under Mac OS, Linux and Windows (http://www.racer-systems.com). Client programs can connect to the RacerPro DL server via a TCP/IP interface based on sockets. Client-side interfaces for Java, Common Lisp, and C/C++ are available. For communication via HTTP RacerPro supports the OWL and DIG standards. As a query language, RacerPro supports nRQL [Haarslev et al., 2004; Wessel and Möller, 2005, 2006] and OWL-QL (with grounded semantics [Galinski et al., 2005; Kaplunova et al., 2006]). In addition to terminological reasoning, RacerPro also provides support for qualitative spatial reasoning [Wessel and Möller, 2006].

8.2.2 Standards for ontology specification

Current standards for ontology specification include the OWL Web Ontology Language and its three sublanguages, namely OWL Lite, OWL DL and OWL Full. OWL Lite represents the least expressive variant of OWL, it supports cardinality constraints with values 0 or 1 and does not allow to use negation and union for the definition of concepts. OWL DL represents a syntactic variant of the DL language $\mathcal{SHOIN}(\mathbf{D})$. OWL Full is the most expressive variant of OWL, with the special feature that it provides meta-modelling. OWL has been extended to OWL 1.1 to support the DL language $\mathcal{SROIQ}(D_n)^-$ allowing for more expressivity w.r.t. roles, and is the language supported by the DIG 2.0 interface (see also below). Another standard for the specification of ontologies is the Semantic Web Rule Language SWRL (see Section 4.1.3), which allows to add rules, more specifically DL-safe rules, on top of OWL DL or OWL Lite.

8.2.3 Standards for Invoking Reasoning Services

The continous standardization efforts by the DL Implementation Group (DIG), have the objective of specifying an interface (also known as DIG) such that access to the functionality of Description Logic reasoners is available through implementation-neutral mechanisms. The latest version (DIG 2.0) supports OWL 1.1 based on the DL language $\mathcal{SROIQ}(D_n)^-$.

- DIG and standard inference services: The standard inference services (see above) offered by a reasoner are accessible in DIG through TELL and ASK statements. Moreover extensions in DIG 2.0 allow for the retrieval of information about axioms or assertions that have been explicitly given through a TELL statement and retraction of information to withdraw previously added axioms or assertions.

- DIG and non-standard inference services: The supported non-standard inferences comprise approximation, LCS, GCS (Good Common Subsumer, which is a variant of LCS where the common subsumer for a set of concept does not need to be the least one), find matching concepts, find matcher and minimal rewriting.

- DIG and query answering: The ABox query language of DIG 2.0, makes no commitment to a certain semantics. The semantics depend on the reasoner. The standard semantics is implemented in the system QuOnto [Calvanese et al., 2005] (for a description logic of the DL-Lite family) whereas the grounded semantics is implemented in RacerPro [Wessel and Möller, 2005] and also Pellet [Sirin and Parsia, 2006] or KAON2 [Hustadt et al., 2004]. For all systems, an interface corresponding to the DIG 2.0 standard will be developed. A possibility to identify the fragment of the query language supported by a reasoner is foreseen in the DIG 2.0 protocol.

  In case a reasoner has to deal large result sets for queries, iterative query answering can help to improve performance. Therefore, in DIG 2.0, result sets can be retrieved iteratively using small chunks of tuples. A reasoner can indicate if resource consumption is likely to increase if further tuples are retrieved. In addition, DIG 2.0 supports instructions to let a query answering engine compute results proactively to support faster retrieval of subsequent chunks of triples.

8.2.4 Matching service

For the sake of ontology matching, we rely on the HMatch (http://islab.dico.unimi.it/hmatch) semantic matchmaker [Castano et al., 2006]. HMatch takes a target concept description c and an ontology O as input and returns the concepts in O which match c, namely the concepts with the same or the closest intended meaning of c. In HMatch, we perform concept matching through affinity metrics by determining a measure of semantic affinity in the range $[0, 1]$. A threshold-based mechanism is enforced to set the minimum level of semantic affinity required to consider two concepts as matching concepts. Given two concepts c and c', HMatch calculates a semantic affinity value $SA(c, c')$ as the linear combination of a linguistic affinity value $LA(c, c')$ and a contextual affinity value $CA(c, c')$. The linguistic affinity function of HMatch provides a measure of similarity between two ontology concepts c and c' computed on the basis of their linguistic features (i.e., concept names). For the linguistic affinity evaluation, HMatch relies on a thesaurus of terms and terminological relationships automatically extracted from the WordNet lexical system [Miller, 1995]. The contextual affinity function of HMatch provides a measure of similarity by taking into account the contextual features of the ontology concepts c and c'. The context of a concept can include properties, semantic relations with other concepts, and property values. The context can be differently composed to consider different levels of semantic complexity, and four matching models, namely, *surface*, *shallow*, *deep*, and *intensive*, are defined to this end. In the surface matching, only the linguistic affinity between the concept names of c and c' is considered to determine concept similarity. In the shallow, deep, and intensive matching, also contextual affinity is taken into account to determine concept similarity. In particular, the shallow matching computes the contextual affinity by considering the context of c and c' as composed only by their properties. Deep and intensive matching extend the depth of concept context for the contextual affinity evaluation of c and c', by considering also semantic relations with other concepts (deep matching model) as well as property values (intensive matching model), respectively. The comprehensive semantic affinity $SA(c, c')$ is evaluated as the weighted sum of the Linguistic Affinity value and the Contextual Affinity value, that is:

$$SA(c, c') = W_{LA} \cdot LA(c, c') + (1 - W_{LA}) \cdot CA(c, c')$$

where W_{LA} is a weight expressing the relevance to be given for the linguistic affinity in the semantic affinity evaluation process. A detailed description of HMatch and related matching models is provided in [Castano et al., 2006]. HMatch implements two different metrics for linguistic matching and two different metrics for contextual matching. For linguistic matching WordNet interaction is available [Castano et al., 2001] as well as edit-distance string similarity measures [Stoilos et al., 2005; Cohen et al., 2003]. For contextual similarity, we provide default asymmetric measure [Ferrara, 2005] as well as a symmetric measure based on Dice coefficient [Ferrara, 2005]. HMatch is compliant with the Ontology Alignment Format for mapping representation [Euzenat et al., 2006]. HMatch is implemented in Java and released as a plugin of the Protègè ontology editor [9]. Currently it

[9] http://protege.stanford.edu

supports the interaction with reasoning services (in particular with RacerPro) through the DIG support provided by the Protègè OWL API for the sake of acquiring a complete definition of the concepts to be matched. The implementation of new matching techniques for structural and logic matching will be investigated in the project [Ferrara, 2005; Zhang and Shasha, 1997; Resnik, 1995; Borgida et al., 2005] as well as the support and integration of instance matching within the system [Linckels and Meinel, 2005; Janowicz, 2005; d'Amato et al., 2006].

9 INNOVATION

The evolution methodology of BOEMIE presents an innovative approach to the problem of ontology evolution [Maedche et al., 2003; Stojanovic et al., 2002] both per se and with respect to the methods and techniques adopted for the methodology activities. In fact, ontology evolution becomes even more complex in the context of BOEMIE, due to the management of multimedia resources and ontologies and to the multi-modal semantic information extraction to be enabled. In order to deal with these challenges, the BOEMIE evolution methodology proposes a novel combination of several techniques (e.g., advanced clustering, standard and non-standard reasoning, instance and concept matching), with the goal of providing a comprehensive and coherent system capable of supporting the whole evolution process. The methodology proposes a new conceptualization of the problem of evolving multimedia ontologies, by presenting a pattern-driven evolution approach, where different evolution patterns are identified, each one dealing with each specific evolution scenario that can occur, automatically identified on the basis of the results of the instance interpretation activity against the background knowledge. Moreover, we deal with multi-knowledge spaces, both internal and external to the ontology framework of BOEMIE. This requires to develop an evolution methodology capable of managing the coordinated evolution of such interlinked ontology knowledge spaces, by which the modification applied to a part of the ontology will be automatically and consistently propagated to other interlinked knowledge spaces, to keep the multiple knowledge spaces aligned and consistent each other. In particular, the use of external knowledge sources enforces an open system approach. In BOEMIE, we support a knowledge discovery approach typical of open networked systems for enriching the local knowledge of a given node. Thus, we design the evolution methodology to provide a concept discovery functionality which comes into play for acquiring new knowledge by probing external knowledge sources for the same domain, looking for candidate concepts semantically matching the incoming instance. This way, the ontology knowledge space is open towards external knowledge sources, which can provide useful knowledge for evolving the ontology adequately. Finally, the BOEMIE methodology as a whole proposes also a solution to the problem of minimizing the human involvement. To this end, we design a methodology providing a set of learning, matching, and reasoning techniques that offer support in the various evolution activities, to allow the ontology expert to refine proposed working knowledge (e.g., discovered candidate concepts) and/or to validate/choose among proposed alternative choices, thus limiting the human involvement as much as possible to expected interactions with support tools.

In the following, the more specific contributions of the various activities are discussed.

9.1 Population and enrichment

The methodology for the population activity uses an innovative approach for the detection of instances which refer to the same real object or event. The contribution with respect to this problem is summarized as follows:

- Operating on the results of semantic interpretation, the population activity performs instance matching by utilizing the similarities found among MLCIs, HLCIs, and also relations. Thus, instance matching has the potential to compare not only instances of a specific concept, but ontology fragments and provide a figure for their similarity. The measurement of similarities among MLCIs is a challenging research issue, due to the various modalities involved. Modalities such as text are easier to handle (as MLCIs are character strings) than image, audio or video modalities, where MLCIs can be complex structures like image regions or audio segments. In addition, similarity figures are used to "group" the instances that represent the same real object or event.

- Use of instance grouping information to disambiguate cases where there are two or more explanations for the same resource.

The ontology enrichment methodology involves the introduction of new concepts/relations, by detecting when such additions are required. This detection is performed by exploiting the results of a clustering task, which is performed on unexplained ABoxes. These unexplained ABoxes can contain instances of MLCs, relations among these instances but also instances of HLCs and possibly relations among these HLCIs. This leads to the investigation of the following innovative techniques:

- Non-standard clustering service, seeking to cluster ontology fragments which involve concept instances, properties and relations.
- When the clustering phase detects enough information to support the introduction of a new concept/relation, this supporting information is enhanced though information retrieved from external knowledge sources. Thus, the involvement of the ontology expert is reduced, as the expert is required to revise an already formed concept/relation than define this concept/relation from scratch.

9.2 Coordination and matching

The goal of coordination activity is to maintain the alignment of the ontology with the external knowledge sources over time. This requires the investigation of the following novel contributions:

- External information provided by other external knowledge sources is part of the evolution process in the BOEMIE methodology. The idea is that the knowledge provided by other ontologies, web directories, and, in general, knowledge repositories can be used to support the ontology expert by increasing his knowledge of the domain and, at the same time, can be used to enrich the information available about the concepts described in the BOEMIE ontology.
- The alignment of the ontology with the external knowledge sources is maintained over time. This means that we investigate methods and techniques to update existing mappings by exploiting a measure of change of a concept in different versions of the same ontology.

Matching techniques are used in BOEMIE both for supporting population and enrichment and for the goal of ontology alignment. With respect to these goals, the main contributions are summarized as follows:

- Ontology population requires instance matching. The problem of instance matching is investigated in the project on the basis of available matching measures for data. The combination of these measures as well as the problem of exploiting semantic techniques for instance matching is a research contribution in the field of ontology matching in the project, leading to the development of new instance matching techniques.
- Matching techniques are used in BOEMIE in combination with reasoning and clustering techniques, thus leading to the development of more flexible and comprehensive approaches to the problem of ontology matching, both with respect to the structural matching techniques and with respect to semantic matching techniques.

9.3 Reasoning

With respect to the state of the art in the field of knowledge representation and reasoning, the novel contribution of the evolution methodology is summarized as follows:

- Formalization of high-level multimedia interpretation as a logical decision problem and its implementation as a non-standard inference service, namely abduction.
- Extension of the knowledge representation formalism (SWRL) in order to support adequate knowledge representation for abduction tasks.
- Development of new optimization techniques for ABox consistency checking and query answering and its integration into a state-of-the art DL reasoner.
- First approach for using non-standard description logic inference problems for formalizing the learning problem in BOEMIE (LCS, MSC, rewriting, etc.).

- Distinction of structural matching and semantic matching (matching w.r.t. models rather than surface structures).
- Participation in the standardization efforts of the DL Implementation Group (DIG) interface for description logic reasoners in order to provide for standard interfaces for interactions with reasoners. In particular, development of concrete domain and ABox query interfaces, which will become an integral part of the upcoming version of DIG 2.0.

10 EVALUATION

The particular application scenario that will be used for the evaluation of BOEMIE concerns the collection of information about sport events in a number of major European cities. The results of the evolution process, i.e., evolved semantic model, will be also used to train the extraction tool and also to augment the coverage capability of the system to represent the domain information. The evaluation criteria described hereafter for the various activities are expected to flow into a comprehensive evaluation scenario leading to a definition of shared test cases for the ontology evolution toolkit. This will be done in the further stages of the project, in accordance with the requirements and the objectives described in the project deliverable "D5.2 Evaluation Strategy". The quantitative measures used for evaluating ontology evolution are given in Table 10. Here the overall goal is to measure the capability of the evolution to identify and correctly add new concepts, both MLCs and HLCs to the ontology, as well as the precision of the mapping definition with external knowledge sources.

Objective	**Evaluation**	**Target**
Ontology population	Hide part (up to 50%) of the instance set of the ontology and attempt to reconstruct it semi-automatically.	Target: to be able to identify a large part (> 70%) of the hidden instances.
Ontology enrichment	Hide part (up to 20%) of the concepts and relations of the ontology and attempt to reconstruct it semi-automatically.	Target: to be able to identify a large part (> 70%) of the hidden concepts and relations.
Ontology coordination	Hide part (up to 20%) of the mappings between concepts of the ontology and concepts if the external knowledge sources.	Target: to be able to identify a large part (> 70%) of the hidden mappings.
Semantic consistency	Reasoning efficiency.	To be able to check consistency for a Horn-SHIQ Abox from up to 10000 individuals in less than 10 seconds.
Implementation of open architecture	Integrate different tools and examine their effect in the system performance, in controlled experiments.	

Table 10: Evaluation for ontology evolution

10.1 Measurable objectives

In this section, we report the measurable objects defined for the main activities of ontology evolution, namely population, enrichment, and coordination.

10.1.1 Population

Measurable objectives for the population activity are the performance of the instance matching and instances grouping operations. There are two scenarios for evaluating these two operations.

The first evaluation scenario uses as input the manually annotated corpus (this is the ideal input expected from the information extraction toolkit; no ambiguous interpretations – evolution pattern P1 case is only covered) to measure the performance of instance matching and instances grouping tasks. A target of 90% or more may be set for this evaluation scenario.

The second scenario takes as input the system annotated corpus (covers both P1 and P2 evolution patterns). The performance of the information extraction toolkit is known from WP2 evaluation, meaning that it is known which of the interpretations are correct, erroneous, etc. The measuring objective of this scenario is how well the population activity manages to populate the ontology without loss of extracted information and at the same time solving the ambiguous interpretations (evolution pattern P2).

10.1.2 Enrichment

Measurable objectives for the ontology enrichment activity constitute the performance measurements of the ability of the enrichment activity to learn a new MLC, HLC and relation concept. In order to perform such an evaluation, the following scenario is proposed: starting from a "complete" version of the domain ontology, 20% of the concepts are hidden, i.e., removed from the ontology. These hidden concepts must contain concepts that are HLCs, MLCs and relations (33% from each). Using as input the manual annotated corpus (which constitutes the ideal input expected from the information extraction toolkit), documents whose explanations involve hidden concepts are selected. Explanations are removed from these documents and the documents are processed by the evolution toolkit. Each of these documents will trigger the selection of evolution patterns P3 and P4, leading to an enrichment operation. At the end, the performance will be measured according to the percentage of the concepts learned by the evolution toolkit.

10.1.3 Coordination

Measurable objectives for the ontology coordination activity provide an evaluation of the coordination techniques with respect to the quality of the ontology alignment and ontology versioning. For what concerns ontology alignment, several evaluation strategies have been proposed to measure the quality of ontology matching techniques [Euzenat et al., 2006]. These techniques are based on the idea of evaluating the precision and recall of the mappings automatically retrieved by a matching system with respect to a set of expected mappings. More in detail, precision is defined as the ratio of the number of matching concepts automatically found by the matching system considered as relevant to the total number of matching concepts found. Recall is defined as the ratio of the number of relevant matching concepts automatically found by the system to the total number of matching concepts (i.e., mappings) expected. In order to use these metrics in BOEMIE, we need to define a set of mappings that are expected. To this end, we hide more than 20% of the mappings retrieved automatically by the system. Then, we manually define mappings among the concepts involved in the hidden mappings and the external knowledge sources. These manually defined mappings will be the set of expected mappings. The target of the evaluation is to execute the alignment on the concepts involved in order to achieve more than 70% of precision and recall with respect to these mapping set. Ontology versioning will be evaluated by recovering different versions of the ontology starting from the previous ones and the change logs, in controlled experiments.

11 RISK ANALYSIS

In this section, we discuss the main risk factors in the BOEMIE ontology evolution together with possible solutions that can be adopted in order to handle them.

11.1 Risk factors

Risk factors in ontology evolution can be summarized as follows:

1. Input for WP4 is not precise enough. As a result, ontology evolution cannot deliver satisfactory results.

2. The performance of ABox consistency checks is not sufficient to deal with the huge amount of requests.

3. Risk related to instance matching for population. If only structural techniques are employed for instance matching, the results may not be satisfactory for the purpose of population.

4. In instance grouping and validation it is difficult to disambiguate whether a different property value in instances involved in the same group is erroneous or correct.

5. Very few/none results are returned after concept enhancement, during ontology enrichment.

11.2 Risk handling

With respect to the risk factors, we provide the following possible solutions:

1. The problem may arise from: i) performance of semantics extraction ii) the ontologies being too general to represent the extraction results and/or to enable high-level multimedia interpretation iii) high-level multimedia interpretation cannot deliver any satisfactory results. Consequently, WP2 and WP3 activities should be optimized with regard to WP4 requirements.

2. The ontology evolution methodology will continuously modify ABoxes and check their consistency. Therefore the reasoning methodology should be improved such that it can exploit existing structures, in case axioms are added to or retracted from an ABox. This will improve the performance of ABox consistency checks.

3. Usually, ontology matching tools working at the schema level implementing a structural matching approach produce satisfactory results, as proven also by international benchmarks [Euzenat et al., 2006]. In the methodology, structural instance matching extends techniques implemented by the schema-based matchmakers to the problem of instance matching. This should guarantee some level of accuracy in the results, which will be also tested through experiments. However, working at the instance level, also semantics-based similarity techniques could be investigated to improve the quality of results by taking into account not only structure but also semantics.

4. Property value compatibility checks are limited to the properties that cannot have values which change dramatically, such as, in the athletic domain, the athlete surname.

5. The problem may arise from: i) the selected external sources not being rich enough for the new concept under construction; in this case new external sources can be identified and probed to extend the search; ii) the matching service not being suitably configured; different configurations can be experimented with different matching models to analyze produced results.

12 CONCLUDING REMARKS

In this document we have defined a methodology for multimedia ontology evolution. We have specified how information from the multimedia semantic model and from the semantics extraction process can be used to achieve the coordinated and consistent evolution of the ontology, both against the internal multi-knowledge organization of the semantic model and against other external knowledge sources used to discover new, relevant knowledge. Additionally, we have introduced four evolution patterns to consider each particular evolution situation that can occur as either a population or enrichment activity over the ontology.

For the methodology we have defined:

- required input and expected output to other modules;
- overview of the internal organization into activities, tasks and operations featuring each evolution pattern;
- evaluation framework.

Moreover, for evolution patterns, we have defined:

- specific support for ontology evolution (i.e., population, enrichment, coordination)
- description concerning MLC detection and low level descriptors for those situations where no information about MLCs is acquired from the information extraction process
- comprehensive example of pattern application in the Athletics domain, considering jumping events in particular.

The outcome of the proposed methodology is an open architecture, which will communicate with the information extraction modules in WP2, accessing existing semantic information model and providing back newly evolved ontology knowledge. The architecture also specifies the interface for the population, enrichment, and coordination tools as well as for the reasoning and matching services defined in WP4.

References

F. Baader, C. Lutz, and B. Suntisrivaraporn. CEL—a polynomial-time reasoner for life science ontologies. In U. Furbach and N. Shankar, editors, *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR'06)*, volume 4130 of *Lecture Notes in Artificial Intelligence*, pages 287–291. Springer-Verlag, 2006.

F. Baader, B. Sertkaya, and A.-Y. Turhan. Computing the least common subsumer w.r.t. a background terminology. In José Júlio Alferes and João Alexandre Leite, editors, *Proceedings of the 9th European Conference on Logics in Artificial Intelligence (JELIA 2004)*, volume 3229 of *Lecture Notes in Computer Science*, pages 400–412, Lisbon, Portugal, 2004. Springer-Verlag.

Franz Baader, Sebastian Brandt, and Ralf Küsters. Matching under side conditions in description logics. In *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001)*, pages 213–218, 2001.

Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005)*, pages 364–369, 2005.

Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications.* Cambridge University Press, 2003.

Franz Baader, Enrico Franconi, Bernhard Hollunder, Bernhard Nebel, and Hans-Jürgen Profitlich. An empirical analysis of optimization techniques for terminological representation systems or: Making KRIS get a move on. *Applied Artificial Intelligence. Special Issue on Knowledge Base Management*, 4:109–132, 1994.

Franz Baader and Philipp Hanschke. A Schema for Integrating Concrete Domains into Concept Languages. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)*, pages 452–457, 1991.

Franz Baader and Ralf Küsters. Computing the least common subsumer and the most specific concept in the presence of cyclic $\mathcal{ALN}$-concept descriptions. In *Proc. of the 22nd German Annual Conf. on Artificial Intelligence (KI'98)*, volume 1504 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1998.

Franz Baader and Ralf Küsters. Matching in description logics with existential restrictions. In *Proc. of the 1999 Description Logic Workshop (DL'99).* CEUR Electronic Workshop Proceedings, `http://ceur-ws.org/Vol-22/`, 1999a.

Franz Baader and Ralf Küsters. Matching in description logics with existential restrictions — Revisited. Technical Report LTCS-Report 99-13, LuFg Theoretical Computer Science, RWTH Aachen, Germany, 1999b.

Franz Baader and Ralf Küsters. Matching in description logics with existential restrictions. In *Proc. of the 7th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2000)*, pages 261–272, 2000.

Franz Baader and Ralf Küsters. Unification in a description logic with transitive closure of roles. In Robert Nieuwenhuis and Andrei Voronkov, editors, *Proc. of the 8th Int. Conf. on Logic for Programming, Artificial Intelligence and Reasoning (LPAR 2001)*, volume 2250 of *Lecture Notes in Computer Science*, pages 217–232. Springer, 2001.

Franz Baader, Ralf Küsters, Alex Borgida, and Deborah L. McGuinness. Matching in description logics. *J. of Logic and Computation*, 9(3):411–447, 1999a.

Franz Baader, Ralf Küsters, and Ralf Molitor. Computing least common subsumers in description logics with existential restrictions. In *Proc. of the 16th Int. Joint Conf. on Artificial Intelligence (IJCAI'99)*, pages 96–101, 1999b.

Franz Baader, Ralf Küsters, and Ralf Molitor. Rewriting concepts using terminologies. In *Proc. of the 7th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2000)*, pages 297–308, 2000.

Alex Borgida, Thomas J. Walsh, and Haym Hirsh. Towards measuring similarity in description logics. In *Proceedings of the 2005 International Workshop on Description Logics (DL2005), Edinburgh, Scotland*, 2005.

Alexander Borgida and Ralf Küsters. What's not in a name? Initial explorations of a structural approach to integrating large concept knowledge-bases. Technical Report DCS-TR-391, Rutgers University, 1999.

S. Brandt, R. Küsters, and A.-Y. Turhan. Approximation and difference in description logics. In D. Fensel, F. Giunchiglia, D. McGuiness, and M.-A. Williams, editors, *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR2002)*, pages 203–214, San Francisco, CA, 2002. Morgan Kaufman.

Sebastian Brandt, Ralf Küsters, and Anni-Yasmin Turhan. Approximation in description logics. LTCS-Report 01-06, LuFG Theoretical Computer Science, RWTH Aachen, Germany, 2001. Available at `http://www-lti.informatik.rwth-aachen.de/Forschung/Reports.html`.

Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. DL-Lite: Tractable description logics for ontologies. In *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005)*, pages 602–607, 2005.

S. Castano, V. De Antonellis, M. G. Fugini, and B. Pernici. Conceptual schema analysis: techniques and applications. *ACM Trans. Database Syst.*, 23(3):286–333, 1998. ISSN 0362-5915.

S. Castano, V. De Antonellis, and S. De Capitani Di Vimercati. Global viewing of heterogeneous data sources. *IEEE Transactions on Knowledge and Data Engineering*, 13(2):277–297, March/April 2001. URL `http://portal.acm.org/citation.cfm?id=628127`.

S. Castano, A. Ferrara, and S. Montanelli. Matching ontologies in open networked systems: Techniques and applications. *Journal on Data Semantics (JoDS)*, V, 2006.

William W. Cohen, Pradeep Ravikumar, and Stephen E. Fienberg. A comparison of string distance metrics for name-matching tasks. In Subbarao Kambhampati and Craig A. Knoblock, editors, *IIWeb*, pages 73–78, 2003.

C. d'Amato, N. Fanizzi, and F. Esposito. A dissimilarity measure for alc concept descriptions. In *SAC 2006 RDRC Track, Dijon, France*, 2006.

Dawei Ding, Qing Li, and Jun Yang. Multimedia data integration and navigation through mediaview: Implementation, evolution and utilization. In Yoon-Joon Lee, Jianzhong Li, Kyu-Young Whang, and Doheon Lee, editors, *DASFAA*, volume 2973 of *Lecture Notes in Computer Science*, pages 721–737. Springer, 2004. ISBN 3-540-21047-4.

A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *Proc. of the 11th Int. World Wide Web Conference (WWW 2002)*, pages 662–673, Honolulu, Hawaii, USA, May 2002.

M. Ehrig. Ontology mapping. In *Information Interpretation and Integration Conference (I3CON)*, August 2004. URL `http://www.isd.mel.nist.gov/research_areas/research_engineering/Performance_Metrics/PerMIS_2004/Proceedings/`.

M. Ehrig and Y. Sure. Ontology mapping - an integrated approach. In Christoph Bussler, John Davis, Dieter Fensel, and Rudi Studer, editors, *Proceedings of the First European Semantic Web Symposium*, volume 3053 of *Lecture Notes in Computer Science*, pages 76–91, Heraklion, Greece, May 2004. Springer Verlag.

C. Elsenbroich, O. Kutz, and U. Sattler. A case for abductive reasoning over ontologies. In *Proc. OWL: Experiences and Directions, Athens, Georgia, USA, November 10-11*, 2006.

Jerome Euzenat, Malgorzata Mochol, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. First results of the ontology alignment evaluation initiative 2006. In *Proc. of International Workshop on Ontology Matching, collocated with the 5th International Semantic Web Conference ISWC-2006*, Athens, Georgia, USA, November 2006.

Waleed E. Farag and Hussein M. Abdel-Wahab. A human-based technique for measuring video data similarity. In *ISCC*, pages 769–774. IEEE Computer Society, 2003. ISBN 0-7695-1961-X.

Alfio Ferrara. *Matching of Independent Ontologies in Open Networked Systems: Methods and Techniques*. PhD thesis, Università degli Studi di Milano, 2005.

J. W. Freeman. *Improvements to Propositional Satisfiability Search Algorithms*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania, 1995.

J. Galinski, A. Kaya, and R. Möller. Development of a server to support the formal semantic web query language owl-ql. In I. Horrocks, U. Sattler, and F. Wolter, editors, *Proc. International Workshop on Description Logics*, 2005.

F. Giunchiglia and P. Shvaiko. Semantic matching. *Knowledge engineering review*, 18(3), 2003.

F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-match: an algorithm and an implementation of semantic matching. In Bussler C., Davies J., Fensel D., and Studer R., editors, *The semantic web: research and applications*, volume 3053 of *LNCS*, Heraklion, 10-12 May 2004.

B. Glimm, I. Horrocks, C. Lutz, and U. Sattler. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence IJCAI-07*. AAAI Press, 2007.

V. Haarslev and R. Möller. Consistency testing: The RACE experience. In R. Dyckhoff, editor, *Proceedings, Automated Reasoning with Analytic Tableaux and Related Methods*, number 1847 in Lecture Notes in Artificial Intelligence, pages 57–61. Springer, April 2000a.

V. Haarslev and R. Möller. Signature calculus: Optimizing reasoning with number restrictions. Technical report, University of Hamburg, Computer Science Dept., June 2001a.

V. Haarslev and R. Möller. Racer: A core inference engine for the semantic web. In *Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools (EON2003), located at the 2nd International Semantic Web Conference ISWC 2003, Sanibel Island, Florida, USA, October 20*, pages 27–36, 2003a.

V. Haarslev and R. Möller. Racer: An owl reasoning agent for the semantic web. In *Proceedings of the International Workshop on Applications, Products and Services of Web-based Support Systems, in conjunction with the 2003 IEEE/WIC International Conference on Web Intelligence,Halifax, Canada, October 13*, pages 91–95, 2003b.

Volker Haarslev and Ralf Möller. Expressive ABox reasoning with number restrictions, role hierarchies, and transitively closed roles. In *Proc. of the 7th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2000)*, pages 273–284, 2000b.

Volker Haarslev and Ralf Möller. High performance reasoning with very large knowledge bases: A practical case study. In *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001)*, pages 161–168, 2001b.

Volker Haarslev and Ralf Möller. RACER System Description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Artificial Intelligence*, pages 701–705. Springer, 2001c. Available at `http://www.racer-systems.com`.

Volker Haarslev and Ralf Möller. Optimization Techniques for Retrieving Resources Described in OWL/RDF Documents: First Results. In *Proc. of the 9th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2004)*, 2004.

Volker Haarslev, Ralf Möller, and Anni-Yasmin Turhan. Exploiting pseudo models for TBox and ABox reasoning in expressive description logics. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Artificial Intelligence*. Springer, 2001a.

Volker Haarslev, Ralf Möller, and Michael Wessel. The description logic $\mathcal{ALCNH}_{R+}$ extended with concrete domains: A practically motivated approach. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2001)*, pages 29–44, 2001b.

Volker Haarslev, Ralf Möller, and Michael Wessel. Querying the semantic web with racer + nrql. In *Proc. of the 2004 Description Logic Workshop (DL 2004)*, 2004.

Ian Horrocks. *Optimising Tableaux Decision Procedures for Description Logics*. PhD thesis, University of Manchester, 1997.

Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The even more irresistible $\mathcal{SROIQ}$. In *Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 57–67. AAAI Press, 2006. ISBN 978-1-57735-271-6. URL `download/2006/HoKS06a.pdf`.

Ian Horrocks and Peter F. Patel-Schneider. Optimising propositional modal satisfiability for description logic subsumption. In *Proc. of the 4th Int. Conf. on Artificial Intelligence and Symbolic Computation (AISC'98)*, 1998.

Ian Horrocks, Ulrike Sattler, Sergio Tessaris, and Stefan Tobies. How to decide query containment under constraints using a description logic. In *Proc. of the 7th Int. Conf. on Logic for Programming and Automated Reasoning (LPAR 2000)*, volume 1955 of *Lecture Notes in Computer Science*, pages 326–343. Springer, 2000a.

Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Reasoning with individuals for the description logic $\mathcal{SHIQ}$. In David McAllester, editor, *Proc. of the 17th Int. Conf. on Automated Deduction (CADE 2000)*, volume 1831 of *Lecture Notes in Computer Science*, pages 482–496. Springer, 2000b.

Ian Horrocks and Stephan Tobies. Reasoning with axioms: Theory and practice. In *Proc. of the 7th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2000)*, pages 285–296, 2000.

Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Reducing $\mathcal{SHIQ}$-Description Logic to Disjunctive Datalog Programs. In *Proc. of the 9th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2004)*, pages 152–162, 2004.

Interop. State of the art and state of the practice including initial possible research orientations. Deliverable d8.1, NoE INTEROP - IST Project n. 508011 - 6th EU Framework Programme, 2004.

K. Janowicz. Sim-dl: Towards a semantic similarity measurement theory for the description logic alcnr in geographic information retrieval. In P. Herrero et al. R. Meersman, Z. Tari, editor, *SeBGIS 2006, OTM Workshops 2006*, number 4278 in LNCS, pages 1681–1692, 2005.

Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review Journal*, 18(1), 2003.

Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and Bernardo Cuenca-Grau. Repairing unsatisfiable concepts in owl ontologies. In *Proc. The European Semantic Web Conference (ESWC)*, 2006a.

Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and James Hendler. Debugging unsatisfiable classes in owl ontologies. *Journal of Web Semantics*, 3(4), 2006b.

Alissa Kaplunova, Atila Kaya, and Ralf Möller. First experiences with load balancing and caching for semantic web applications. Technical report, Institute for Software Systems (STS), Hamburg University of Technology, Germany, 2006. See http://www.sts.tu-harburg.de/papers/techreports.html.

S. Linckels and Chr. Meinel. A simple application of description logics for a semantic search engine. In *Proc. of the IADIS International Conference of Applied Computing (IADIS AC2005)*, pages 306–311, 2005.

C. Lutz, F. Wolter, and M. Zakharyaschev. A tableau algorithm for reasoning about concepts and similarity. In *Proceedings of the Twelfth International Conference on Automated Reasoning with Analytic Tableaux and Related Methods TABLEAUX 2003*, LNAI, Rome,Italy, 2003. Springer.

A. Maedche, B. Motik, L. Stojanovic, R. Studer, and R. Volz. An Infrastructure for Searching, Reusing and Evolving Distributed Ontologies. In *Proc. of the 12th Int. World Wide Web Conference (WWW 2003)*, pages 439–448, Hungary, 2003.

Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Improving automatically created mappings using logical reasoning. In *Proc. of International Workshop on Ontology Matching, collocated with the 5th International Semantic Web Conference ISWC-2006*, Athens, Georgia, USA, November 2006.

G. A. Miller. WordNet: A lexical database for english. *Communications of the ACM (CACM)*, 38(11):39–41, 1995.

Malgorzata Mochol, Anja Jentzsch, and Jerome Euzenat. Applying an analytic method for matching approach selection. In *Proc. of International Workshop on Ontology Matching, collocated with the 5th International Semantic Web Conference ISWC-2006*, Athens, Georgia, USA, November 2006.

Ralf Möller, Volker Haarslev, and Michael Wessel. On the scalability of description logic instance retrieval. In Chr. Freksa and M. Kohlhase, editors, *29. Deutsche Jahrestagung für Künstliche Intelligenz*, Lecture Notes in Artificial Intelligence. Springer Verlag, 2006.

Boris Motik, Ulrike Sattler, and Rudi Studer. Query answering for OWL-DL with rules. *J. of Web Semantics*, 3(1):41–60, 2005.

S. Muggleton. Inductive Logic Programming. In *The MIT Encyclopedia of the Cognitive Sciences (MITECS)*. MIT Press, 1999. URL `citeseer.ist.psu.edu/muggleton92inductive.html`.

Bernhard Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43: 235–249, 1990.

N. Noy. Semantic integration: a survey of ontology-based approaches. *SIGMOD Record Special Issue on Semantic Integration*, December 2004.

Natalya F. Noy and Michel Klein. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems*, 6(4):428–440, July 2004. URL `http://www.cs.vu.nl/~mcaklein/papers/KAIS03.pdf`.

Bao Hua Qiang, Kai Gui Wu, Xiao Feng Lao, and Zhong Fu Wu. Similarity determination based on data types in heterogeneous databases using neural networks. In *IEEE International Conference on Neural Networks and Signal Processing*, Nanjing, China, December 2003.

P. Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence*, 11(11), 1995.

Murray Shanahan. Perception as Abduction: Turning Sensor Data into Meaningful Representation. *Cognitive Science*, 29:103–134, 2005.

P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. *Journal on Data Semantics (JoDS)*, 1, 2005.

Evren Sirin and Bijan Parsia. Pellet System Description. In *Proc. of the 2006 Description Logic Workshop (DL 2006)*. CEUR Electronic Workshop Proceedings, `http://ceur-ws.org/`, 2006.

R.M. Stallman and G.J. Sussman. Forward reasoning and dependency-directed backtracking in a system for computer-aided circuit analysis. *Artificial Intelligence*, 9(2):135–196, 1977.

G. Stoilos, G. Stamou, and S. Kollias. A string metric for ontology alignment. In *Proc. of the 4th International Semantic Web Conference (ISWC 2005), Galway, 2005*, 2005. URL `http://www.image.ece.ntua.gr/publications.php`.

L. Stojanovic, A. Maedche, B. Motik, and N. Stojanovic. User-driven Ontology Evolution Management. In *Proc. of the 13th Int. Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web (EKAW 2002)*, pages 285–300, Siguenza, Spain, 2002.

Dmitry Tsarkov and Ian Horrocks. FaCT++ Description Logic Reasoner: System Description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006)*, 2006. To appear.

Anni-Yasmin Turhan and Christian Kissig. Sonic—Non-standard inferences go OilEd. In *Proc. of the 2nd Int. Joint Conf. on Automated Reasoning (IJCAR 2004)*, volume 3097 of *Lecture Notes in Computer Science*, pages 321–325. Springer, 2004.

M. Wessel and R. Möller. A High Performance Semantic Web Query Answering Engine. In I. Horrocks, U. Sattler, and F. Wolter, editors, *Proc. International Workshop on Description Logics*, 2005.

M. Wessel and R. Möller. A flexible dl-based architecture for deductive information systems. In G. Sutcliffe, R. Schmidt, and S. Schulz, editors, *Proc. IJCAR-06 Workshop on Empirically Successful Computerized Reasoning (ESCoR)*, pages 92–111, 2006.

K. Zhang and D. Shasha. *Pattern matching in strings, trees, and arrays*, chapter Approximate tree pattern matching, pages 341–371. Oxford University Press, 1997.