Implementing OBDA for an end-user query answering service on an educational ontology

MASTER OF SCIENCE THESIS

Date: 18/10/2016

Supervisor

PhD. Maria Ribera Sancho

Service Engineering and Information Systems

This thesis is presented by Rodrigo Cueva Rueda

In fulfilment of the requirements for the degree of

Master in Innovation and Research in Informatics

with specialization in

Service Engineering

ACKNOWLEDGEMENTS

I would like to express gratitude to my supervisor PhD. Maria Ribera Sancho Samso for the valuable advice and guidance provided for the development of this study. Her continuous instruction helped me not just in research time but also in the Master's Course progress.

My very profound thankfulness is for my lovely wife Isabel, to whom this effort is entirely dedicated. Her undoubtedly love has fed my courage in some distressing times and in others had inspired me to pursue my dreams, granting me unique moments in this incredible quest of personal and professional growth. To my parents Consuelo and Guillermo and my sister Daniela, who with their unconditional kindness and family love has been the most important pillar of strength.

To my new friends Shiva, Daria, Kate and Eric with whom I shared alike experiences and funny moments, and they had become like our family in this country. Thank you for all the support.

ABSTRACT

In the age where productivity of society is no longer defined by the amount of information generated, but from the quality and assertiveness that a set of data may potentially hold, the right questions to do depends on the semantic awareness capability that an information system could evolve into. To address this challenge, in the last decade, exhaustive research has been done in the Ontology Based Data Access (OBDA) paradigm.

A conspectus of the most promising technologies with data integration capabilities and the foundations where they rely are documented in this memory as a point of reference for choosing tools that supports the incorporation of a conceptual model under a OBDA method. The present study provides a practical approach for implementing an ontology based data access service, to educational context users of a Learning Analytics initiative, by means of allowing them to formulate intuitive enquiries with a familiar domain terminology on top of a Learning Management System. The ontology used was completely transformed to semantic linked data standards and some data mappings for testing were included. Semantic Linked Data technologies exposed in this document may exert modernization to environments in which object oriented and relational paradigms may propagate heterogeneous and contradictory requirements. Finally, to validate the implementation, a set of queries were constructed emulating the most relevant dynamics of the model regarding the dataset nature.

CONTENTS

LIST OF FIGURES

LIST OF LISTINGS

LIST OF TABLES

CHAPTER 1

1 Introduction

Nowadays, one of the most relevant topic for academic research is how to access assertive information for decision making from knowledge representation. It demands that engineers have an acceptable familiarity with some cutting-edge technologies.

This set of technologies comes from the Semantic Modelling Paradigm. This paradigm has been dealing with a common issue when trying to represent data in a flexible and generic fashion. One of the approaches to deal with this issue implicates semi-structured data modelling [1].

On one hand, the technologies that follow the Extracting, Transforming and Loading method (from now on, as ETL) over data sources has been the traditional way to support this type of data modelling, providing an exploration layer for data exploitation. The problem comes when the natural dynamism of organizations make that information requirements differ, affecting the initial design of a solution. In this scenario, an entire redesign effort means that engineers expend more time and the organization loses money.

On the other hand, to deal with massive data is an opportunity to find a solution trough the development of semantic models. Those models should be based on user domain contexts with a tolerate flexibility to manage diverse information.

In that sense, the search and test for an appropriate methodology is relevant in order to settle basic guidelines when developing semantic aware solutions for intelligent systems with both perspectives: ETL and “Big Data-like”[1].

In the educational context, any domain modelling initiative that can hold semantic awareness over an operational data source in Spain, hasn’t been still developed. An eventual effort could add linked data benefits to the current information management systems used to gather student’s behaviour in e-learning platforms. This study proposes a model implementation based on an educational ontology for end-user’s data access.

The whole document is organized as follows:

Chapter 1 shows the context related to this study and a problem statement. It also introduces a scientific background about all semantic linked data technologies for applying OBDA modelling and their relation over two challenges of the big data.

Chapter 2 focus on the general motivation of the research and a detailed description of the main objectives with its respective rationale description.

Chapter 3 explains in detail, the criteria of all technology selected and being worked on for the model implementation.

Chapter 4 summarizes a State of the Art over OBDA modelling tools.

Chapter 5 describes technical details for the development of the ontology translation and mappings definition.

[1] Tutorial on Big Data - http://videolectures.net/eswc2012_grobelnik_big_data/

Chapter 6 covers the evaluation process and results to validate the model.

Chapter 7 exposes a discussion including the contributions of this work, conclusions and future work for this implementation.

1.1 Domain Context and Problem

An ontology was developed in the Unified Modelling Language (commonly known as UML[2]) by [2], for an information system project [3] of secondary high schools in Cataluña called "Learning Analytics for Secondary" (also known as LA4S). This conceptual model relies on the information from the Learning Management System (from now on, LMS) Agora.

Some traces of Educational Data generated from the mentioned LMS allow a branch like Learning Analytics, to determine the best way to represent the behaviour of students in academic aspects, related to virtual environments.

The main purpose of the project developed by inLab FIB in 2013 in collaboration with Departament d' Enseyament of the Generalitat de Catalunya, is to create indicators (learning analytics) based on operational data sources for investigative purposes in high schools and colleges in the province of Catalonia, Spain.

Part of collaboration regarding data management is in hands of UPCnet, whom based on a rigorous procedure of data protection and privacy of information; allow clearance access to a dataset dump of Agora's database, which is the official LMS customized for high schools, locally.

The project has an ETL layered component integrated. Some deficiencies of this architectural design are: bottlenecks to valuable information access, natural scalable limitations of the installed infrastructure and no sufficient end-user power for data exploration.

The general scope of the present study starts from the conceptualization effort of the previous ontology definition for LA4S and the lack of domain end-users access to incomplete but relevant information. The problem raises when, in order to retrieve relevant portions of data for building these learning analytics indicators in LA4S project; it is required at the same time, a deep knowledge of the educational context domain and also an acceptable awareness of the LMS data repositories.

This causes a sharp distinction between two human roles: the group that is expert in the domain and does data analysis and interpretation, and the group that is expert in the extraction of the data. Naturally, a bottleneck comes up when one of these groups try to enquire a specific question over the information. This issued scenario may repeat in many other contexts and in different sort of organizations and can be covered by the construction of a reliable semantic end to end connection between users and databases.

Thus, the transformation of this ontology to Linked Data standards is of interest to validate and refine the information that the present system has been using for developing indicators of student's motivation. By standardizing a conceptual model, design misfeasance is avoided and also, provides portability and versatility to the prototype

[2] As described conventionally on - http://www.omg.org/spec/UML/

product resulting from this project. The prototype is based in theory-backed practical techniques.

1.2 Scientific Background

For a better understanding of the technology that is being applied and the goals of the current study a deductive description of the origins, challenges, definitions, models and frameworks will be presented in this section.

1.2.1 Big Data Challenges

In order to tackle and get the entire picture of notions for applying semantic aware solutions to informational systems is fundamental to understand the nature of semantic modelling basic concepts, and so forth the called “Big Data” challenges that can be covered by applying this technology.

Whereas under the public general view, the notions about this trendy paradigm are related incorrectly to the size and the capacity of an organization to produce massive amounts of information. The actual reality under the Science Field perspective, denotes the main notion of the 4th Paradigm of Science [4], which states that the very action of producing massive information in any field, is developing a feedback system that recurrently makes that knowledge expands towards new outcomes.

This singular reason, implies that an interdisciplinary work is needed to overcome the traditional 4 traits or V’s (former 3 in 2001 by Douglas Laney [5]) related to the “Big Data” phenomenon.

These traits are general features for describing this paradigm, with their own set of characteristics:

- **Volume:** Dimension of data science referred to aspects of massive data generation and collection on informational systems. From this trait is where most of processing and storage issues[6] derive. Supercomputing and Database disciplines are the most likely interested audience in this feature[7].

- **Velocity:** Dimension of data science related to massive data creation in real time and fast generation way that handle multiple sources in a timeliness period. Issues most prominent are related to data transport and management and the audience interested are researchers from the Internet of Things discipline[7].

- **Variety:** Dimension of data science that referred to the heterogeneity level of data sources. A broad gamma includes structured and unstructured types. Issues commonly raised up in this feature are storage and management kind. Data and conceptual modelling are the direct disciplines that handle this challenge [8].

- **Veracity** (or Value): Dimension of data science that defines the benefits that atomic data brings after being correct treated, integrated and queried into for any informational system. Common issues are related to management problems, derived from the information genus sources. The audience in this case, is all sub-groups or disciplines that comes from human and social sciences (healthcare, education, etc.) [7] which their domain context is based on the accurate definition of their conceptualizations and data processing.

These trait concepts, should not be considered as formal descriptions of the literature; in fact, there are several studies in academic articles with a huge set of perspectives or attempts to formally describe a correct taxonomy. For instances, the use of some other v-words in [9], extending new feature definitions like in [10], [11], [12], the use of seven axes for comparing features between, what should be considered as “small” data and “Big Data” in qualitative differences in the study of [13],[14] and a five-fold taxonomy categorization presented in [15] to measure them.

Kitchin et. al. [16] explored 26 datasets for describing common ontological characteristics from several dimensions of data concluding that the first two v’s (volume and velocity) actually misleads the definition of “Big Data” and suggested that any dataset, depending on how it evolves could potentially become a candidate that dives between any trait analysed. There has been an extensive discussion to formally define the "Big Data" context and more information can be found in [17].

The definition of this 4 features, leads to an understanding of the challenges from a general perspective and allow to entangle them with the purposes of this study. The main purpose is not to create an implementation over a massive dataset but to provide the foundations and general knowledge that must be considered to initialize a potential solution

Regarding to the **variety** trait of information, semantic technologies are the ad-hoc path for developing most of the aware solutions. They may contain a great spectrum of properties such as representational formats of data, correctness in conceptualizations and dependencies as it is described in [18]; or even become part from an already defined system as a complimentary toolset that enforces the desired flexibility in a modifying environment (basic notions are extended in Sections 1.2.2.1 and 1.2.2.2).

It is worth to notice that the typical fields that involve the core of their study in semantical contexts are databases, artificial intelligence and cognitive science since many years. Surprisingly, there has not been an actively involvement of the use of semantics over pure educational context datasets.

For the last trait perspective **(veracity),** the interest of correctness in concept modelling is critical. This comes from the need to manage consistent information and knowledge generation.

1.2.2 Semantic Linked Data

Part of the linked data definitions are derived from the web semantic technology viewpoint. Initially envisioned by the inventor of World Wide Web, Sir Tim Berners-Lee on a design note[3], Semantic Linked Data is commonly described as the capacity of data on web to be expressed under developing standard languages. This note also refers that data is rich and embedded in self-describing information with the particularity of relational power with additional data. The 4th rules that Berners-Lee recommends in the note are textually included:

1. Use URIs as names for things.

2. Use HTTP URIs so that people can look up those names.

[3] https://www.w3.org/DesignIssues/LinkedData.html

3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).

4. Include links to other URIs. So that they can discover more things.

In contrast of the four rules which promote open free access for general public, not all data can be available for everybody. That's why linked data is referred specifically to the level of richness expressivity and relational capacity of a model (i.e., relationships are treated as first class citizens [19]) rather than the capacity to be promulgated in the web. Regardless the use on URIs for things or resources, Berners-Lee provide a five star ranking deployment scheme where the fourth qualifying ranking position describes the advantages and disadvantages for linked initiatives[4].

A semantic solution should be designed considering all affordable technology pillars described by the Semantic Linked Data. The ideal technology stack implementation for a semantic-aware solution is illustrated in Figure 1:

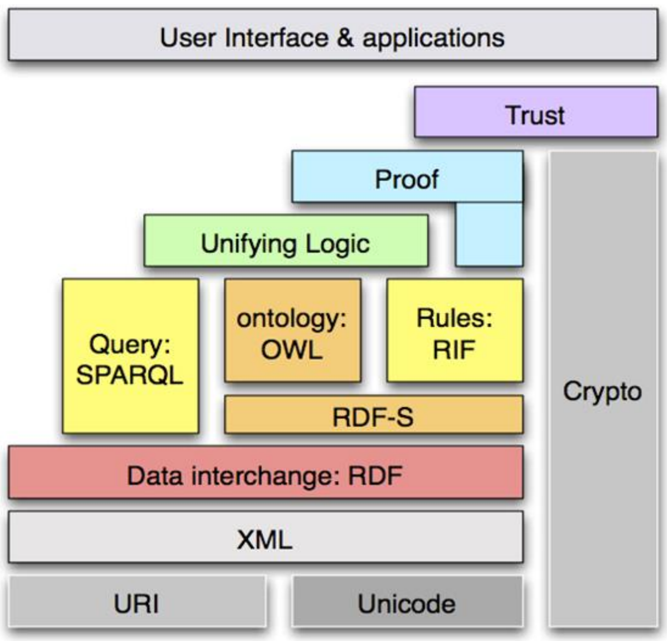


Figure 1. Semantic Web Standards Stack[5]

Notice that the ontology model component is represented in the central part. The ground grey portion is the physical technological base where objects/values are identified (e.g., URI and/or Unicode) and serialized (trough XML) and the last box at the top, denotes the user interface intended after all the model is implemented. In most of the cases, system design in organizations would not allow the execution of the entire technology stack presented, some reasons are originated by the detailed limitations discussed in [20].

[4] More info in http://5stardata.info/en/
[5] Adapted from original work of Marek Obitko in 2007. https://www.w3.org/DesignIssues/diagrams/sweb-stack/2006a.png

1.2.2.1 Technology behind Linked Data

The next paragraphs will describe in general the most prominent technologies used for linked data.

According to the third rule the global standard for formal representation of data is the use of **Resource Description Framework** or **RDF**[6], a standard model for data interchange. It also can be considered as a formal ontology language that includes literals and properties, this technology is based on XML serialization and its conventions (i.e., standards of W3C[7]). Its principal function is to store facts with basic semantical sense, under a triple statement logic of binary relations from resources composed of subject, predicate and object information. The great benefit of RDF standards is the support for schema models construction also called ontologies (see next Section 1.2.2.2), allowing to materialize the principle of linking information. RDF technology may also be used as triplestore[8] structure database that is capable of semantic graphs generation which is a feasible way to transfer first level of semantic information. Some additional strong points and shortcomings of this technology are specified on the web[9]. **RDFS** is an extension of the same language and provides a higher level of semantics for data constraint modelling and taxonomy creation between classes and properties for individuals [21].

OWL or Web Ontology Language is the computational logic-based language defined as another W3C specification, which is designed to show the data schema and represents rich and complex knowledge about hierarchies and the relations between individuals (i.e., instances) or sets of individuals (i.e., classes) [22]. This language deals with logical representation, which are sets of axioms of property assertions between classes, providing a higher level of semantics for system information inference (i.e., reasoning [21]) (see Section 1.2.2.2 for further details). It is considered complementary to RDF standards, because uses RDF as its underlying representation. In general, the language provides the most complete and formalized way to model any data schema/ontology structure in a given domain by separating from the data itself.

OWL essence comes from object properties, which are commonly defined as the relations between individuals of two OWL classes; and datatype properties which are the relations of individuals of OWL classes to literal values[10].

OWL main characteristic is to provide the ability of processing the content of a specification in a machine readable way. This language is highly expressive as may it provide additional vocabulary with formal semantics (i.e. Description Logics - DL), further details over fundamental logics are provided in[23]. Additionally, there are language extension definitions that are used to complement the terminological pair from RDF/OWL for developing initiatives. The most popular are SKOS[11] (Simple Knowledge Organization System) and FOAF[12] (Friend of a Friend).

OWL version 1.0 is divided in three sub languages: Lite (further details can be found on Section 3.1), DL and Full OWL. But after several unifying conventions and new feature developments (i.e., syntactic facilities, user datatype definition, class attributes and

[6] https://www.w3.org/RDF/
[7] https://www.w3.org/XML/
[8] https://ontotext.com/knowledgehub/fundamentals/what-is-rdf-triplestore/
[9] https://www.w3.org/TR/2004/REC-rdf-concepts-20040210/
[10] Conceptual adaptations are from - http://www.linkeddatatools.com/introducing-rdfs-owl
[11] http://www.w3.org/TR/skos-reference/
[12] http://xmlns.com/foaf/spec/

primary keys, etc.), in the release of version OWL 2, the sub type Lite was reformed. The result are three new profiles: OWL RL, OWL QL y OWL EL[13].

OWL 2 EL: Is a profile based on the lightweight description logics EL++ (i.e., Existential quantification), that is suitable for large number of covered properties/classes applications; its level of computational complexity reasoning is polynomial regarding to the terminological (TBOX) component as explained in [24].

OWL 2 QL: Is a profile based on description logics DL-Lite$_R$. The most important characteristic is that can handle most of the ER – Entity/Relational and UML expressiveness for reasoning. The level of computational reasoning complexity is reducible to the size of a database. The main function of this profile is to provide a feasible modelling logic for applications with very large data and when query answering is the most important requirement[14].

OWL 2 RL: Is purely based in normal logic, this profile is based on rule languages, and is used to scalable reasoning applications by not limiting expressivity, its level of reasoning can be polynomial regarding the size of the ontology where it is built.

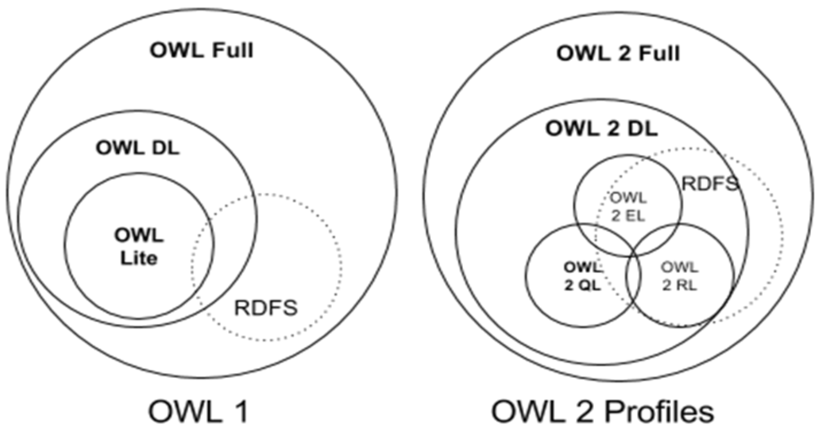


Figure 2. Level of Expressivity and Complexity in OWL. [15]

Figure 2, shows a comparison of the level of expressivity and computational complexity between the two version of the ontological OWL versions for knowledge representation including their subtypes, profiles and RDFS integration:

SPARQL[16] is by default the semantic query language for the Semantic Web, which is specifically designed to be the query protocol and language across various systems and databases to retrieve and process data stored in RDF format, also OWL (limited). Its underneath logic is based on pattern matching for RDF graphs used as query patterns.

[13] More details on the language can be found on - https://www.w3.org/TR/owl2-overview/

[14] Explained details can be found on - https://www.w3.org/TR/owl2-profiles/#OWL_2_QL

[15] OWL Profile General features support and syntax in - https://www.w3.org/TR/owl2-overview/

[16] Definition adapted from - https://www.w3.org/TR/sparql11-query/

1.2.2.2 Ontology models behind Linked Data

In order to consider the best approach to design a solution to effectively add semantic capability, it is relevant to review modelling fundamentals with ontology concepts and methods. The reason of using ontologies as building blocks for informational systems, lies on the capacity to centralize knowledge from any field to a common logic framework. This common framework is known as description logics [25].

Description Logics, is a modelling paradigm based on **First Order Logic (FOL)**. It provides capabilities of knowledge representation without the sense of ambiguity between human and machine interaction. Besides, the deduction ability to inference information explicitly stated in the vocabulary of a domain and the relation structure of an ontology (i.e., this is known as reasoning). Deep discussion about reasoning on description logics can be found in [26]. Description Logic is considered as the most suitable knowledge formal representation family because considers to information sharing as a pillar in focus design. The modelling approach is decentralized in relation to schema conceptualization rather than data oriented. The reasoning feature is based on the open world assumption (i.e., model features expectation should closely approximate to the real world resemblance). More details of the paradigm can be also found in [23]. To sum up, the greatest asset of this paradigm is that it can be applied to any type of context.

A crucial aspect in semantic modelling with description logics is to specify a basic form of knowledge representation. This form is known as **Knowledge Base (KB)** [27]. A knowledge base is composed of two main components or also called “boxes” (i.e., sets of axioms) which are description logic constructs (see Figure 3). These components are:

TBOX (or Terminological Box): Is a set of logical concept constructs that are used to define explicit formal concepts (classes) and roles (relations) of a vocabulary’s domain. It is also known as the terminological knowledge of a KB.

ABOX (or Assertion Box): Is a set of logical fact constructs for property specification between the objects/concepts defined, roles and individuals (instances of data). They are also referred as “assertions” (i.e., factual description of the “known world”).

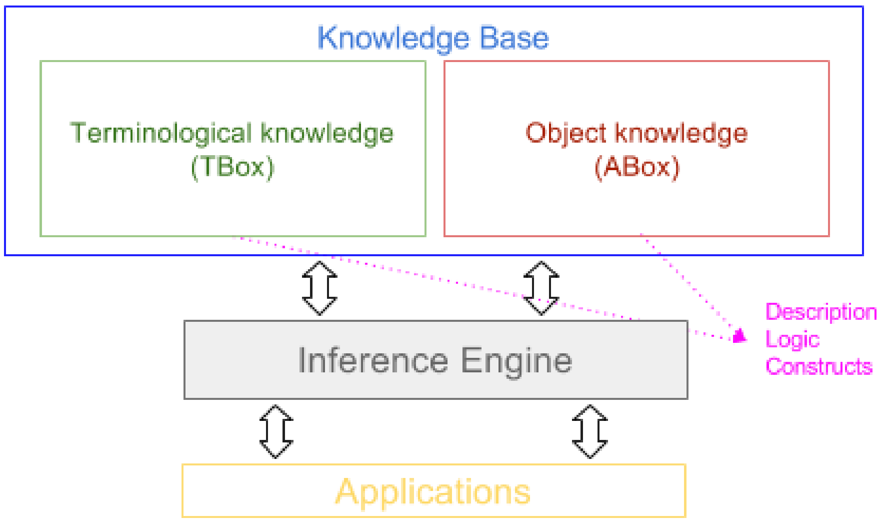


Figure 3. Knowledge Base Basic Structure[17]

Ontology does not have a formal definition, but it is known as the formal set of methods to encapsulate knowledge representation, this encapsulation procedure has limited level of identification and relational generation over types and entities. In addition, object class definition encodes specific domain conceptualizations in a hierarchical order.

A particular aspect of ontologies, is that they caryy inferenced knowledge inside constraints and object properties [28] that yields higher computational semiotics level for problem solving as described in [26].

Ontology modelling applications had been evolving in the last few years due to the rationality of the semantic web, the importance of deploying informational solutions for knowledge representation is reflected in any multidisciplinary science research field such as in Biomedical [29] for statistical analysis, Sustainable Science [30] for concept exploration, Genetics [31] in data reuse by phyloinformatics, Urban Computing [32], specially urban planning for smart cities modelling in [28]; Humanities research [33] for ontology based data federation among many others. Notice that the ontological approach provides to users a different path for organizational applications, by removing several and divergent integration between them (see Figure 4).

[17] This summarized adaptation was included from [87] for explaining purposes only

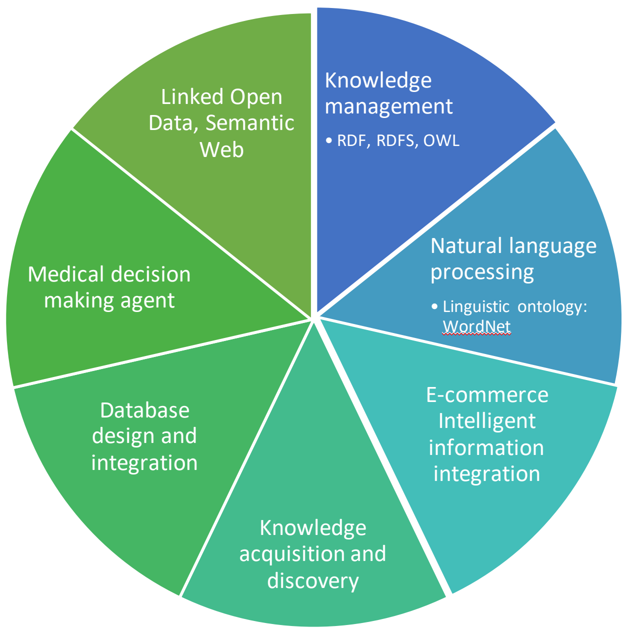

Knowledge management
• RDF, RDFS, OWL
Natural language processing
• Linguistic ontology: WordNet
E-commerce Intelligent information integration
Knowledge acquisition and discovery
Database design and integration
Medical decision making agent
Linked Open Data, Semantic Web

Figure 4. Multiple disciplinary spectrum of Fields using Ontologies

The evolution of ontologies' s use depends of the typology compilation that a domain scope has (i.e., domain granularity) or to the language expressivity selection [34].

For simplification purposes three different types will be considered in this document. **Upper ontologies,** used to describe new conceptual constructs like in [35], [36] or [37], **Domain Ontologies** (i.e., related to specific domains) like FIBO for finance industrial sector in [38], and **Hybrid Ontologies** like the ones detailed in [39], [40] o [41].

The general structure of an ontology may contain the following components regardless the underlying logical or syntactical power that they are built on (next referential structure details are considered from [42]). Simplified definitions are provided as it follows:

Classes: Are entities where a definition of a concept can be grouped by sets or collections of objects (things).

Relations: Are the actions or roles that denotes associations between two objects of a conceptualization. The type of association is binary.

Instances: Are the instances of the objects referred in classes or in relations, also called as individuals because of their elemental meaning in the ontology.

Attributes: Characteristics or parameter descriptions from concepts, also called as data properties.

Restrictions / Rules: Formal boundaries or natural statements that are used to set conditional inference towards ontological assertions.

Axioms: Logical expressions that set the constraint level of the ontology for domain description.

1.2.3 Ontology-Based Data Access and Integration (OBDA/OBDI)

Operational flaws from systems in organizations derivate from inherited traditional computational systems, which are based on outdated designs, traditional transactional implementations and legacy architectures. This reality produces data being stored in relational databases.

By considering this natural drawback, a paradigm that combines database access and ontology reasoning power (i.e., needed to increase semiotics in a domain context) as an information integration approach is key.

OBDA or Ontology-Based Data Access, is a modelling method started since 2000s, the primary goal is to design intelligent systems for data manipulation in database sources [43]. The core idea is to have declarative mappings specification for domain ontology axioms in order to expose data on RDF graphs from a relational data source. This RDF graphs are materialized in triples.

The idea is that triples do not materialize and remain virtual, then query processes are generated in a second stage where they are executed. Hence the common strategy used is query rewriting by avoiding recursion and property chains [44].

The complete set of detailed advantages of OBDA systems can be found in [45]. In a nutshell, they are schemed for domain specific patterns necessities. This means, that the conceptual schema conjunctively used in the model is also the basic interface with the Structured Query Language (SQL) database system attached. When combining relational logic with schemas, impedance mismatch is the main challenge. Further discussion about how impedance mismatch issue is generally addressed by this approach, is mentioned on Section 2.3.

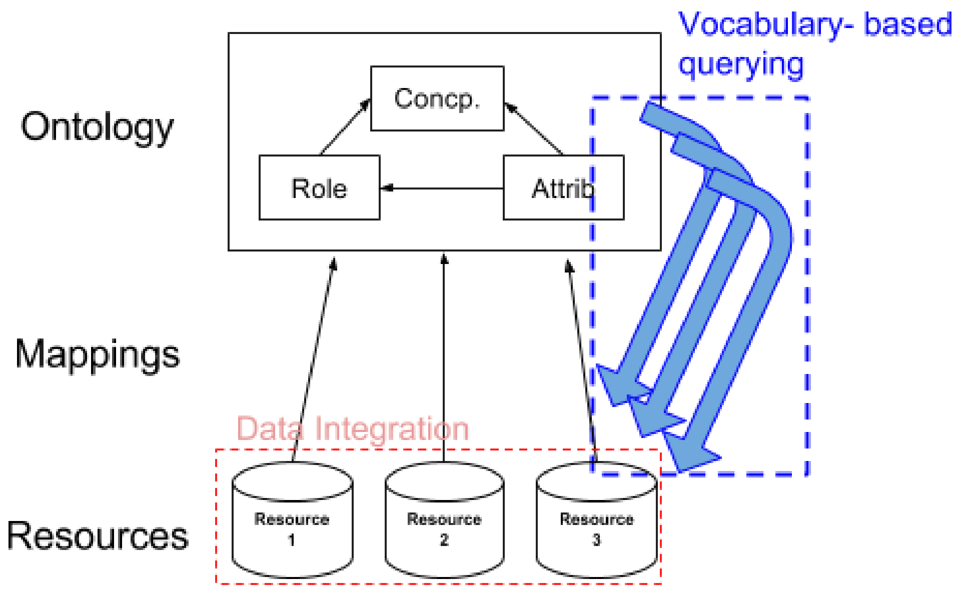


Figure 5. Extended OBDA Architectural Model

In Figure 5, notice that information integration is not the unique asset that OBDA systems could outreach, a global data schema “lookout” is possible from an appropiate query definition if correctly enforced. In Chapter 3, more advantages are mentioned in general but the central idea of using knowledge-based systems is that their level of expresivity and reasoning complexity is well balanced between ontology vocabulary and query processing for “Big Data” datasets [45] (i.e., “Data lakes”).

CHAPTER 2

2 Motivation and Goals

2.1 General Considerations

For developing a semantic linked solution two premises must be considered initially:

1. Interoperability is the core aspect in any Information System Design, it is also determined by the unification of conventions and standards. Paradoxically; these standards are not adopted because of heterogeneous perspectives and inconstant goals.

2. Some trending technologies are shaped from their detailed requirements and from the current state of the art. The singular meaning of promoting a solution and implementing it into a social system like an organization, makes that processes, dramatically change.

In this sense, and due the maturity level that has been reached over the last years in semantics field, a standardized way to model analytical systems should be part of the generic tone of any project development.

The task of gathering requirements for developing solutions sometimes, does not provide enough extraction power for a tacit knowledge of a domain. Specially, in the era where distribution systems are increasing data and making scalability as part of any project complexity in the evolution of a organization. For that reason, the externalization of requirements engineering are mandatory in their initial stages, a complete review of this discussion is detailed on [46].

Aligned with above idea, the main motivation for this study is to provide a guideline to implement OBDA in the educational context. Some related aspects are described:

- Traditional approaches uses modelling frameworks that are outdated or that cannot handle a superior quality of representation formalisms for any domain. High police maintenance in previous implementations and tedious upfront configuration derive into time consuming investment.

- The set of advantages from creating linkages between a formal knowledge base and raw data from a transactional sort of system, is that data would be accessed, stored, queried and available to compatible open source tools on regular hardware. But one must be aware that the centric orientation of this technologies could constrain system evolution as described in [47], once the information flow is standarized inside any proposed architecture.

- By providing a data access user tool which its main structure is composed of updated technology, based on a brainy engineering requirements and sufficient semantic annotation; this crude effort is meant to be a validation layer over the developed ontological solution implemented. The basis of the current work is to coercively provide a natural evolution of the current data to an “end-user access open window for information retrieval”.

Therefore, heterogeneous and federation data sources capabilities are aspects not being considered in the current scope. Instead, some other aspects related to user centric orientation over the semantic solution are being taked into account. In this sense, it is expected that, this ontology-aware solution will provide large data fast acquisition, superior concept correlation for data migration projects and finally self ontological mechanisms for recursive vocabulary improvement due the continuous data retrieval for the domain expert user.

2.2 Pretended goals and methodology applied.

The following objectives regarding to the educational context mentioned on Chapter 1 are:

- To translate into an OWL 2 QL profile, the resulting conceptual model based on the ontology definition for LA4S project realized in [2], this allows users to be able to use educational context vocabulary and portability to technical experts in future system designs.

  The ontology assembly will be completed by manual means; usign the best ontology editor, considering some OWL restrictions based on its analogous description logics (further details are described in Chapter 3), of the UML level of expressivity constructs (i.e., multiplicity and cardinality constraints not included in OWL 2 type). First, to define which ontology language is the most appropiate, a deep literature review is done in order to identify the most reliable one, Then, a solid knowledge of the set of constructs used by this DL are described for general reference. Finally on Chapter 5, the construction process specification is detailed to the most complete case with some examples in the ontology editor tool. The conceptual schema resultant can be easily understand and visualized by education context domain-users. From the technical expert's perspective, further developments are expected not to require expert domain dependancy. Also, this method may be applied to other general cases.

- To describe the technological OBDA implementation process of OWL 2 QL LA4S ontology, for reconciliate information between a data schema from a learning management database and a educational ontology.

  The use of an intermediate stage, like the one proposed by the OBDA model permits an evolution of data by adding great semantic power to information systems. Some other objectives are derivated:

  - To define a small set of engineered mappings, that allows to link tacit knowledge from a local database instance, with the conceptual model previously translated.

The rationale behind this goals is better explained in the following statements:

In order to achieve the first objective, the UML ontology outcome from LA4S is being taken as a reference for translation and after a deep analysis of characteristics in former studies of the most appropriate DL language (i.e., according to the correct trade-off between expressivity and reasoning ability), the OWL 2 QL profile is chosen. The criteria description of the construction tool that is being used is on Section 3.2 .

Likewise, to achieve the second objective a short but consistent literature review was applied. The current effort is inspired in the research performed in [48]. This study it leads to a set of steps in order to facilitate access problems to massive amounts of data, and proposes a description logic version for the problem of the impedance mismatch between data items and objects. A simplified workflow of this logic is described as it follows:

(i) To choose an ontology language that suits large amounts of data and that has the sufficient level of expressivity without rising the level of computational reasoning complexity.
(ii) To perform a in-depth data source, domain-expert supported analysis to define the most accurate mapping language; that avoid structure differences between data sources and ontology based elements, and map tables from a provided dataset.
(iii) Test query validations of the functionality of the model implemented

The ground idea of the sub goal derivated, is to create a virtual schema layer. This is known as an hybrid approach, where declarative mappings act as rationalized data flows for data discovery and/or integration for end-users. It also represents for technical experts, a feasible way for system maintenance. The logic underneath is to define an explicit level of semantic power on top of the schema global view of the operational information, without intermediate datawarehousing processes or to complement their platform robusteness. By adding direct extraction power for data and translate information for specific domain mapping requirements, LMS sources and Learning Analytics concepts are being blended.

CHAPTER 3

3 Description of the Technological Criterion

3.1 Selection Criteria for the underlying Description Logics

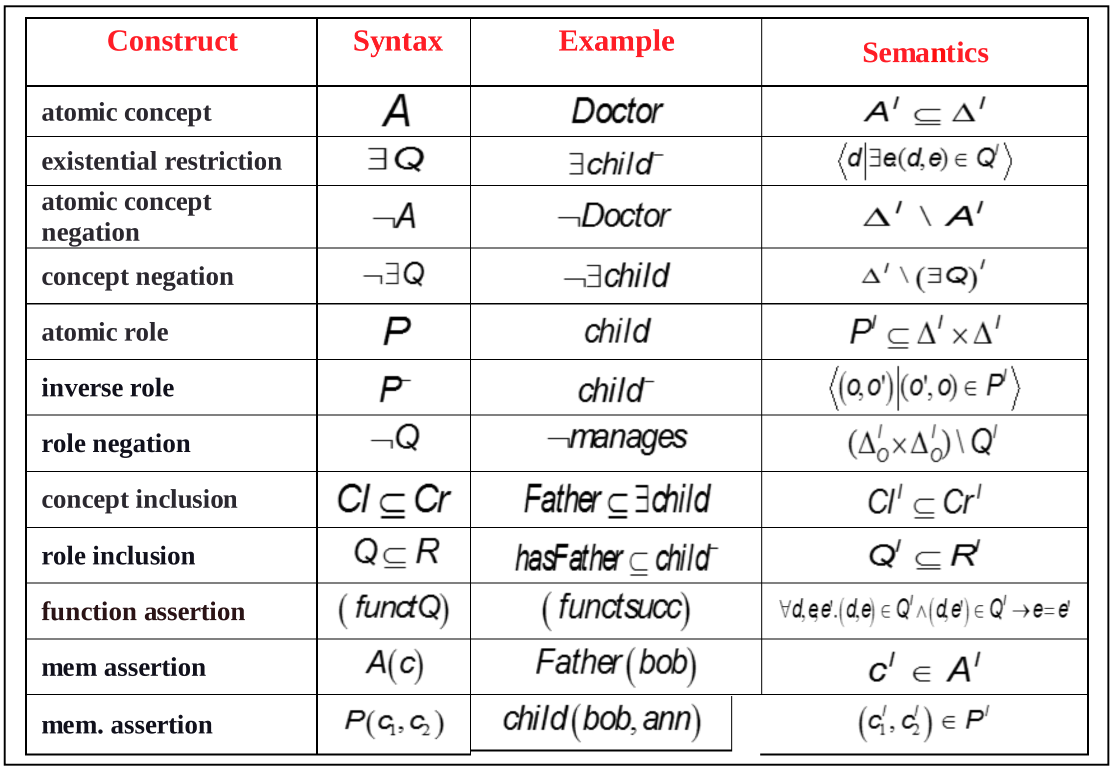
Despite of the existence of some methods to transform UML models to OWL avoiding OCL constraints in an arbitrary mode like in [49] or serialization procedures like the one presented in [50] based on XML – XMI definitions; the fact of performing a "bottom-up" manner for a KB implementation, regardless the context, is more time consuming. Additionally, the potential outcome from TBOX and ABOX components could include intermediate vices that need to be fixed. In other words, logic construct crafting in DL language expressions and data duplication in a given Relational Database Management System (RDBM).

Likewise, is essential to determine from the DL gamma, which family is the most appropriate for the ontology transformation at hand. Hereof, a retrospective view from many literature sources will allow suitable selection criteria.

Originally, DL-Lite family definition is considered aside from the heavy weighted description logics, which were created to capture most of the features of data modelling approaches. This differentiation allowed an intensive yet simple way to express any domain representation by keeping the reasoning level of complexity low [51].

For accomplishing the ability to capture modelling features, a superset of constructs was defined to enable the description of formalisms used in the dual logic of Entity – Relation. Approach that use databases and is also helpful for class diagrams definitions in UML, Table 1[18] expands the syntax and semantics of the mentioned set of constructs for reference.

[18] Adapted from Master's lecture in [87]



Construct	Syntax	Example	Semantics
atomic concept	A	$Doctor$	$A^I \subseteq \Delta^I$
existential restriction	$\exists Q$	$\exists child^-$	$\langle d \mid \exists e (d,e) \in Q^I \rangle$
atomic concept negation	$\neg A$	$\neg Doctor$	$\Delta^I \setminus A^I$
concept negation	$\neg \exists Q$	$\neg \exists child$	$\Delta^I \setminus (\exists Q)^I$
atomic role	P	$child$	$P^I \subseteq \Delta^I \times \Delta^I$
inverse role	P^-	$child^-$	$\langle (o,o') \mid (o',o) \in P^I \rangle$
role negation	$\neg Q$	$\neg manages$	$(\Delta_O^I \times \Delta_O^I) \setminus Q^I$
concept inclusion	$Cl \subseteq Cr$	$Father \subseteq \exists child$	$Cl^I \subseteq Cr^I$
role inclusion	$Q \subseteq R$	$hasFather \subseteq child^-$	$Q^I \subseteq R^I$
function assertion	$(funct Q)$	$(funct succ)$	$\forall d,e,e'.(d,e) \in Q^I \wedge (d,e') \in Q^I \rightarrow e = e'$
mem assertion	$A(c)$	$Father(bob)$	$c^I \in A^I$
mem. assertion	$P(c_1, c_2)$	$child(bob, ann)$	$(c_1^I, c_2^I) \in P^I$

Table 1. DL - Lite Semantics

Notice that from previous table some additional composability may be group as: **"Is–a" conception for classes** (i.e., which simplifies the ability to specify taxonomies between objects) syntactically represented as $A_1 \subseteq A_2$, **disjointness between classes** noted like $A_1 \subseteq \neg A_2$, **domain/range of roles** expressed such as $\exists P \subseteq A_1 / \exists P^- \subseteq A_2$ and **mandatory participation of roles** represented like $A_1 \subseteq \exists P / A_2 \subseteq \exists P^-$.

Previous classification of constructs shows the power of DL - Lite family to express UML semantics. Additionally, the right trade-off between levels of expressiveness and data complexity are satisfied by two additional subtypes of the family which includes the following additional group of constructs:

- DL – Lite$_F$: **Cyclic assertions**, **inverse on roles** and **functional restrictions on roles** (i.e., syntactically represented like *(funct P) / (funct P⁻)*) as described in [52].
- DL – Lite$_R$: **RDFS fragments**[19] plus **"Is–a" conception for roles** (i.e., syntactically like $P_1 \subseteq P_2$), and **disjointness between roles** like $P_1 \subseteq \neg P_2$. As previously mentioned on Chapter 1, OWL 2 QL[20] profile is based on this Lite family.

Next Table 2[21], summarizes the groups of constructs (i.e., language aspects) of OWL 2 QL Profile which is based on DL – Lite$_R$ and its supported and unsupported features with a functional example aside:

[19] Fragments detailed on - https://www.w3.org/TR/2014/REC-rdf-schema-20140225/
[20] DL – Lite R is the base for OWL 2 QL - https://www.w3.org/TR/owl2-profiles/#OWL_2_QL
[21] The table is a compilation of the current research done on OWL 2 QL

SUPPORTED	
LANGUAGE FEATURE	**Functional Syntax**
SUBCLASS AXIOMS	SubClassOf $(C_1 C_2)$
EQUIVALENCE	EquivalentClasses $(C_1 \ldots C_n)$
DISJOINTNESS	DisjointClasses $(C_1 C_2)$
INVERSE OBJECT	InverseObjectProperties $(P_1 P_2)$
EQUIVALENCE	EquivalentObjectProperties$(P_1 \ldots P_n)$
DOMAIN	ObjectPropertyDomain(P C)
RANGE	ObjectPropertyRange(P C)
DISJOINT	DisjointObjectProperties$(P_1 P_2)$
SYMMETRIC	SymmetricObjectProperty(P)
REFLEXIVE	ReflexiveObjectProperty(P)
NOT SUPPORTED	
LANGUAGE FEATURE	**Functional Syntax**
EXISTENTIAL	ObjectSomeValuesFrom(P C)
REFLEXIVITY	ObjectHasSelf(P)
INDIVIDUAL VALUE	ObjectHasValue(P a)
ENUMERATION	ObjectOneOf$(a_1 \ldots a_n)$
UNIVERSAL	ObjectAllValuesFrom(P C)

CARDINALITY	ObjectMaxCardinality(n P)
	ObjectMinCardinality(n P)

DISJUNCTION	ObjectUnionOf
FUNCTIONAL	FunctionalObjectProperty(P)
KEY	HasKey $(C(P_1 \ldots P_m)(R_1 \ldots R_n))$
TRANSITIVE	TransitiveObjectProperty(P)

Table 2. OWL 2 QL language features

Moreover, DL – Lite$_R$ family is considered the most appropriate floor to convex the complexity between UML modelling expressiveness and relational querying answering trade-off. This notion is well known in the Semantic Field and the reason states from the potential taxonomic complexity of polynomial time (PTime) w.r.t. the size of the TBOX and, also the potential data complexity size of Logarithmic Space (LogSpace) w.r.t. the ABOX defined under this type of description logics [51]. The following cite corroborates last statement and is textually included from [48]:

"These logics allow for answering complex queries (namely, conjunctive queries, i.e., SQL select-project-join queries, and unions of conjunctive queries) in LogSpace with respect to data complexity. More importantly, after a pre-processing phase which is independent of the data, they allow for delegating query processing to the relational DBMS managing the data layer."

However, dealing with large taxonomic or data complexity limits direct access to the information required by end users; therefore in [43] and [48] the construction of a descriptive logics that entailed the best of subtypes $_F$ and $_R$, (named DL- Lite$_A$) started. This was the basis of MASTRO framework system [53] and dealt with more emphasis the problem of impedance mismatch, particularly on structure and manipulative matters.

Since then, research intensification about this topic has been increasing in the last 5 years. This is reflected in some work improvement done for lightweight DLs like: improvement of conjunctive query answering as in [54] and high performance query rewriting in [55] with some extensional constraints in [56]. Many of this technologies have been oriented to deliver feasible solutions to variety and value challenges described on Chapter I.

Evolutionally in [57], Quest was presented as the SPARQL query rewriting engine, that based its core on efficient reasoning with large volumes of data allowing the use of some OWL 2 QL and RDFS technologies behind it. Afterwards was called Ontop in [44], and basically includes most of all standard technologies used for Linked and Semantic Data.

In Table 3[22], a summary of at least ten years of ontology based for knowledge modelling research, into the language profiles used as conventions nowadays is presented. The table includes a complexity comparison depicted by measurement features in taxonomic, data and query perspective:

Language	Taxonomic	Data	Query
OWL 2 EL	PTIME-c	PTIME-c	NP-c
OWL 2 QL	NLogSpace-c	In AC^0	NP-c
OWL 2 RL	PTIME-c	PTIME-c	NP-c

Table 3. Complexity types between OWL 2 Profiles

Is by these, previous heavy-supported considerations, and also to provide a standard alignment with W3C conventions, that the chosen ontology language for transforming the given conceptual model for LA4S is the profile OWL 2 QL.

3.2 Selection criteria for ontology editor

For the sake of practicality, Protégé has been chosen due its direct nature of OWL 2 ontologies specification capability. Further, some pragmatic criteria beneath this selection are described under two points of view:

(i) **Technical**: Is based on process details or the “methodum” elements used to develop something. From this concept, the most prominent technical characteristics related, are the high level of logic representation reflexed in multiple inheritance set-up capabilities and hierarchical relationship quality. This allows meta-classes and instances specification support, combined with constrained axioms explicit definitions.

Additionally, ontology consistency checking from reasoning services (i.e., consistency algorithms) trough plugins. Provide to this tool a pre-validation stage for ontology correctness. In matters of OWL 2 direct reasoners, there exists Hermit and FaCT++.

(ii) **Technological**: Refers to the level of abstraction that the tool possesses in respect of the lurking technology where is based and how it interacts with other interoperable facilities. In this case, a great set of features are:

[22] Adapted from Master’s lecture in [87]

a. Is open-source, based on BSD 2-Clause license[23].
b. Is plugin-based (i.e., functional extensibility trough intermediate developments)
c. Allows primary interactive navigation of objects (e.g., views, menus, tabs, etc.)
d. Refactoring operations are permitted.
e. Improved friendly-user interface, current on its 5th version release.

Protégé is also considered the leading content or knowledge management technology for ontological engineering because its enriched editing environment [8] as mentioned above.

3.3 Agora Dataset details

In this section is detailed minimum general aspects of Agora Moodle database, specifically to technical dataset structures related to the current Moodle version 2.8 customized by UPCNet. Also a description of drawbacks from the local dataset worked on is depicted. Finally, some highlights and argue about how previous technologies, can address the impedance mismatch issue are mentioned in general.

For any Moodle[24] virtual environment, a relational database is the core data storage component included on a “sandbox” architecture. The abstraction layer of the database component supports principally RDBMs (e.g., MySQL, PostgreSQL, MariaDB, Oracle and SQL Server) with a solid security platform included.

Moodle as a modular system, is designed to work with modules that behave with a hierarchical tree logic in order to increase process and procedures performance, thus provides interdependency between events within its internal system operation [58]. Therefore, the system database structure depends from the principal modules; users and activities that are in turn related to additional transactional tables that stores complimentary data.

Whereas this core is used by organizations for indistinct purposes, and since the database structure schema may differ one from another in heterogeneous environments; it is important to stablish general guidelines in order to extrapolate mappings efforts to any suitable case.

For the present scope of the study, privacy and data protection are the main pillars on which grasp reduction derive into a subset of the original schema from Agora, therefore preventing the access to the mentioned core (i.e., users and activity tables) directly. This means that information availability is constrained to data extraction efforts from the exploratory analysis performed on [2] (i.e., LA4S data extraction and data processing platform). Some structure logic could be aligned to most of the general cases of Moodle deployments based on the research done from Moodle official site[25]. The reason is that ideally, OBDA model should be directly implemented on top of the Agora production database as the semantic web cake suggests on Section 1.2.2 .

On one hand, a completeness ontology validation is not being considered regarding Agora system instance. This additional limitation affects directly to the starting point of the mappings in terms of information consistency. However, some complex mapping

[23] Further information of this type of license can be found in https://opensource.org/licenses/BSD-2-Clause
[24] More information of Moodle general architecture cab be found in https://docs.moodle.org/dev/Moodle_architecture
[25] Properties of events and more developers information in https://docs.moodle.org/dev/Event_2#Properties

definitions and basic optimization techniques are expected to be executed. Integrity constraints were rebuild on the set of tables which contains relational information, this task was done according to Moodle official documentation[26].

On the other hand, it is worth to mention that current data available from the Agora database dump is not reliable for the proposed conceptual schema, hence the domain's knowledge representation of the Learning Analytics system would be incomplete. Therefore, the resultant coarse outcome of declarative mappings to the global database schema, are just a clear-cut subset of the domain in question.

Due to simplification reasons, a fragment of the most relevant tables of the relational database are shown in Figure 6. This standard E-R diagram shows boxes that represent tables with their names at the top. Below rests the first section divided by a line that represent a list of columns. Relations are represented by arrows, where the head means the origin of the referenced table using its primary key and how this attribute becomes in to a foreign key to the forked part of the arrow in another table. Any relational approach for developing database systems uses this principle as a rule. The rule is known as referential integrity constraint [59].

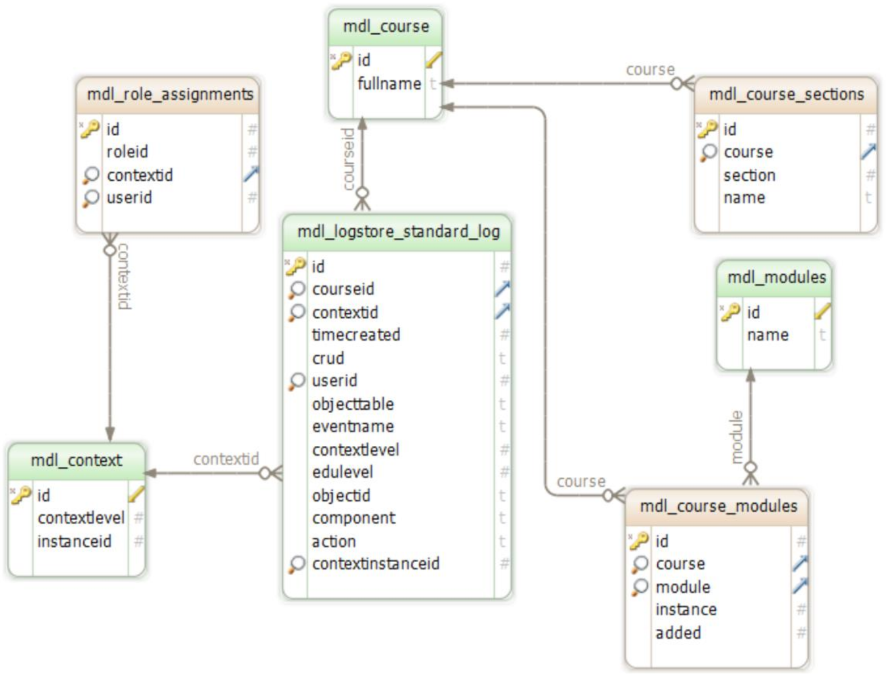


Figure 6. A fragment of the relational database of Agora

Tables names are composed of the prefix "mdl" joined with the actual name of the table. In particular, notice tables "mdl_logstore_standard_log" (from now on refer just like **"logstore"**), "mdl_role_assignments" (from now on **"role_assignments"**) and "mdl_course_modules" (from now on **"course_modules"**) which contain the core information and will be mostly used for next implementation description. **"logstore"** table contains data related to all the events from the LMS, such as identification number and reference columns from the other relevant tables: "role_assigments" and "course_modules". **"role_assignments"** stores data about the user id (userid) and the role id (role_id) and table **"course_modules"** stores data about the object learning id

[26] http://www.examulator.com/er/2.9/moodle29_erd.png and https://docs.moodle.org/dev/File:Course_modules_database.png

(id), the course where this object belongs (course) and the module or object subtype (module) where it comes from.

It is worth of mentioning, that the id from users in “role_assignments” does not keeps the property of uniqueness because the column userid is not a primary key. This could cause additional “join” operations and “distinct” functions in mapping definitions, causing bad performance in querying.

As part of a set of well known, information technical difficulties when delivering data integration solutions exists impedance mismatch. The origin of this problem occurs when applications made in an object programming language are served from the data of a RDBS. At a deeper level, is the attempt of the mixing of declarative and procedural programming paradigms, causing that there is no cohesion and common syntax management and disparity in the set of data types for each case. Causing in practice, additional efforts for syntax conversion for interfacing Object Oriented (OO) applications and RDBs on code scripting.

For developing intelligent systems, this is also an issue that needs to be taken into consideration. In fact, the attempt to transform data from a physical storage system into a set of queried concepts or a portion of it; by means of a specific domain vocabulary is not a trivial task. It implies to implement a formal mapping procedure in order to blend object and relational conceptions trough a principle of identification. Through the construction of data identifiers, it is possible to map the potential abstract objects that represent the ABOX component of any ontology (i.e., to materialize the consulted information) bypassing the interfacing problem from impedance mismatch. The proposed method behind this approach is detailed in terms of descriptive logics in [60] inspired by [61] for further minutiae.

CHAPTER 4

4 State of the Art

4.1 Virtual RDBS Querying

Taking into account data changing dynamics and management model implementations; traditional approaches requires to complement themselves with new and tested technologies when any organization tries to access their transactional systems without resource shortages [62] , [63].

As mentioned earlier, the great bottleneck problem is when any analysis to relevant data is performed. This action will crave multiple people roles involvement, some of them had the role to interpret and some other, the role of extracting information requirements at the same time, using the same interface channel.

In this way, domain experts clearly would depend of the participation of IT engineers for producing ad-hoc queries that in most of the cases are embedded on applications. Also, if their need changes a functionally requirement to new adaptations will cause time consumption for redesigning structure from working platforms but with a high potential of query requirements mismatching.

This man-in-the-middle is an outdated approach if it comes as merely a standing and static metadata solution, limiting the potential of the implicit knowledge evolution in any sort of organization (e.g., including industry, business, academic, research, etc.). The main reason relies on the traditional use of known paradigms such as: transactional database and object oriented modelling that were not created to fulfil any domain semantics requests [64].

For instance, in the context of developing urban models for city utility systems (e.g., building projects, water utility, energy conditions, biodiversity, waste facilities, etc.) is that in case that this model would change over time, new schemata assembly for functional applications and transitively, data storage issues may rise. These are potential problems because they involve the heavy task of designing patched backend proposals that affects to several interconnected subsystems that works concurrently.

The reckoning question then is how this can be managed, if 80-90% of organizations are dealing with structured or semi-structured data worldwide. The typical exploitation techniques do not have the scalable ability needed, to share semantics between relational database systems. The reason is that on ETL approaches, data exploration is too strict as they require predefined access patterns; thus partial semantic gap is just being answered for local, static-format enquiries.

A brief but concise summary of most of the representative use cases for the academia, industry and research perspective (which are of great interest for the current review in this work), are applying ontological solutions in their backend architectures for answering the kind of issues that were previously noted.
On 2013, a case study over complex situation recognition using OWL 2 DLs for a specific service management system was detailed in [65]; the core idea was to make an inference tool solution that recognizes adaptation techniques for balance loading in a service distributed system. This tool was processing realistic amounts of data to perform the reasoning from the DL formal definition stated under the classes defined in the TBOX,

and from another independent database instance of an ABOX where the assertions materialized instances from the original data. Then it was tested over the most common reasoned engines related to the OWL 2 profiles analysed (QL & RL) finding fast and powerful level of querying from light weighted ontologies over conventional data source types.

From 2014, another case study from Siemens Energy company [66]; describe that the analysis of sensor data monitored process was enhanced by the use of three integrated local predefined ontologies within the Optique system [67] (that is based on the –Ontop- platform), using OWL 2 QL profiles. The result then, was positively assessed by end users who evaluate three main aspects: data access integration, fast query formulation and implicit information derivation. In the same year another research article begun to formalize streaming data access by introducing stream and temporary structure features query language (STARQL) on OBDA paradigm [68] used for continuous domain-context queries.

Furthermore, from an historical-academic context, the article [33] on 2016, show details from a integrated semantic endpoint that allows historians experts to perform queries about food production from the EPNet project. This effort took into account three different data sources but that conceptually stored relevant query information. The backend is based on a fragment of the ORM2 conceptual model that has been transformed into OWL 2 QL. Then, with some mappings trough –Ontop- the system allowed end users to use their domain vocabulary to merge and extract useful information.

Finally, a publication on 2016 describes the same approach that was established to federate the access to static and stream position data of vessels as part of the Real-time Maritime Situation Awareness System (RMSAS), producing a detection tool for routine traffic and abnormal vessel behaviour [69]. External federation of external SPARQL endpoints with local –Ontop- ones was achieved.

It is worth to notice that, -Ontop- framework seems to be the tailored decision for delivering an integration layer based on OBDA modelling that grants early data access to end users, extrapolating a natural development capacity for embedded semantic systems to the analytical ability intended by the adjacent LA4S project.

4.2 Ontop: An OBDA Tool description

Aside from data integration, it is important to emphasize the potential contributions to the current scenario; in this case, by adopting an approach of Ontology Based Data Management as is detailed in [70] the resulting immediate aftermath, is summarized in:

1. The unification of informal conceptualizations between expert groups involved in sharing the same domain specific terminology

2. Standardization of information through a methodology that makes data tractable; data which initially, has lost its original modelling shape over time.
3. The creation of certain accessibility levels of representation of information, for a non-expert audience in technology databases, and that this specific knowledge is biased to a small group of experts in systems.

To simplify the technological use of this kind of approach (i.e., OBDA system), many technical details must be taken into account to build a robust solution despite the existing documentary limitation. In this section, technological features from –Ontop- tool will be briefly described, certain native standard techniques used for linking data are mentioned, and a brief feature comparison between OBDA most prominent tools will be exposed.

-Ontop- in general terms, can be defined as the mappings specification tool from a data source; that traduces data with an ontology vocabulary predefined in OWL 2 over RDFS conventions and under a OBDA approach, semantic querying integration for end-users.

This tool, was developed in sound base foundations and supports most popular W3C standards [44] such as RDF/XML, R2RML, SPARQL, SWRL [71], and the OWL2 QL entailment regime in SPARQL detailed in [72]. By the use of the principle of global-as-a-view (GAV) in mapping declarations, the ontology expressiveness (i.e., level of vocabulary expressions and/or concepts) permits recognizable information for conjunctive queries. This tool also uses optimizing techniques for SQL query rewriting over a data source.

The main components of –Ontop- are related to data access integration. This core functionality is attuned with: the database support technology, a reliable mapping language and the query rewriting engine for a data workflow.

There also exist some complementary components that will be not detailed. These are related to mapping / ontology bootstrappers like BootOX referenced in [73], reflexed-ontology matching of LogMap mentioned in [74], data source and query federation for ontology-based interfaces like the one cited in [75] (Optique).

In terms of database connections, -Ontop- relies in Java Database Connectivity Framework (JDBC). This allows the tool to integrate to most of the relevant and standard database engines, like MariaDB, MySQL, MonetDB , PostgreSQL and Teiid (which is typically used for data virtualization services); and from a commercial perspective to DB2, H2, SQL Server and Oracle[27]. For the connection to a local database instance provided in MySQL, a Connector/J driver installation is being considered[28] is this study.

For declaring mappings over relational data sources, -Ontop- supports R2RML[29] mapping language. This language takes as an input to a table, view or a SQL query; and without modifying the database, virtualize logical tables in terms of triples maps (see Figure 7 for a better illustration).

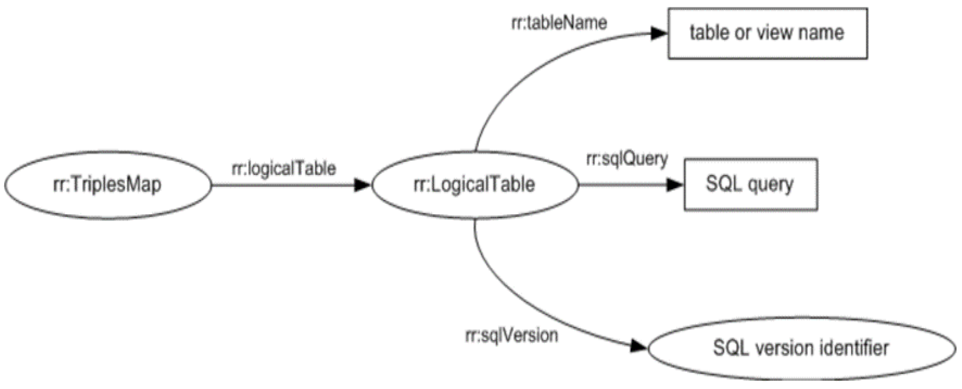


Figure 7. Logical Table in R2RML[30]

[27] Further information can be find in: https://github.com/ontop/ontop/wiki/ObdalibPluginJDBC#JDBC_Connections_and_Drivers
[28] Installation Instructions for MySQL are in https://dev.mysql.com/downloads/connector/j/
[29] Descriptive information on mapping convention standard in https://www.w3.org/TR/r2rml/ and https://github.com/ontop/ontop/wiki/ObdalibRdb2rdf#Description
[30] A more detailed explanation of R2RML can be found in W3C Consortium web

In order to simplify the use of previous mapping language structure, the tool offers its own native mapping language that can be serialized also in a native format file.

The structure of this mapping file is composed of three parts: mappingId, source SQL query and target triple template[31]. The first is an identifier, where a name is arbitrary assigned. Next, the target template is a subject-predicate-object (SPO) graph declaration that can be written in turtle [44]. Lastly, is the SQL query definition block which a predefined request over the values of a relational schema may be indicated. Listing1 represents an example of a standard mapping declaration.

```
mappingId      mapping_example_id
target         :a_concept_template/{pid} a :Concept .
source         SELECT pid FROM tbl_example
```

Listing 1. Mapping Example Structure

A set of mappings are declared inside the serialization of ". obda" file format from the tool. The main reason for choosing the native mapping option, is that is more compact and uses turtle[32] syntax for mapping declarations (i.e., they are human readable).

As mentioned earlier on Chapter 3, **Quest** (i.e., -Ontop- reasoner) is the query rewriting engine of the tool. The core is based on a rewriting algorithm, that translates SPARQL queries over virtual RDF graphs to its analogous SQL type in the relational source. The algorithm name is Tree Witness Algorithm and it is detailed in [76] for further reference.

Alike its predecessors QuOnto or MASTRO-OBDA System [53], -Ontop- uses KB building techniques for ABOX components under a "bottom-up" manner detailed in [48], which is called Classic ABox[33] mode (this setting feature is located in Data Configuration section of the reasoner preferences as part of the plugin in Protégé editor). Under this method, the reasoner works as a typical OWL triplestore. The engine takes data assertions (i.e., individual assertions) and materialize them on a specified data base. The engine considers the TBOX axioms in the ontology to perform query rewriting over relational data sources and retrieves information.

In addition, this tool also includes a "top-down" method that dispenses the materialization of the ABOX, called "virtual mode" which makes it innovative in the sense that it does not duplicate data and does not require to increase resources in a constrained environment.

In the realm of query answering systems for Linked Data, there are two known segments that applies the technology mentioned. These are triplestores and OBDA frameworks [44]. In next paragraphs simplified descriptions from several tools are mentioned. Notice that is considered just the criteria of using OBDA models and that the OWL 2 profile is the basis for the current ontology construction. All popular triplestore frameworks (like Virtuoso, GraphDB, Stardog y RDFox and sofort) are not part on the next comparison.

[31] Technical details are in https://github.com/ontop/ontop/wiki/TurtleSyntax
[32] Human readable syntax for RDF conventions https://www.w3.org/TR/turtle/
[33] A brief description of both modes is depicted on https://github.com/ontop/ontop/wiki/ObdalibQuestIntro

By contrast, only query rewriting tools will be pointed out with their most significant capabilities.

D2RQ[34] is the first crude-attempt of OBDA system development. This tool relies just on some limited query optimizations without any reasoning capabilities. The framework supports a native mapping language and a limited fragment of R2RML [77]. It also has a simple interface for data graph navigation and limited direct mapping installation [66].

Morph-RDB is based on self-join eliminations as the query optimization technique used. This tool does not have inference facilities as it was initially built for supporting Direct Mapping standards on R2RML engine[35]. It lacks of bootstrapping techniques and friendly user interface mechanisms for query formulation [78].

In its beginnings, Ultrawrap[36] didn't' support inference capabilities but, with an extension with QODI system [79] (i.e., includes inverse and transitive properties over RDFS [80]), Ultrawrap incorporates a query driven way to implement OBDA systems by a built-in ontology import feature. It also lacks of any query visualization technique for user oriented aspects.

Mastro as previously mentioned on Section 3.1, supports reasoning, particularly on OWL 2 QL ontologies. The main drawback is that aims just to a fragment of SPARQL querying (i.e., conjunctive queries) making it hard to define a flexible set of specific-domain inquiries in a system. Analogously as Ontop, it has a limited support on data federation that it has been improved for R&D over the last years as it is specified in [81], [82] and [70]. This tool lacks of a complete deployment support on bootstrapping [66].

From the set of tools presented; excluding Mastro and Ultrawarp whom provide academic and commercial licenses, the rest offers Apache 2 license type. Additionally, just Mastro, Ontop and Ultrawarp provide SPARQL end-points with predefined queries capabilities[37].

To sum up, -Ontop- is the most complete option for OBDA implementations and the most complete tool that includes the most reliable Linked Data technologies. The advantage of working with native mapping declarations adds an edge over the rest of tools. This use of the ".obda" format file for serialize mappings plus the powerful engine of inference in RDF/XML ontologies and the capacity of SPARQL querying is transparently parsed when implementing OBDA. In this sense, the merging feature over all the set of technologies used, permits that the engineering work become more intuitive for the construction of deductive queries in any data schema form. This capability becomes a user-friendly practical tool that decreases the learning curve for getting involved in semantic technology practices.

[34] Technical details on http://d2rq.org/

[35] Adapted from https://github.com/oeg-upm/morph-rdb

[36] Capsenta comercial web of the tool in http://capsenta.com

[37] Ontop has less restrictions when developing SPARQL queries over endpoints.

CHAPTER 5

5 OBDA model implementation for LA4S Ontology.

In this chapter, the definition process of the domain ontology construction for LA4S and the OBDA model implementation will be described. Some general restrictions are considered in the domain assembly regarding the UML model that it is taken as an initial reference. Likewise, some mappings limitations will be exposed in the implementation process as well.

5.1 LA4S Ontology Construction Process

Taking into account the conceptual reference model presented in [2], and in order to transform its classes for future transformations; a description of the most prominent taxonomies and classes, attributes, associations and some particularities will be presented.

5.1.1 General considerations

The ontology had been formally modelled in UML by means of Modelio framework[38]. The main drawback is that this design tool does not possess an automatic feature that allows to transform / convert the ontology into a reliable OWL 2 QL profile. For the implementation of the query answering under an OBDA implementation, this is mandatory. It is also important to consider the actual size of the ontology and all the particularities given in the constraints.

The UML model (see Appendix A is composed of a total of 63 classes; 61 are mean to be singular entity definitions and 2 are associative classes. There are also a total of 42 relations; 38 are associations and 4 are aggregations. To the interest of the present work, and since any DL language chosen does not support identification constraints encoding, n-arity associations and/or associative classes[39]; a reification technique has been implemented for an associative class (named UserInteraction) and the relation that associate the interactions that a user has over a learning object (interacts). The reification basis of this class is adapted from the logic of the method proposed in section 6 in [83]. Additionally, aggregation relations are considered as equivalent to the association ones under the perspective of the RDF schema constructs.

Moreover, it is worth mentioning that the fragment of how user interactions and learning objects relate between them, will be used for query-answering mapping and testing, and some vocabulary extensions may be added for concept relevancy rather than domain modelling.

Going further, an association definition in UML is represented in two object property's in OWL syntax, each one is the inverse of the other. In other words, an association end (also called role) translates as a single object property.

[38] Website on: https://www.modelio.org/

[39] Based on the model differences between RDF and UML in: https://www.w3.org/TR/NOTE-rdf-uml/

Integrity constraints from a database are of crucial importance for the tool. To model them axioms of domain must be explicitly specified in object properties in the ontology. In addition, OCL constraints are not considered in this transformation as they were not explicitly specified in the document description of the LA4S ontology in [2].

Some construction considerations are part of the unsupported features in the OWL 2 language chosen. Specially those refer to the existential and cardinality axioms that are used to describe multiplicity of members in classes and the cardinality of elements instantiated (see Section 3.1). Therefore, 8 associations that includes specific multiplicity (i.e. 1..*, 0..1, 1) from the original UML model, would not be translated with all the level of expression originally intended. Alternately, they will be represented in the allowed notation from OWL 2 QL syntax.

All previous considerations, will not affect to the reasoning engine of the OBDA tool. The input used for parsing the ontology requires a lightweight version, that is the most appropriated for query rewriting techniques [52].

It is expected that, rather than cause an inaccuracy, the techniques that are going to be applied help to point out an initial aspect related to the improvable expressiveness in the ontology. In what level is possible to transform the pretended context domain, and how to provide a reliable query answering service.

5.1.2 Taxonomies, hierarchies, classes and properties modelling

The three main taxonomies that will be depicted for this transformation process and their classes starts from the traditional taxonomy of **Thing**, "where everything is a thing"[40] defined in schema.org. The first is named **CreativeWork**, the second derives from the **Intangible** superclass, and is called **User**, and the third is **Event**. (see Figure 8 and Figure 9 for illustration of the hierarchies)

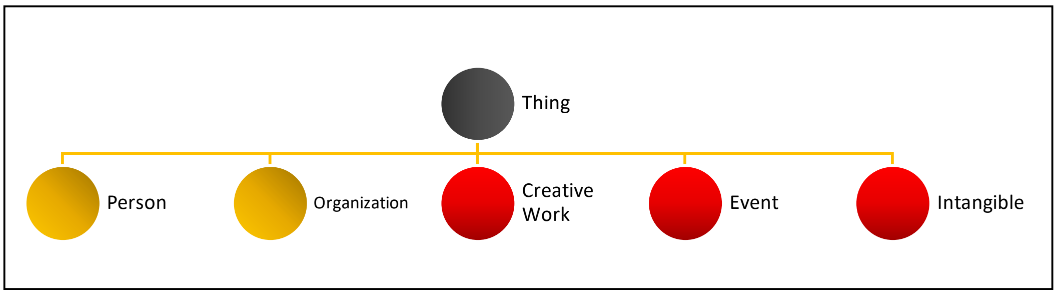


Figure 8. Hierarchy of Thing

On next paragraphs, every concept will be encoded with the form: "***:[Class_name]***" notation in order to identify the name of the UML class transformed to the OWL axiomatic like-syntax (i.e., Turtle) as an example of the semantic language used.

[40] A thing: "the most generic type of item", definition taken from https://schema.org/Thing

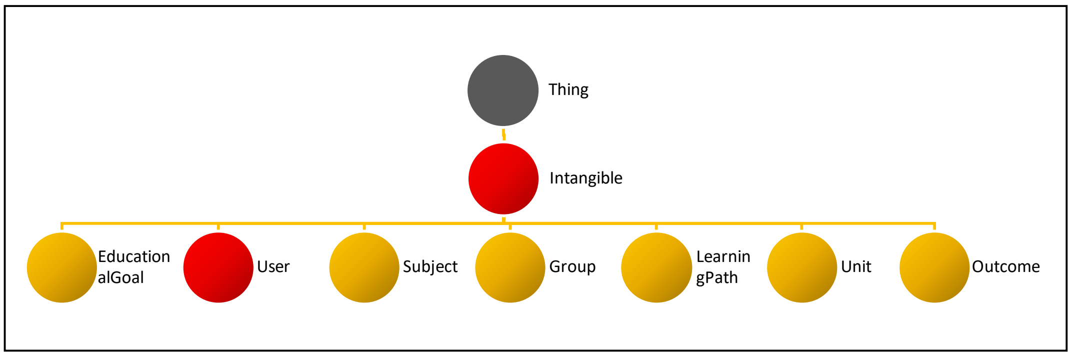


Figure 9. Hierarchy of Intangible

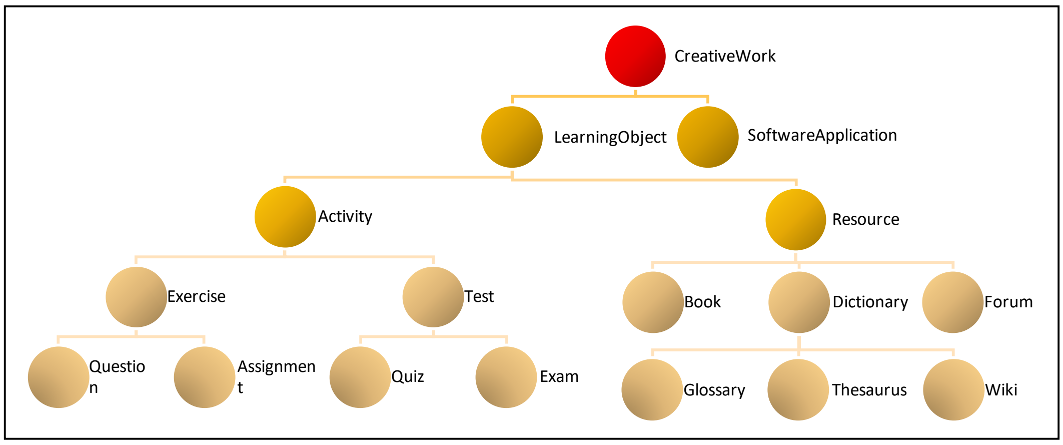


Figure 10. Hierarchy of CreativeWork

In software engineering is commonly known that, (i) **:CreativeWork** is the most generic kind of creative work and includes software programs. In the ontology defined in UML, there is a taxonomy and a superclass by this name (see more details on Appendix A). This class is subdivided in **:LearningObject** and **:SoftwareApplication** classes. On one hand, **:LearningObject** is the intended representation of all objects used to support learning [84]. **:Activity** and **:Resource** classes emerge from this concept and their division is based on the level of interaction that a user can actively participate with them. **Activity** taxonomy subdivides in **:Exercise** and **:Test**, where **:Assignment** and **:Question** classes, and **:Exam** and **:Quiz** definitions derive from, respectively. Further, **Resources** taxonomy subdivides in **:Book**, **:Dictionary** and **:Forum** classes. The purpose of **:Dictionary** concept is to define publications that defines terms, it is subdivided in **:Glossary**, **:Thesaurus** and **:Wiki**. On the other hand, **:SoftwareApplication** is the class that encodes the information of the software that supports the learning process. It is subdivided into the **:WebApplication** class[41]; and this in turn, contains **:VirtualLearningEnvironment** class that keeps the information of platform tool that contains previous learning objects mentioned. More level of granularity is not showed in Figure 10, notice that **:LearningObject** hierarchy is expanded down to three levels as maximum due to space reasons. Analogously, the formal definition of

[41] Specifications and taxonomies are taken as reference from https://schema.org/WebApplication

CreativeWork hierarchy, is presented in turtle syntax (see Listing X2) as an example[42] of the translated ontology in the real XML output file:

```
:CreativeWork  rdfs:subClassOf :Thing.
        :LearningObject rdfs:subClassOf :CreativeWork.
        :SoftwareApplication  rdfs:subClassOf :CreativeWork.
                :Activity rdfs:subClassOf :LearningObject.
                :Resource rdfs:subClassOf :LearningObject.
                        :Exercise rdfs:subClassOf :Activity.
                        :Test rdfs:subClassOf :Activity.
                                :Assignment rdfs:subClassOf :Exercise.
                                :Question rdfs:subClassOf :Exercise.
                                :Exam rdfs:subClassOf :Test.
                                :Quiz rdfs:subClassOf :Test.
                        :Book rdfs:subClassOf :Resources.
                        :Dictionary rdfs:subClassOf :Resources.
                        :Forum rdfs:subClassOf :Resources.
                                :Glossary rdfs:subClassOf :Dictionary.
                                :Thesaurus rdfs:subClassOf :Dictionary.
                                :Wiki rdfs:subClassOf :Dictionary.
                : WebApplication rdfs:subClassOf : SoftwareApplication.
                :VirtualLearningEnvironment rdfs:subClassOf :WebApplication.
```

Listing 2. CreativeWork taxonomy axioms

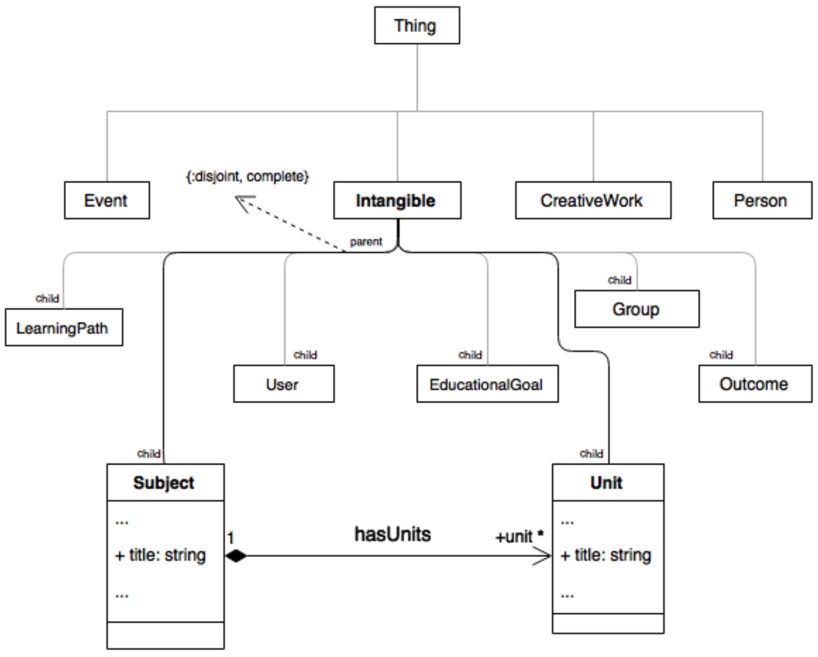


Figure 11. A subview in UML of Subject and Unit classes

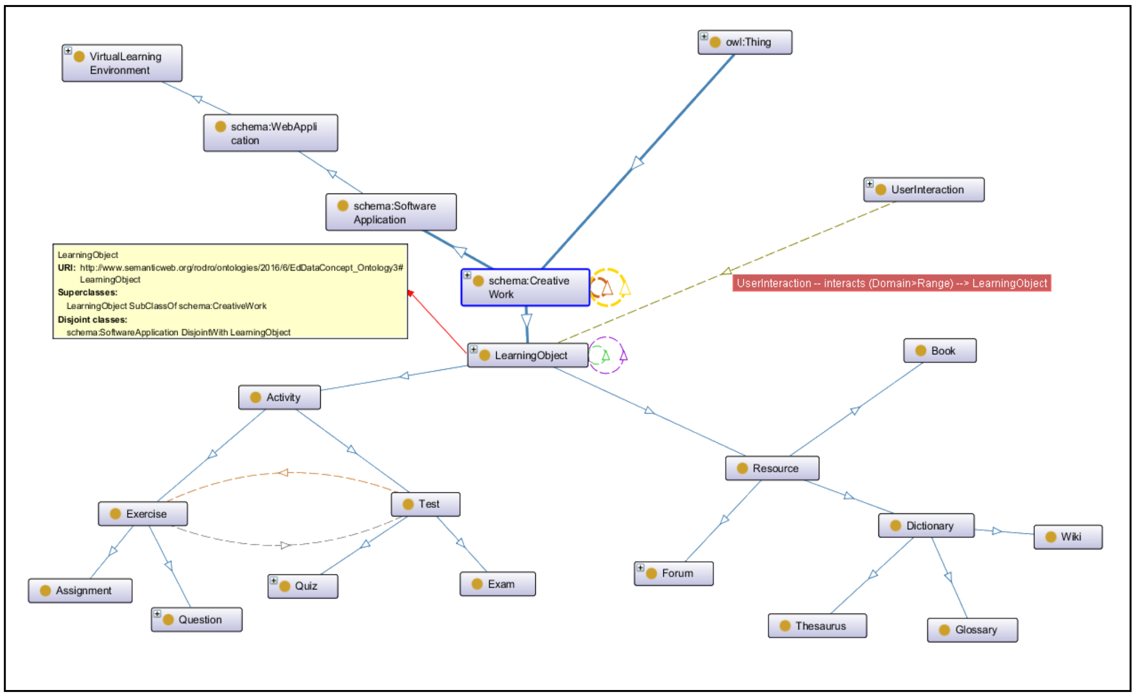
[42] This notation is used for explanation purposes only. Real format file of the ontology in OWL 2 QL, is serialized in a RDF/XML format and can be found on.

Furthermore, as another example to illustrate that disjointness and associations between classes of the current ontology (see Figure 11), can also be represented into axioms; concepts **:Subject** and **:Unit** are being considered. These classes are part of the Intangible taxonomy, and are mutually excluded between them (i.e., instances created of one class cannot be in the other). The **:hasUnits** relation links these classes in domain and range axioms, with the respective inverse counterpart. Next axioms states that the attribute **:title** is shared, so they are represented with the domain **:Subject** and **:Unit** classes. Lastly, the range axiom characterises **:title** to the **:String** datatype. Notice that in next snippet (see Listing X3) these properties are translated, literally.

```
:Subject owl:disjointWith :Unit.
:hasUnits rdfs:domain :Subject.
:hasUnits rdfs:range :Unit.
:isUnitOf rdfs:domain :Unit.
:isUnitOf rdfs:range :Subject.
:title rdfs:domain :Subject.
:title rdfs:domain :Unit.
:title rdfs:range :String.
…
```

Listing 3. Example of disjointness and association axioms

It is worth to mention at this point, that the same translation procedure and axiom definitions are applied to the other two relevant taxonomies mentioned (i.e., User and Event) and to the rest of classes on the LA4S ontology, in order to completely transform from the UML definition into OWL 2 QL. The final representation of **CreativeWork** taxonomy is shown in Figure 12.


LearningObject
URI: http://www.semanticweb.org/rodro/ontologies/2016/6/EdDataConcept_Ontology3#LearningObject
Superclasses:
LearningObject SubClassOf schema:CreativeWork
Disjoint classes:
schema:SoftwareApplication DisjointWith LearningObject

UserInteraction -- interacts (Domain>Range) --> LearningObject

Figure 12. CreativeWork taxonomy in OWL 2 QL

Key: Blue arrow lines represent "Is-a" relations. Expanded information of LearningObject class displays the arbitrary use of a customizable URI, the superclass of the concept (CreativeWork), and the disjointness relations with other classes (in this case with SoftwareApplication) inside the yellow box. Following below, LearningObject class relates to UserInteraction trough the interacts object property (depicted on the red box that represents the axiom in OWL 2 QL). There are also reflexive properties cites and hasNext that are represented by the yellow and red dotted circle, for CreativeWork and LearningObject respectively. Finally, Exercise relates to Test in terms of has Exercises object property. The model is exhibited in a Spring Diagram from the OntoGraf plugin in Protégé framework (ii): User is defined as the entity that represents a person registered in an information system and that may use a service from it. This class subdivides in **:Administrator**, **:ContentDeveloper**, **:Instructor**, **:Learner** and **:Manager** concepts. **:SystemAdministrator** is a specialization of **:Administrator**, **:Instructor** subclass is **:Lecturer**, **:Learner** subclass is **:Student** and **:Manager** class subdivides in **:Area Manager** and in **:CourseCoordinator**. A visual representation of **User** taxonomy is shown in Figure 14.

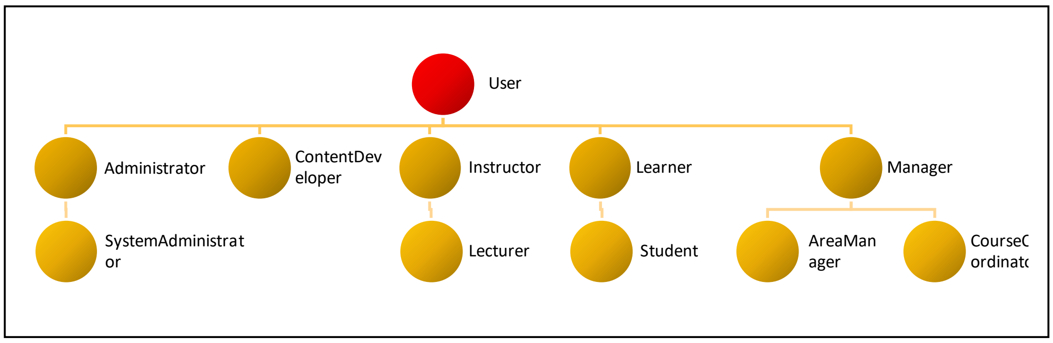


Figure 13. Hierarchy of User

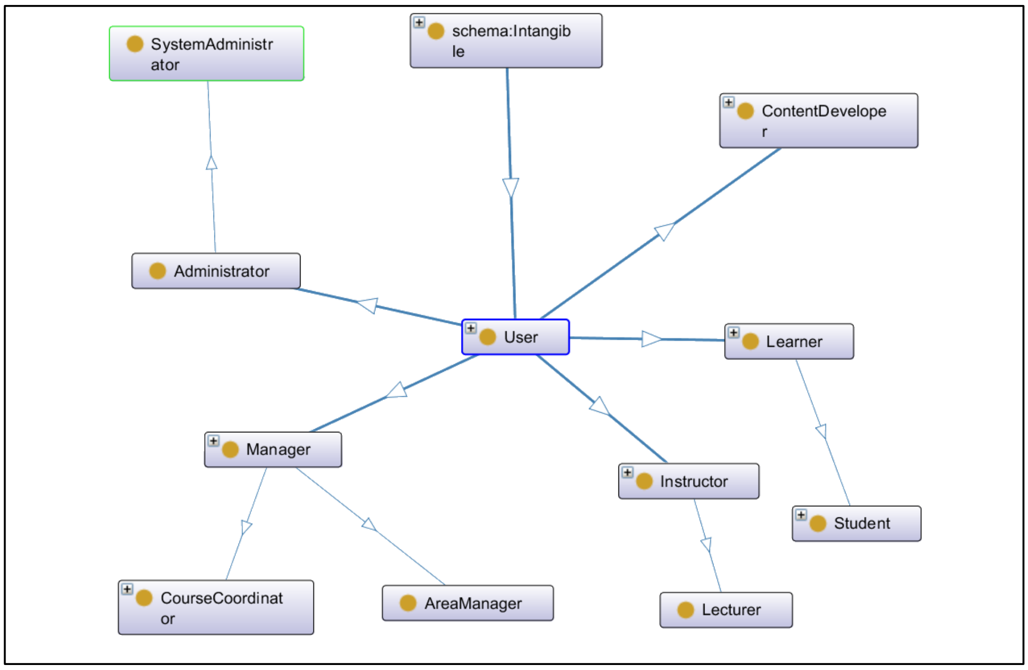


Figure 14. User taxonomy in OWL 2 QL

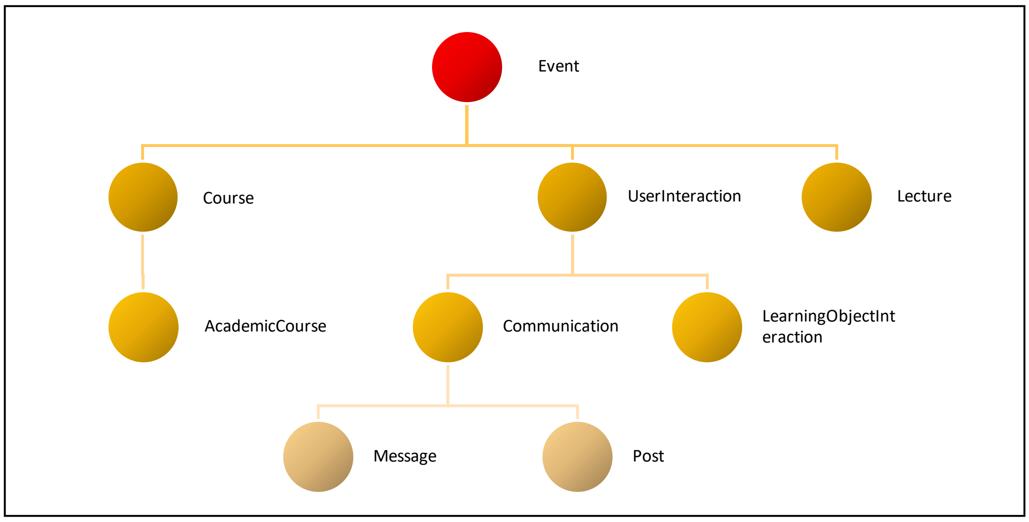


Figure 15. Hierarchy of Event

(iii) **:Event** is the concept that involves the happening of an action in a certain period of time or location[43]. In this ontology the class is subdivided in **:Course**, **:Lecture** and **:UserInteraction** sub-classes. **:AcademicCourse** is a specialization of **:Course**, **:UserInteraction** is meant to be the core interactivity centre of the LA4S Ontology, as in the UML definition is considered an association class that relates with **LearningObject** concept. This class subdivides in **:Communication** and **:LearningObjectInteraction**. **:Communication** class has two sub-types: :**Message** and **:Post.** Final representation of **Event** taxonomy is shown in Figure 16.

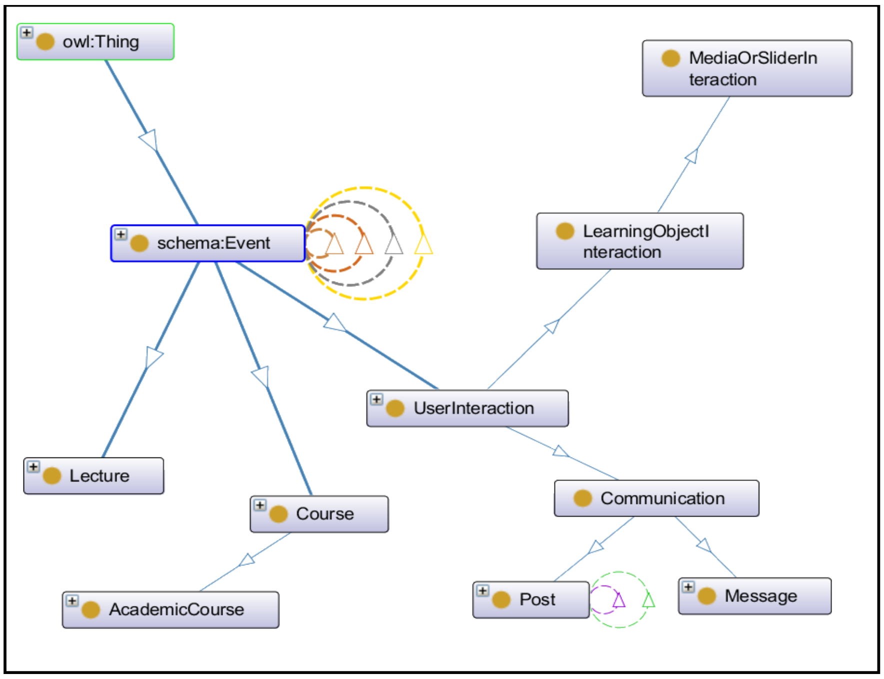


Figure 16. Event taxonomy in OWL 2 QL

[43] Adapted from: https://schema.org/Event

5.2 Mappings engineering

In this section a definition of what is a declarative mapping (in the context viewpoint) and the importance of applying it under a OBDA method for the LA4S ontology is provided. Limitations and a conceptual example of the underlying mechanisms used over the given dataset are mentioned in detail.

A declarative mapping is a serialized description of the SQL query over the database, and the construction template of the RDF triple from the TBOX delineation in the ontology[44]. The power of retrieving correct information lies in the level of accuracy defined in the source part of the mapping (see Chapter 4).

In order to connect previous translated ontology to the dumped database, the process of defining accurate mappings must be done for a feasible OBDA model implementation under the use of OntopPro plugin. These mappings encode database enquiries into classes (concepts), object (roles) and data (attributes) properties predefined (as in previous section). The use of the vocabulary of a specialized ambit derives from the ontology's definition, making information queries more accessible for users and the possibility to enrich raw data to hidden and valuable information from large datasets.

As it was detailed on Chapter 4, -Ontop- will be used to implement query answering over RDF virtual graphs under the OBDA approach. The type of mapping's definition is denominated query-driven as in [85], and does not affect to data tables values as there is no writing operation behind it. Queries performed over RDF graphs are translated automatically in their SQL analogous, this function is done over the ABOX virtual mode of the tool (see Chapter 4 for technical details).

5.2.1 Technical considerations

There are some initial conditions for performing mappings development in-Ontop-regarding LA4S ontology summarized as [45]:

MySQL configuration: For querying construction, configuration option ANSI_QUOTES should be enabled for proper working. Also, the use of quotes in MySQL for identifiers is mandatory as the tool has issues with case sensitivity and registered keywords.

In the local schema instance of Agora, a configuration over the options file of the database is necessary, to automatically set the database initialization for ANSI_QUOTES allowed for SQL querying.

Mapping issues in RDF: Blank nodes are not allowed in the engine parsing and/or in the mapping language OBDA or RDB2RML. Datatypes defined in the domain ontology are not supported for RDF triples (e.g., range, powertypes, etc.).

For avoiding blank nodes in triples, null values in columns that are not primary keys must be treated. In this case the function IFNULL() can be inserted in the query sources of the mappings for a better control.

[44] Adapted from https://github.com/ontop/ontop/wiki/ObdalibQuestMappinglanguage#Mappings_in_plain_english
[45] Those consideration are grouped in resemblance of https://github.com/ontop/ontop/wiki/ObdalibIssues#Databases_and_SQL

SQL syntax rules
Two columns that share the same name are treated as equal. The use of not repeated aliases for columns in tables, should be explicitly defined in the target component of the mapping. The reason behind it, is that the tool recognizes the individuality of the symbols in the ontology and variables of the view created by SQL queries, especially if a mapping is reusing the same table as a common source.

A complete application of the required syntax for mappings definition ensures that the tool works neat over the data sources and performs SPARQL queries without shortcomings or setbacks.

Implicit Integrity Constraints
The definition of primary and foreign keys, and indexes in the schema database is essential for -Ontop-, especially in the use of JOINS. This natural optimization implication in a database can represent up to 100 times of improvement in any query answering.

Integrity constraints on were partially rebuild from zero. Since the dataset is an incomplete dump of the original Agora schema, not all desirable optimization or either the use of the vocabulary mapping of the ontology can be ensured. Further, many assumptions over the data exposed are being contemplated.

Limitations in SQL functions and operations
The following functions / operators are strictly not supported in mapping declarations: CASTING, ORDER BY, EXISTS, UNIQUE, TOP[46]. The use of them will be rejected by the parser causing errors in the interface.

None of the source queries for the construction of the mapping are using any of the previous operators.

There are also some operators / functions that will affect to the optimization of the query answering like: MIN/MAX, CASE - WHEN clause, DATE FUNCTIONS, NESTED SELECTS, RIGHT/FULL/SELF/CARTESIAN JOIN, SUBJOIN, ALL, ANY, UNION, MINUM and EXCEPT[47]. They are converted in to sub-views that can impact to the performance of the responses in the database engine.

Some definition of the queries includes the use of some of these functions, for specific value retrieval to populate classes, object or data properties. Such reason is due to a database limitation in which relations are not explicitly stored in a table, but in a SQL query produces a view.

Also, the use of DISTINCT translate to using sub-views by the tool in the rewriting technique. For the LA4S database schema is mandatory the use of this operator as far as many individuals defined in the ontology, are not unique in tables in some cases (as mentioned in Chapter 3). The use of a discriminator like distinct function, allows to retrieve unique identifiers for materializing the IRI instances of the RDF graphs.

5.2.2 Mapping Examples on LA4S OWL 2 QL

The main principle of mappings in -Ontop- is the instantiation of values of the variable used in the SPO target template. In this template those variables are identifiers that refer to a specific column in a relational table. For instance, if the concept :**CreativeWork** is

[46] Issues information taken from https://github.com/ontop/ontop/wiki/ObdalibIssues
[47] Known problems reported from https://github.com/ontop/ontop/wiki/ObdalibIssues

declared in the URI template of the target in the form “:creativework/{id}/”, then every tuple answered from the source query will be instantiated inside the enclosed term “{id}”. Giving as a result, e.g., assuming that the evaluation is just one tuple; the URI form like “:creativework/{1}/” .

Populating classes

Next Listing 4 shows an example of how to populate the **:Forum** class, a subclass of the **CreativeWork** hierarchy using the aforementioned instrument.

```
mappingId       ForumCMapp
target          :creativework/{id}/ a :Forum .
source          SELECT DISTINCT cm."id"AS"id"
                FROM "mdl_course_modules" cm
                JOIN "mdl_modules" m ON cm."module" = m."id"
                WHERE m."name" = 'forum'
```

Listing 4. Forum Class Mapping script

Notice, the use of “a” in the target template of the mapping, this is the equivalent form of **rdf:type** in previous syntax version of the tool. In this mapping a relative complex query is performed over two tables in order to retrieve just the tuples that represent the forum objects of the Agora database. The use of “distinct” operator in this case is to ensure uniqueness of the individuals, due the inexistency of a direct table that keeps the whole information of the class Forum[48].

Populating data properties

To map data properties from columns of the database, it is imperative to have in mind that in the target template, the class variable identifier has to be included. The reason is to relate this identifier with the variable that represents the value of the column of the pointed table. Listing 5 shows either the sample where the class is involved in the mapping or without it.

```
mappingId       UnitCMapp
target          :unit/{idUnit}/ a :Unit ; :section {unitNumber} .
source          SELECT "id" AS "idUnit",  "section" AS "unitNumber"
                FROM "mdl_course_sections"
```

```
mappingId       SectionDMapp
target          :unit/{idUnit}/ :section {unitNumber} .
source          SELECT "id" as "idUnit",  "section" AS "unitNumber"
                FROM "mdl_course_sections"
```

Listing 5. Valid mapping scripts of :section data property

[48] This criterion is assumed in most of the mapping development as the database dump did not include all the desirable structure of information from a complete Moddle LMS dataset.

It is worth of noticing that in this case, the **:Unit** class is being related to its attribute **:section** for populate it.

Populating object properties

An example of how to create relations of the object properties in the ontology trough mappings is showed in Listings 6, where class **:Group** relates trough **:isFormedBy** object property with **:Learner** class.

```
mappingId           isFormedByOMapp
target              :group/{sourcedId}/ :isFormedBy :user/{userid} .
source              SELECT DISTINCT g."id" AS "sourcedId", ra."userid" AS
                    "userid"
                    FROM "mdl_role_assignments" ra
                    JOIN "mdl_groups_members" gm ON gm."userid" =
                    ra."userid"
                    JOIN "mdl_groups" g ON gm."groupid"=g."id" WHERE
                    ra."roleid" IN ('5')
```

Listing 6. Mapping of :isFormedBy object property

A crucial aspect to take into consideration is the use of the same URI template format defined as previous mappings, in order to avoid new individuals' introduction to the current model. The use of the Join operation in the source query is coherent because three tables are using the same two foreign keys, that is implicitly keeping the relations between the two classes being mapped.

5.2.3 Reification implementation in an Associative Class

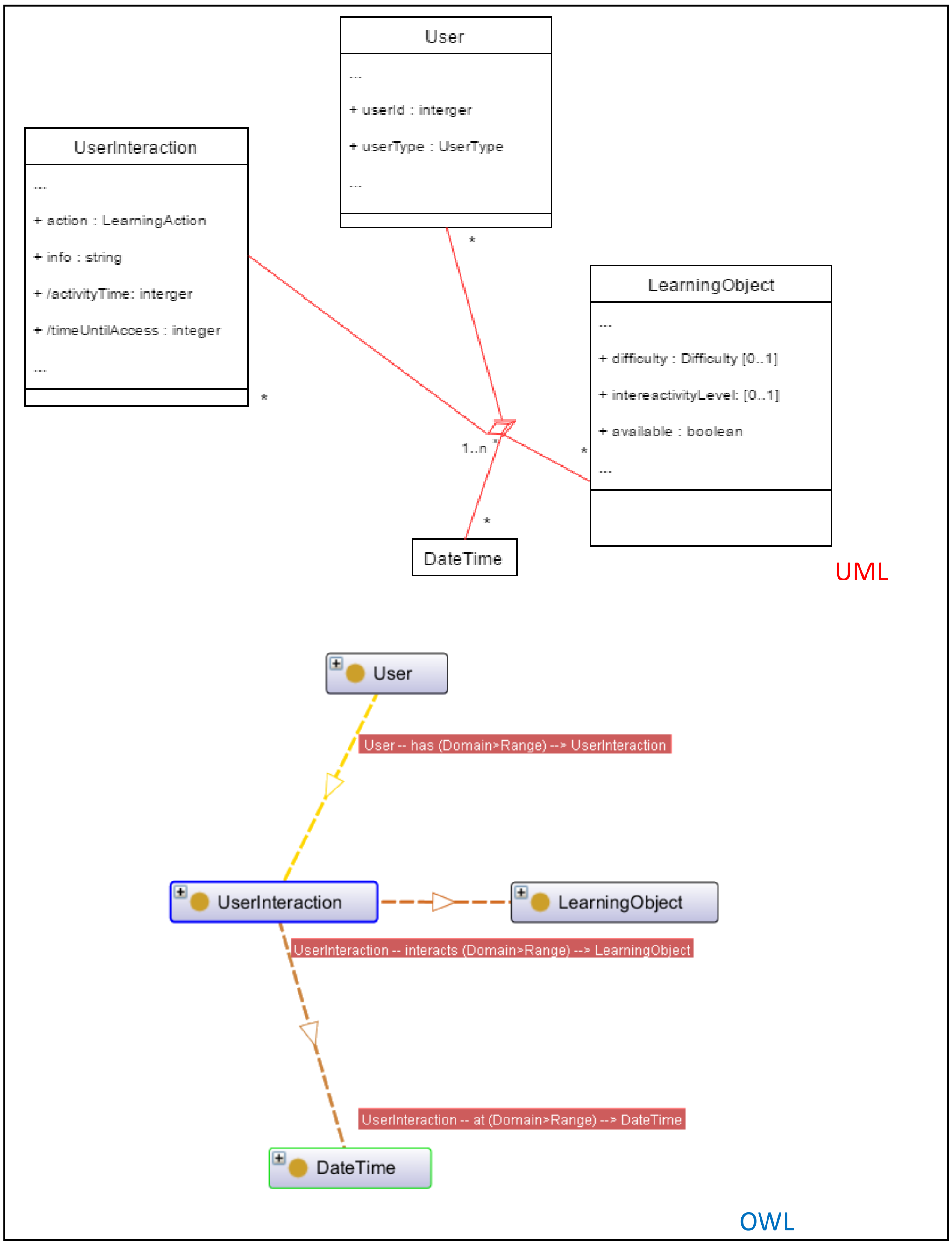

User
...
+ userId : interger
+ userType : UserType
...
UserInteraction
...
+ action : LearningAction
+ info : string
+ /activityTime: interger
+ /timeUntilAccess : integer
...
LearningObject
...
+ difficulty : Difficulty [0..1]
+ intereactivityLevel: [0..1]
+ available : boolean
...
DateTime
1..n
UML
User
User -- has (Domain>Range) --> UserInteraction
UserInteraction
LearningObject
UserInteraction -- interacts (Domain>Range) --> LearningObject
UserInteraction -- at (Domain>Range) --> DateTime
DateTime
OWL

Figure 17. Reification applied in an associative class/role

When dealing with binary relations, all previous methods are suitable to populate and interconnect concepts and roles, the exception comes when a relation involves more than two entities. This kind of associations are called n–ary relations and are very common in modelling paradigms like UML for constructing a natural behaviour of objects in systems. OWL/RDF cannot handle n-ary relations directly, so in order to reconstruct the logic beneath the associative class **UserInteraction**, a reification method in the translated ontology and in the mapping component of the model must be stablished.

The ground idea of reification is to manage a n-ary tuple as an entity and connect it to each member of the tuple with a binary link. It is commonly known that there are two representation patterns to apply this technique. The one used in this study is by introducing a new class that will represent the class itself and also the association[49]. This change in the OWL 2 QL ontology is reflected when the UML "UserInteraction" class and the n-ary relation "interacts" transforms into the **:UserInteraction** class with three new object properties that connects the arity sense with the other classes needed (i.e., **:User**, **:LearningObject** and **:Datetime**). New object properties are defined like natural words for the vocabulary of the ontology:

:has (User, UserInteraction)	Which models a binary tuple between user and user interaction concepts
:interacts (UserInteraction, LearningObject)	Which models a binary tuple between user interaction and learning object concepts
:at (UserInteraction, Datetime)	Which models a binary tuple between user interaction and date time concepts.

In Figure 17 it is shown graphically, the reification applied on the associative class in UML. Notice that in the UML portion of the diagram, **UserInteraction** is being related to other classes with more than two relation arity. The difference with respect of the OWL 2 diagram, is that is not being treated as an associative kind. The multiple arity relation and the class is represented as a single OWL class with three new object properties linking other classes. By keeping the binary logic for each instantiated tuple, data consistency is assured between all relations. An RDF/XML output file may contain the translation syntax similar to:

```
:UserInteraction rdfs:subClassOf :Event.
:User rdfs:subClassOf :Intangible.
:LearningObject rdfs:subClassOf :CreativeWork.
:DateTime rdfs:subClassOf :Thing.
:has rdfs:domain :User.
:has rdfs:range :UserInteraction.
:interacts rdfs:domain :UserInteraction.
:interacts rdfs:range :LearningObject.
:at rdfs:domain :UserInteraction.
:at rdfs:range :DateTime.
```

Listing 7. OWL 2 QL axioms after reification on UserInteraction

As mentioned before, the same associative class represents the n-ary associations with the other classes. It would be redundant to create a new entity in the OWL 2 QL definition and, by consequence an extra mapping declaration in the native mapping of the tool.

With regard to the mappings definition, the structure is modified for populating classes and roles. Entity derivations resultant after applying the reification technique in the ontology, relies in the most promising set of values (i.e., table) that keep the record of various components that determine the value of another's (i.e., columns). This leads to choose those tables that contains compound keys or at least, tables that can handle the information of multiple relations from other tables. In this particular case the "log" table

[49] Based on W3C Convention in https://www.w3.org/TR/swbp-n-aryRelations/#representation

of the entire dataset is “logstore”, that keeps record of all the objects, users and interactions in the LMS. It is worth to mention that this table does not have a real compound key but, for implementation purposes, it is being considered the creation of indexes over the columns that are related to the compound key.

When implementing reification inside the structure of the mappings, a customized definition over the template target and source fields must be declared like the one presented above[50]:

```
mappingId       UserInteractionCMapp
target          :event/{userid}/{id}/{datetimeId}/ a
                :UserInteraction .
source          SELECT lo."userid" AS "userid", lo."objectid" AS
                "id", lo."timecreated" AS datetimeId",...(complex
                projection)
                FROM "mdl_logstore_standard_log" lo

mappingId       hasOMapp
target          :userid/{userid}/ :has
                :event/{userid}/{id}/{datetimeId}/ .
source          SELECT DISTINCT lo."userid" AS "userid",
                lo."objectid" AS "id", lo."timecreated" AS
                "datetimeId" FROM "mdl_logstore_standard_log" lo
                JOIN "mdl_role_assignments" ra ON
                ra."userid"=lo."userid"

mappingId       interactsOMapp
target          :event/{userid}/{id}/{datetimeId}/ :interacts
                :creativework/{id}/ .
source          SELECT DISTINCT lo."userid" AS "userid",
                lo."objectid" AS "id", lo."timecreated" AS
                "datetimeId" FROM "mdl_course_modules" cm JOIN
                "mdl_logstore_standard_log" lo ON
                lo."objectid"=cm."module"

mappingId       atOMapp
target          :event/{userid}/{id}/{datetimeId}/ :at
                :datetime/{datetimeId}/ .
source          SELECT DISTINCT lo."userid" AS "userid",
                lo."objectid" AS "id",
                lo."timecreated" AS "datetimeId"
                FROM "mdl_logstore_standard_log" lo
```

Listing 8. Class and object properties mappings after reification

Notice that the structure of the target template for URI construction in the first mapping declaration, is joining three different identifiers for stablishing the “**UserInteraction**” concept of the ontology (i.e., :event/{userid}/{id}/{datetimeId}/ that are delimited by brackets in red lined boxes). In the source declaration the mapping occurs with the columns selected from the compound key of the “logstore” table (i.e., lo.”userid”, lo.”objected” and lo.”timecreated”).

[50] This is just a specific example for better comprehension, the real implementation includes more property mappings inside the obda file

Next mapping declaration connects the URI previously defined, with the correspondent component of the n-ary relation with regard of the **:has** object property. Thus, it preserves uniqueness at the relations for potential graphs instantiations. The other object properties **:interacts** and **:at** behave mappings have a similar rationale.

An example of a retrieved row will generate the following example graph:

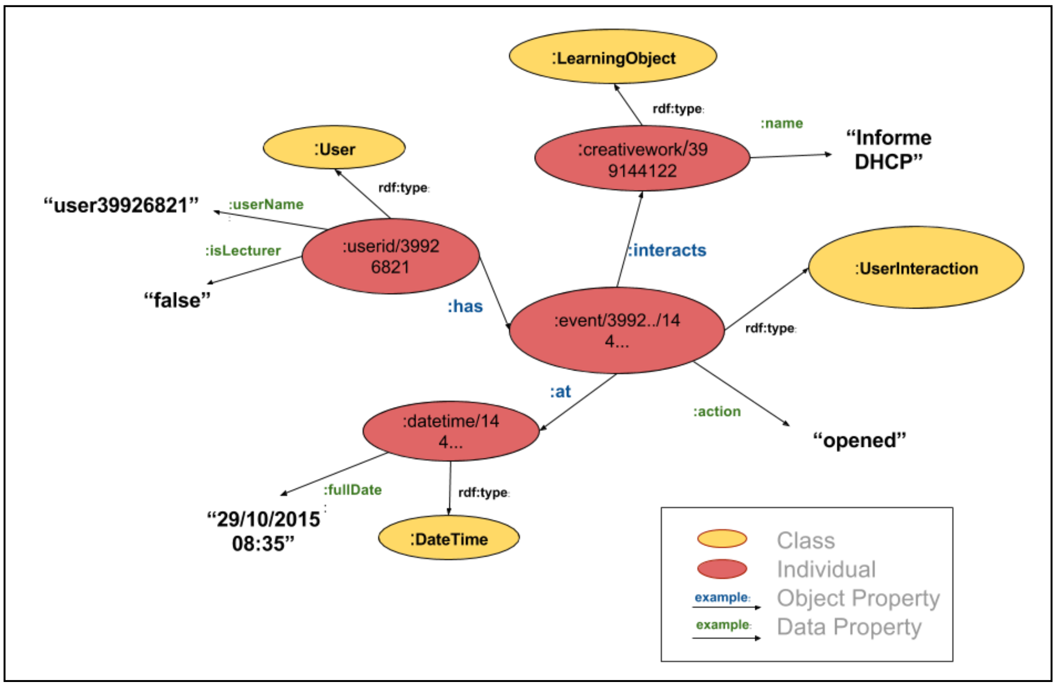


Figure 18. A graph example from a retrieved row

Applying this reification method in the ontology and in the mappings portion of the OBDA model, does not just ensure its functionality but also keeps the binary relation principle between classes which is the basis of reification for OWL 2 QL.

5.3 Ontop shortcomings

In order to map axioms for the ontology it is required the use of complex queries for complete information retrieval. This means using more than one select-project-join query types. Likewise, some unavoidable inclusion of union operators in the creation of logic tables cause a bulky source query declaration, that may overflow the JSQL parser used by the engine. This is a problem if several attributes from the same classes in different levels must be specified to populate a hierarchy.

As a workaround solution to deal with this problem, developers advice is not to use massive number of unions in queries or to break them down in several mappings. Another solution is to raise up attributes from the lower levels to their father classes in the ontology, in this way consistent information retrieval for semantic querying is being kept.

Another issue is to deal with attribute names, either in in database as columns or in the mapping declaration as an alias. No workaround is found for this issue.

Appendix B shows the final conceptual schema in a Protégé basic image. The detail of the output file in OWL of the translated ontology can be found at the following link: http://bit.ly/2dmT0yb. In addition, Appendix C contains the .obda script which explicitly shows 29 mappings developed for testing the implementation of the OBDA model.

CHAPTER 6

6 OBDA Model Validation

In this chapter a brief description of relevant queries, initial premises for evaluation, the experimental method and results are presented.

6.1 Relevant queries and premises

The criteria to choose the right enquiries in a LMS is assumed due the analysed behaviour of the data, and the research performed with Moodle v2.8 official documentation found on the web[51]. This allows the development of all conceivable mappings for the ontology with respect of the limited dataset.

Particularly those refer to user 's actions over some learning objects in a course. The most suitable case for testing the ontology's vocabulary in the educational context, involves the class :**UserInteraction.** This can be considered the centre of interoperability regarding the conceptual model.

For a first example, suppose a user is interested in retrieving all the group names of the students. To be expressed in SPARQL syntax, this query may take the following form:

```
SELECT DISTINCT ?Group ?groupName ?Learner WHERE {
        ?Group a :Group.
        ?Group :name ?groupName.
        ?Group :isFormedBy ?Learner.
        ?Learner a :User
}
```

Listing 9. SPARQL Query for retrieving Groups and Learners

As it can be observed, the query declares a relative natural language expression; in fact, the vocabulary provided by the ontology is being used. The expected outcome includes the ids of the **:Group** and **:User** classes, as part of the construction of the URI template in the mappings. Additionally, the **:name** property of the group is part of the enquiry in the statement.

The relation **:isFormedBy** links both classes in the condition clause, this means that the mapping of this relation is expected to work, regardless being part of the selection. For an end-user perspective, this underlying process is transparent and allows an intuitive way to formulate a broad set of queries including combinations of them.

Consider another query where a consumer needs to retrieve all user interactions over a specific type of learning object. In SPARQL, it can be defined like:

[51] Further information can be found in https://docs.moodle.org/dev/Event_2#Properties

```
SELECT DISTINCT ?User ?Assignment WHERE{
        ?User a :User.
        ?User :has ?x.
        ?x a :UserInteraction.
        ?x :interacts ?Assignment.
        ?Assignment a :Assignment
}
```

Listing 10. SPARQL Query for retrieving all activity objects of users

Where the learning object of type “Assignment”, is being filtered in the query at the end of the statement, passing through the relations of the classes involved. A variable definition in a SPARQL set of triples is identified by the use of the question mark character “?”. In the selection part of the query there are two variables that represent the classes required. Then, the set of conditions are declared after the WHERE clause (inside the brackets) explicitly expressing the navigability performed by the relations already mapped. In other words, the “User” class with the relation “has” over “UserInteraction”, which is related trough “interacts” over the “Assignment” class; are expected to return all the possible individuals defined by this multiple arity relation. Furthermore, this query is able to test the entanglement power provided by the model implemented between the three classes in the ontology; **:User, :UserInteraction** and **:LearningObject**, which is the ancestor of the **: Assignment** in the taxonomy with the same name (LearningObject).

Notice that the variable declaration is expected to return the identifier number of the user queried in both examples. For testing purposes this is sufficient due the limited information in the data source (no specific information is expected to be retrieved from users, due privacy matters as previously mentioned on Chapter 3).

For testing the reification technique of the associative class, the following query is declared:

```
SELECT DISTINCT ?User ?action ?Exercise ?fullDate WHERE{
        ?User a :User.
        ?User :has ?x.
        ?x a :UserInteraction.
        ?x :action ?action.
        ?x :interacts ? Exercise.
        ? Exercise a :Exercise.
        ?x :at ?date.
        ?date a :DateTime.
        ?date :fullDate ?fullDate
}
```

Listing 11. SPARQL Query of user and dates of events over Forum objects

The logic behind this query is to retrieve users that have performed a pre-defined action over a learning object of type Exercise, with the date associated to the event. Navigability in this query, is expected to be performed between the three classes that conforms part of the reification applied. Moreover, three data properties are included in order to test more information retrieval from the mappings: “?action” and “?fullDate” which are attributes of the concepts **:UserInteraction** and **:LearningObject** respectively (represented by the **:Exercise** class, a subtype of a learning object).

Some other important aspects to consider, are the validation of disjointness of classes and inverse properties in the model. These type of axioms are part of the set of constructs allowed in OWL 2 QL profile that the engine in Ontop parse them before query rewriting.

To test disjointness between concepts, a query formulation over any class of the taxonomies in the ontology can be adapted in the following form:

```
SELECT DISTINCT * WHERE{
        ?x a :Instructor.
        ?x :userName "user151824"
}
```

Listing 12. SPARQL Query for testing disjointness between classes

The result of this query will be empty for the reason that beforehand, it is known that the individual does not belong to the class: Instructor. This individual is of type **:Learner**, thus disjointness consistency of the model is reflected trough the axiomatic power of the ontology, that is being considered by the inference tool and does not allow to include it inside the answer expected.

To test inverse properties, that were part of the criteria for translating associations from UML to OWL language; next two following queries express both directions of one single relation (**:hasUnits**):

```
SELECT DISTINCT ?titleSubject ?titleUnit  WHERE {
        ?x a :Subject.
        ?x :title ?titleSubject.
        ?x :hasUnits ?y.
        ?y a :Unit.
        ?y :title ?titleUnit
}
```

```
SELECT DISTINCT  ?titleUnit ?titleSubject WHERE {
        ?y a :Unit.
        ?y :title ?titleUnit
        ?y :isUnitOf ?x.
        ?x a :Subject.
        ?x :title ?titleSubject.

}
```

Listing 13. SPARQL query samples of an inverse property

Where a subject is composed of units, hence a unit belongs to a subject. Two data properties of the classes **:Subject** and **:Unit** conforms the selection chunk of the SPARQL query, **:titleSubject** and **:titleUnit**.

Previous definition of SPARQL queries and the validation of disjointness among classes and inverse properties over roles, will confirm that the OBDA setting stablished is consistent unrelatedly to the restraints of the data provided.

Domain and Range axioms represent the sense of the predicate, represented by the object or data property in the triples. With the tool supporting the model implementation, is expected that these axioms are also considered in the inference parsing process, when answering relevant enquiries. A particular experiment setting for this matter, will be described onwards.

To evidence the OBDA model implementation from the ontology translated and the data instance analysed, the following catalogue of queries are described:

q_1:	"All different users interactions "
q_2:	"All different interacted Assignments"
q_3:	"Names and URI's of the User hierarchy, including child classes: Instructors, Learners and Content Developers
q_4:	"All different tests interacted by an User"
q_5:	"Title of Units contained in All Subjects and their relations"
q_6:	"All distinct dates of the events"
q_7:	"Units and Exercises objects of interactivity type 'Mixed'"
q_8:	"Groups and name groups that contains learners"
q_9:	"Subject titles containing Units"
q_{10}:	"Relations of user interactions and their actions with a book type of object"
q_{11}:	Reification validation: "All exercise interactions performed by a user with the specification of the date"
q_{12}:	"Forum interactions performed by a user in a specific date"

Table 4. OBDA Validation Queries catalogue.

6.2 Evaluation method

The execution of previous queries was initially thought for the entire set of data. This set represents all the information about learning object interactions of Agora system from over 615 users. Natural restrictions from the use of a local machine hardware, combined with a database structural weakness in terms of basic referential constraints and the absence of reliable information from tables; does not allow an approximate validation to the real advantages of the tool. These advantages are specified by the tool developers announcing a strong basis on the description logics DL – LITE_R.

Despite this, the sample of the population in the dataset is reduced to 85 users, considering all the possible events and the information that derives from the mapping definitions. Arbitrary indexes in the tables were implemented for the columns with more level of inquiring. The reason is, as mentioned before on Chapter 3, most of the primary and foreign constraints (implicit database constraints) cannot be rebuild with integrity due the lack of information reliability (i.e., not all values of ids from a reference table were part of the values of another column).

Then, necessary reduction adjustments over the catalogue of queries are done. This reduction is basically an "overhauling" of the SQL declaration in the source mapping definition over the elements that are linked to the User class. The filtering for the reduced sample are precisely effected upon mappings of **:Event** and **:UserInteraction** classes; and on to **:has**, **:interacts** and **:at** object properties. Notice that by contrast, the actual **:User** class is not being affected, this is done by virtue of the query requirement on how many real users exists on the dataset.

The run of query executions is tested over this hardware specifications:

Intel Core i7-4710HQ CPU @ 2.5 GHz
X64 Architecture
12 RAM memory
SATA HDD velocity of 5400 rpm

6.3 Results

To illustrate the execution times of the catalogue of queries, Figure 19 reports the time in seconds in a bar comparison chart:

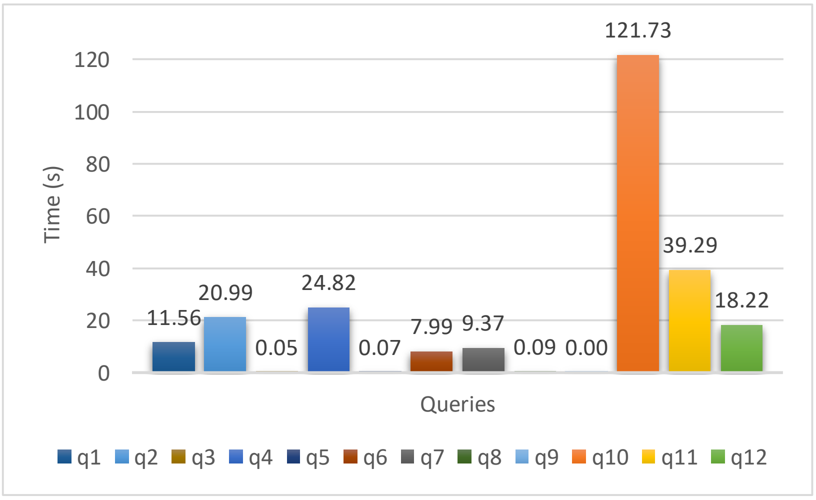


Figure 19. Query Run Execution Times

Notice that the subset of queries q_1, q_2, q_4, q_6, q_7, q_{10}, q_{11} and q_{12}, run behind 130s (seconds). The explanation of their relative low performance can be explained by two circumstances.

The first is related to the use of more than two JOIN or UNION operations in the mappings rationale. For instance, in order to get the information of the Creative Works taxonomy, most of the SQL queries that are being used to construct the levels and singular classes; employs up to 3 JOIN operations and an average of 5 UNION declarations for retrieving the correct ids of the learning objects involved in the LMS. This cannot be avoided since it is needed to maintain the classes that contains the objects that are not being defined in the ontology, but that appears in the dataset.

Secondly, the navigability within the ontology lies in the correct definition of the relations between concepts of the model, meaning the use of several JOINS when the engine rewrites the query in SQL means, affecting directly to the performance of prompted responses. Notice this behaviour specially in time execution of the 10th query (q_{10}). Internally, join operations are again executed in the table “logstore” (with the most level of cardinality) to add the new data property called **:action** as part of the vocabulary in the SPARQL query and as a new retrieved variable in the outcome of the result.

In order to raise the level of query optimization an advice from the wiki page of the developers[52] is taken. This advice state not to define domain and range properties unless good foreign keys are implemented in the dataset. Domain and range axioms from the most valuable relation (i.e., “**:interacts**” object property) is not included in the OWL file, for the execution of the query run in a second test.

The next Figure 20, expose marginal improvement in general over the execution time of the queries:

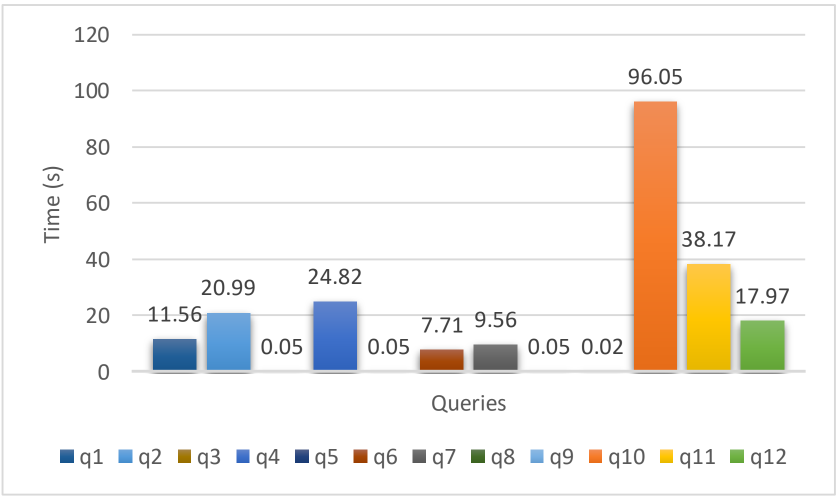


Figure 20. Query Run Times after applying optimization

It is worth of mentioning that just in the case of q_{10} a significant improvement is reflected in around 20s. Other queries like q_{11} and q_{12}, just improve in less than a second. By contrast, the rest reflect no enough variability in ms (miliseconds) and their details are not pertinent.

The other subset of the queries; q_3, q_5, q_8 and q_9 run in less than a second in both experiments, being considered as quite fast. This is explained since in three of them, the use of at most one JOIN operation are declared on their correspondent mapping. Besides, the tables that allows the relations **:hasUnits** and **:isFormedOf** are not

[52] Taken literally from https://github.com/ontop/ontop/wiki/MappingDesignTips

implicitly related under the logic of a regular Moodle database, so no indexes are implemented in the tables that are used for mappings. It is worth of mention that q_5 and q_9 denotes the inverse property testing and their execution is included in this group of fast queries.

The results reveal that the quality level of implementation of the model is directly constrained to the level of referential integrity of the source. This quality is defined by the level of query optimization. The proof is evidenced in the second experiment; when data was used to populate classes and properties of the ontology and the domain and range axiom considered by the tool is a weak premise compared to the SQL query definitions in the mapping.

CHAPTER 7

7 Discussion, Conclusions and Future Work

7.1 Discussion

In general there has been two traditional approaches for designing data access solutions for end-users: (i) The first one address to small scopes which the system data is handable in technological meanings, and approaches such as the creation of data warehouses which allows end users to see ad-hoc queries. Drawbacks of this approach are mainly adjudicated to maintenance costs and reengineering process, regardless the level of robustness from a given architectural design and their query orientation. (ii)The second one is based on the load first, model later paradigm; where the data treatment is performed after extraction or loading processes. In this way, properties such as flexibility and scalability of systems are better accounted[53].

For any of the previous two cases, by providing an abstract representation of a domain to an end-user, heavy loaded data analysis is not anymore a recurrent problem for a technical expert. On the contrary, it is an opportunity for creating an additional feedback channel for data behaviour with regard to any application development.

This early access to datasets from an end user perspective will improve data modelling. By transforming passive participation of any stakeholder to become an active agent in the evolution of a solution, it is being facilitated the engineering work over the information in any type of source. Furthermore, the consolidation of data and its integration evolves to joint oriented efforts of all actors in an organization, avoiding outsourced actions from IT experts. This derives into potential synergy produced by the implicit knowledge created of any domain and supports discretionary decisions.

Additionally, the use of technologies like RDF and OWL over the model definition of any domain, boost a system to become a potential linked data initiative. OBDA or OBDM (Ontology Based Data Management) is a new modelling paradigm in which several challenges had been originated, according to [86] two of them are:

1. To efficiently design a way to translate a data source trough an ontology into a useful conceptual schema.

2. To implement consistent and tolerant query answering methods.

In none of this challenges, an OBDA model implementation is considered from a previous conceptual definition. Findings of the present work have shown a set of steps that must be realized when this situation is latent.

Another contribution from the present work, is that this document abridges the whole knowledge of OBDA background required to find a proposal that re-use traditional relational methods. Carrying out an effective analysis of information based on zooming technologies that enables a sectional view of all layers used in any engineering process.

[53] Adapted from - https://tdwi.org/Articles/2013/10/15/Load-First-Model-Later.aspx?Page=2

7.2 Conclusions and Future Work

In this study it has been reviewed a subset of theory background regarding Semantic Linked Technologies. The main objective was to apply a state of the art data-access paradigm based on domain conceptual modelling to an educational context. The first input was a formal ontology definition in UML. Then, a dual Protégé-Ontop framework has been chosen to transform the ontology manually and to support the model implementation.

In particular, with the result of the correct execution of the inverse properties, it can be assumed that the model allows to use and expand the vocabulary in the aforementioned study for LA4S. By the use of words or "verbs" in the ontology, and their respective inverse form (e.g., "hasUnits" – inverse: "isUnitOf"); it has been proven that any expert domain user, who is familiarized with the ontology concepts can make use of this capability when querying information.

By one hand, the navigability implemented between the three main hierarchies is a strong point in this prototype. Query answering comes from a mixture of several relations between the classes that represent the most important taxonomies. On the other hand, the information provided by the dataset has not been enough to map all taxonomies. Nevertheless, the core interoperability of the model is working.

As mentioned before, just a fragment of the ontology was possible to map and test. If a better scenario may be prompted in the future, more mapping level and testing could be achieved. Previous query evaluation demonstrates that the implementation of OBDA model is feasible even with some adverse conditions (revealed in the study), and the translation of the ontology in OWL 2 standard for querying services is virtually possible.

It is worth to notice that, even with the power of expressivity of OWL 2 QL over disjointness constraints in classes and the entire feature support provided with reasoning techniques of the tool, data consistency was not guaranteed completely. Instead, the model can be used to corroborate data integrity within the database provided.

As the core interoperability of the model has been implemented, this could mean the beginning to map the entire Agora database in a production environment. If this is not possible to achieve, an intermediate effort must include constraints contemplations that have been highlighted in this work, with additional datasets before providing an SPARQL end point for users.

One practical use for the current prototype is to convert it as an interface for data exploration layer. This helps to normalize or enrich current context vocabulary for educational systems in order to improve future ontology designs. The use could be also extrapolated for migration and integration initiatives of different learning systems that coexist together in an organization. These type of LMS are based on relational databases where the technology that has been tested can fit.

Additional future work could be developed aiming to extend end-user interface capabilities through the Optique platform after mapping a consistent dataset. Optique is the visual query system and uses -Ontop- as part of its system core.

8 REFERENCES

[1] S. Abiteboul, “Querying Semi-Structured Data,” *Proc. 6th Int. Conf. Database Theory*, pp. 1–18, 1997.

[2] J. C. Muñoz, “Ontology for modelling and understanding educational data and concepts: an application to Learning Analytics for Secondary project,” 2016.

[3] M.-R. Sancho, A. Canabate, and F. Sabate, “Contextualizing learning analytics for secondary schools at micro level,” in *2015 International Conference on Interactive Collaborative and Blended Learning (ICBL)*, 2015, pp. 70–75.

[4] K. M. Tolle, D. S. W. Tansley, and A. J. G. Hey, “The fourth Paradigm: Data-intensive scientific discovery,” in *Proceedings of the IEEE*, 2011, vol. 99, no. 8, pp. 1334–1337.

[5] D. Laney, “3D Data Management: Controlling Data Volume, Velocity, and Variety,” *Application Delivery Strategies*, no. February 2001, p. 4, 2001.

[6] S. Kaisler, F. Armour, J. A. Espinosa, and W. Money, “Big Data: Issues and Challenges Moving Forward,” in *46th Hawaii International Conference on System Sciences (HICSS)*, 2013, p. 995–1004 {997–998}.

[7] P. Hitzler and K. Janowicz, “Linked data, big data, and the 4th paradigm,” *SWJ*, vol. 4, no. 3, pp. 233–235, 2013.

[8] D. Ga, D. Djuric, and V. Deved, *Model driven engineering and ontology development*. 2009.

[9] E. Uprichard, “Focus: Big Data, Little Questions?,” *Discov. Soc.*, no. 1, pp. 1–5, 2013.

[10] D. Boyd and K. Crawford, “Critical Questions for Big Data,” *Information, Commun. Soc.*, vol. 15, no. 5, pp. 662–679, 2012.

[11] N. Marz and J. Warren, *Big Data - Principles and best practices of scalable realtime data systems*, vol. 37. 2013.

[12] K. C. Viktor Mayer-Schönberger, “A Summary of ‘Big Data: A Revolution That Will Transform How We Live, Work, and Think’ by Viktor Mayer-Schonberger and Kenneth Cukier [blog post],” *New Books Br.*, vol. 7, p. 242, 2013.

[13] R. Kitchin, “Big Data, new epistemologies and paradigm shifts,” *Big Data Soc.*, vol. 1, no. 1, p. 2053951714528481, 2014.

[14] R. Kitchin, “The opportunities, challenges and risks of big data for official statistics,” *Stat. J. IAOS*, pp. 471–481, 2015.

[15] O. P. John and S. Srivastava, “The Big Five trait taxonomy: History, measurement, and theoretical perspectives,” *Handb. Personal. Theory Res.*, vol. 2, no. 510, pp. 102–138, 1999.

[16] R. Kitchin, G. McArdle, D. Boyd, K. Crawford, T. Bucher, M. Dodge, R. Kitchin, N. Marz, J. Warren, V. Mayer-Schonberger, and K. Cukier, “What makes Big Data, Big Data? Exploring the ontological characteristics of 26 datasets,” *Big Data Soc.*, vol. 3, no. 1, pp. 662–679, Feb. 2016.

[17] *Big data now current perspectives from O’Reilly radar.* O’Reilly Media, 2011.

[18] K. Janowicz and P. Hitzler, "The Digital Earth as knowledge engine," *Semant. Web*, vol. 3, no. 3, pp. 213–221, 2012.

[19] D. Harkes and E. Visser, "Unifying and Generalizing Relations in Role-Based Data Modeling and Navigation," *Int. Conf. Softw. Lang.*, 2014.

[20] H. Happel and S. Seedorf, "Applications of ontologies in software engineering," *Semat. Web Enabled Softw. Eng. …*, 2006.

[21] S. Abiteboul, I. Manolescu, P. Rigaux, and M. Rousset, *Web data management*. 2011.

[22] V. Maniraj and R. Sivakumar, "Ontology languages-A review," *Int. J. Comput.*, p. 889, 2010.

[23] F. von Baader, *The Description logic handbook : theory, implementation, and applications*. Cambridge : Cambridge University Press, 2003.

[24] F. Baader, S. Brandt, and C. Lutz, "Pushing the EL envelope," *IJCAI*, 2005.

[25] M. Krötzsch, F. Simancik, and I. Horrocks, "A description logic primer," *arXiv Prepr. arXiv1201.4089*, 2012.

[26] T. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquis.*, 1993.

[27] N. Guarino, "Concepts, attributes and arbitrary relations: some linguistic and ontological criteria for structuring knowledge bases," *Data Knowl. Eng.*, 1992.

[28] M. C. Marinescu, "Multidisciplinary research – big data in smart cities, PATC Course: Big Data Analytics." Barcelona Supercomputing Center, Barcelona, 2016.

[29] Z. Xiang, Y. Lin, and Y. He, "Ontobat: An Ontology-based semantic web approach for linked data processing and analysis," *Notes*, 2010.

[30] T. Kumazawa, O. Saito, K. Kozaki, T. Matsui, and R. Mizoguchi, "Toward knowledge structuring of sustainability science based on ontology engineering," *Sustain. Sci.*, vol. 4, no. 1, pp. 99–116, Apr. 2009.

[31] M. Panahiazar and A. Sheth, "Advancing data reuse in phyloinformatics using an ontology-driven Semantic Web approach," *BMC Med.*, 2013.

[32] P. Bellini, M. Benigni, R. Billero, and P. Nesi, "Km4City ontology building vs data harvesting and cleaning for smart-city services," *J. Vis.*, 2014.

[33] D. Calvanese, P. Liuzzo, A. Mosca, and J. Remesal, "Ontology-based data integration in EPNet: Production and distribution of food during the Roman Empire," *Appl. Artif. …*, 2016.

[34] C. Roussey, F. Pinet, M. Kang, and O. Corcho, "An introduction to ontologies and ontology engineering," *Ontol. Urban*, 2011.

[35] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari, "The wonderweb library of foundational ontologies," 2002.

[36] R. Mizoguchi, "YAMATO: yet another more advanced top-level ontology," *Proc. Sixth Australas. Ontol.*, 2010.

[37] S. Reed and D. Lenat, "Mapping ontologies into Cyc," *AAAI 2002 Conf. Work. Ontol.*, 2002.

[38] M. Bennett, "The financial industry business ontology: Best practice for big data," *J. Bank.*

Regul., vol. 14, no. 3–4, pp. 255–268, 2013.

[39] J. Fox, A. Alabassi, V. Patkar, T. Rose, and E. Black, “An ontological approach to modelling tasks and goals,” *Comput. Biol.*, 2006.

[40] A. Abdalla, Y. Hu, D. Carral, and N. Li, “An ontology design pattern for activity reasoning,” *Proc. 5th*, 2014.

[41] K. Hiroko, K. Kouji, H. Motohiro, and H. Takaaki, “Development of Ontology for Information Literary,” *Procedia Comput. Sci.*, vol. 60, pp. 170–177, 2015.

[42] O. Corcho, M. Fernández-López, and A. Gómez-Pérez, “Building legal ontologies with METHONTOLOGY and WebODE,” *Law Semant.*, 2005.

[43] D. Calvanese, G. De Giacomo, and D. Lembo, “Ontologies and databases: The DL-Lite approach,” *Inf. Syst.*, 2009.

[44] D. Calvanese, B. Cogrel, and S. Komla-Ebri, “Ontop: Answering SPARQL queries over relational databases,” *Semantic*, 2016.

[45] M. Kogalovsky, “Ontology-based data access systems,” *Program. Comput. Softw.*, 2012.

[46] E. Yu, “Towards modelling and reasoning support for early-phase requirements engineering,” *Eng. 1997., Proc. Third IEEE …*, 1997.

[47] A. Sheth, C. Ramakrishnan, and C. Thomas, “Semantics for the semantic web: The implicit, the formal and the powerful,” *Int. J.*, pp. 1–18, 2005.

[48] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati, “Linking Data to Ontologies,” Springer Berlin Heidelberg, 2008, pp. 133–173.

[49] D. Berardi, D. Calvanese, and G. De Giacomo, “Reasoning on UML class diagrams,” *Artif. Intell.*, 2005.

[50] D. Gasevic, D. Djuric, and V. Devedzic, “Converting UML to OWL ontologies,” *Proc. 13th*, 2004.

[51] D. Calvanese, G. De Giacomo, and D. Lembo, “Tractable reasoning and efficient query answering in description logics: The DL-Lite family,” *J. Autom.*, 2007.

[52] D. Calvanese, G. De Giacomo, D. Lembo, and M. Lenzerini, “Data complexity of query answering in description logics,” *Artif. Intell.*, 2013.

[53] D. Calvanese, G. De Giacomo, and D. Lembo, “The MASTRO system for ontology-based data access,” *Semantic*, 2011.

[54] R. Rosati and A. Almatelli, “Improving Query Answering over DL-Lite Ontologies.,” *KR*, 2010.

[55] M. Rodrıguez-Muro and D. Calvanese, “High performance query answering over DL-Lite ontologies,” *Proc. 13th Int.*, 2012.

[56] R. Rosati, “Query rewriting under extensional constraints in DL-Lite,” *25th Int. Work. Descr. Logics*, 2012.

[57] M. Rodrıguez-Muro and D. Calvanese, “Quest, a system for ontology based data access,” *OWL Exp. Dir.*, 2012.

[58] H. Simon, “The architecture of complexity,” *Facet. Syst. Sci.*, 1991.

[59] S. Lightstone, T. Teorey, and T. Nadeau, *Physical Database Design: the database professional's guide to exploiting indexes, views, storage, and more*. 2010.

[60] D. Calvanese, G. De Giacomo, D. Lembo, and M. Lenzerini, "Ontology-based Database Access.," *SEBD*, 2007.

[61] R. Hull, "A survey of theoretical research on typed complex database objects," *Databases*, 1988.

[62] M. Giese, D. Calvanese, P. Haase, and I. Horrocks, "Scalable end-user access to big data," *Big Data*, 2013.

[63] B. Henderson-Sellers, "Bridging metamodels and ontologies in software engineering," *J. Syst. Softw.*, 2011.

[64] F. Ruiz and J. Hilera, "Using ontologies in software engineering and technology," *Ontol. Softw. Eng. Softw.*, 2006.

[65] W. Dargie, Eldora, J. Mendez, C. Mobius, K. Rybina, V. Thost, and A.-Y. Turhan, "Situation recognition for service management systems using OWL 2 reasoners," in *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2013, pp. 31–36.

[66] E. Kharlamov, N. Solomakhina, and Ö. Özçep, "How semantic technologies can enhance data access at Siemens Energy," *Int. Semant.*, 2014.

[67] E. Kharlamov, E. Jiménez-Ruiz, and D. Zheleznyakov, "Optique: Towards OBDA systems for industry," *Ext. Semant. Web*, 2013.

[68] Ö. L. Özçep and R. Möller, "Ontology Based Data Access on Temporal and Streaming Data," Springer International Publishing, 2014, pp. 279–312.

[69] S. Brüggemann, K. Bereta, G. Xiao, and M. Koubarakis, "Ontology-Based Data Access for Maritime Security," Springer International Publishing, 2016, pp. 741–757.

[70] C. Daraio, M. Lenzerini, C. Leporelli, and P. Naggar, "The advantages of an Ontology-Based Data Management approach: openness, interoperability and data quality," *Scientometrics*, 2016.

[71] G. Xiao, M. Rezk, and M. Rodriguez-Muro, "Rules and ontology based data access," *Conf. Web …*, 2014.

[72] R. Kontchakov, M. Rezk, M. Rodríguez-Muro, G. Xiao, and M. Zakharyaschev, "Answering SPARQL Queries over Databases under OWL 2 QL Entailment Regime," Springer International Publishing, 2014, pp. 552–567.

[73] E. Jiménez-Ruiz, E. Kharlamov, and D. Zheleznyakov, "BootOX: Practical mapping of RDBs to OWL 2," *Int. Semant.*, 2015.

[74] E. Jiménez-Ruiz, B. Grau, Y. Zhou, and I. Horrocks, "Large-scale Interactive Ontology Matching: Algorithms and Implementation.," *ECAI*, 2012.

[75] A. Soylu, E. Kharlamov, and D. Zheleznyakov, "Ontology-based visual query formulation: An industry experience," *Symp. Vis. …*, 2015.

[76] S. Kikot, R. Kontchakov, and M. Zakharyaschev, "Conjunctive Query Answering with OWL 2 QL.," *KR*, 2012.

[77] C. Bizer and A. Seaborne, "D2RQ-treating non-RDF databases as virtual RDF graphs," *Proc.*

3rd Int. Semant., 2004.

[78] F. Priyatna, O. Corcho, and J. Sequeda, “Formalisation and experiences of R2RML-based SPARQL to SQL query translation using Morph,” *Proc. 23rd Int.*, 2014.

[79] A. Tian, J. Sequeda, and D. Miranker, “Qodi: Query as context in automatic data integration,” *Int. Semant. Web*, 2013.

[80] J. Sequeda and D. Miranker, “Ultrawrap: SPARQL execution on relational data,” *Web Semant. Sci. Serv. Agents*, 2013.

[81] C. Civili, M. Console, G. De Giacomo, and D. Lembo, “MASTRO STUDIO: Managing ontology-based data access applications,” *Proc.*, 2013.

[82] D. Lembo, M. Lenzerini, R. Rosati, and M. Ruzzi, “Inconsistency-tolerant query answering in ontology-based data access,” *Web Semant. Sci.*, 2015.

[83] A. Queralt, A. Artale, D. Calvanese, and E. Teniente, “OCL-Lite: Finite reasoning on UML/OCL conceptual schemas,” *Data Knowl. Eng.*, vol. 73, pp. 1–22, 2012.

[84] IEEE, “IEEE Standard for Learning Object Metadata,” *IEEE Std 1484.12.1-2002*, p. i-32, 2002.

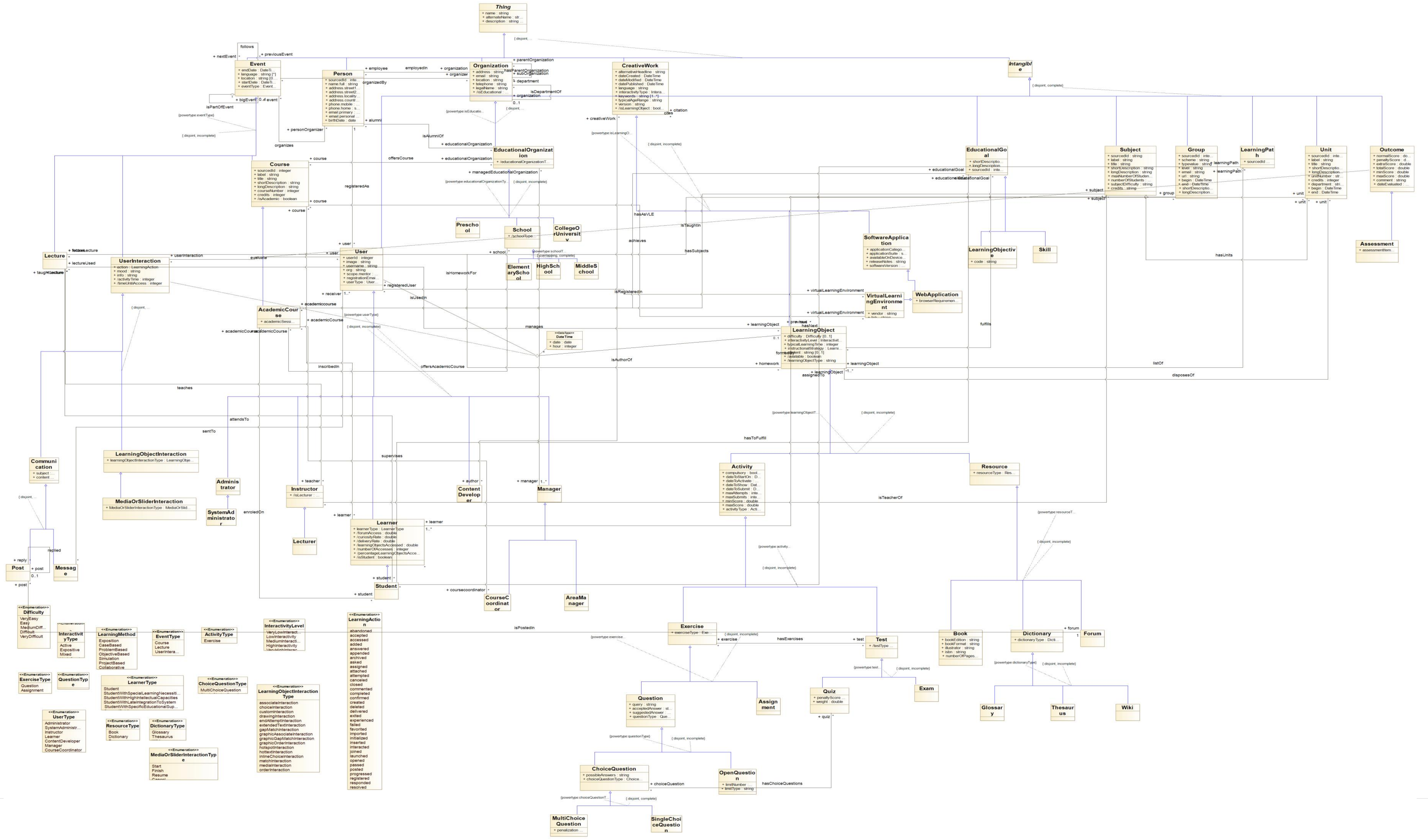
[85] R. Ghawi and N. Cullot, “Database-to-ontology mapping generation for semantic interoperability,” *VDBL’07 Conf. VLDB*, 2007.

[86] M. Lenzerini, “Ontology-based data management,” *Inf. Knowl. Manag.*, 2011.

[87] O. Romero, “Ontology Languages: OWL (and Description Logics), Open Data Course.” Universitat Politècnica de Catalunya, Barcelona, 2015.

Appendix A

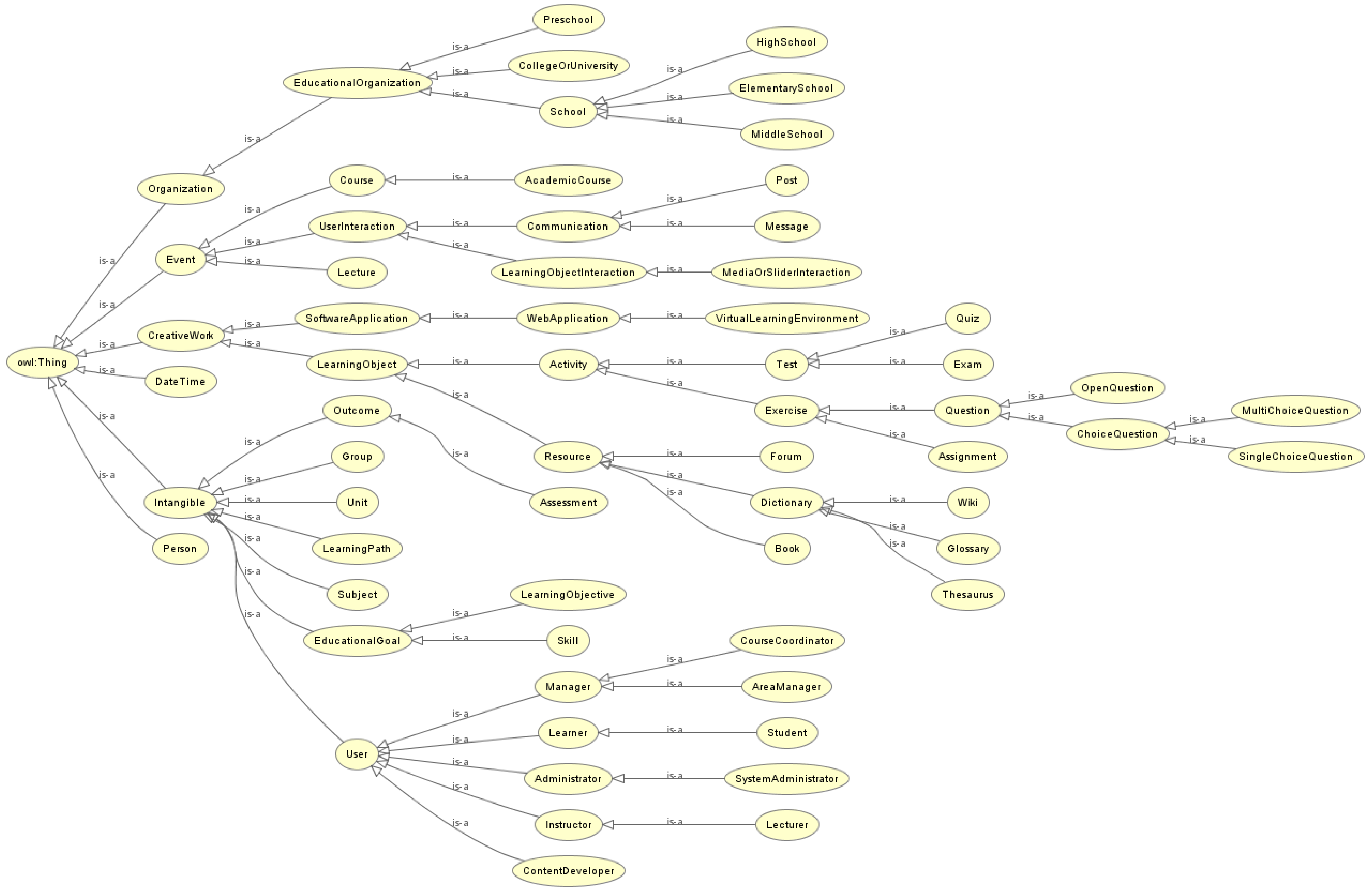
LA4S UML Ontology Schema



Thing
Event
Person
Organization
CreativeWork
Intangible
EducationalOrganization
Course
Prescho ol
School
CollegeOrUniversity
ElementarySchool
HighSchool
MiddleSchool
Lecture
UserInteraction
User
AcademicCourse
LearningObject
SoftwareApplication
VirtualLearningEnvironment
WebApplication
EducationalGoal
Subject
Group
LearningPath
Unit
Outcome
LearningObjective
Skill
Assessment
Activity
Resource
LearningObjectInteraction
MediaOrSliderInteraction
Communication
Administrator
SystemAdministrator
Instructor
Lecturer
ContentDeveloper
Manager
Learner
Student
CourseCoordinator
AreaManager
Post
Message
Exercise
Test
Question
Assignment
Quiz
Exam
ChoiceQuestion
OpenQuestion
MultiChoiceQuestion
SingleChoiceQuestion
Book
Dictionary
Forum
Glossary
Thesaurus
Wiki
<<Enumeration>> Difficulty
<<Enumeration>> InteractivityType
<<Enumeration>> LearningMethod
<<Enumeration>> EventType
<<Enumeration>> ActivityType
<<Enumeration>> InteractivityLevel
<<Enumeration>> LearningAction
<<Enumeration>> ExerciseType
<<Enumeration>> QuestionType
<<Enumeration>> LearnerType
<<Enumeration>> ChoiceQuestionType
<<Enumeration>> LearningObjectInteractionType
<<Enumeration>> UserType
<<Enumeration>> ResourceType
<<Enumeration>> DictionaryType
<<Enumeration>> MediaOrSliderInteractionType
follows
isPartOfEvent
employedIn
organizedBy
isAlumniOf
offersCourse
registeredAs
isDepartmentOf
hasAsVLE
isTaughtIn
achieves
hasSubjects
hasUnits
isHomeworkFor
isUsedIn
isRegisteredIn
manages
isAuthorOf
offersAcademicCourse
inscribedIn
teaches
attendsTo
sentTo
supervises
enroledOn
hasToFulfill
isTeacherOf
fulfills
listOf
disposesOf
isPostedIn
hasExercises
hasChoiceQuestions
evaluate
replied

Appendix B

LA4S OWL Ontology Schema



Appendix C

OBDA Mapping File

```
[PrefixDeclaration]
:               http://www.semanticweb.org/ontologies/2016/1/LA4SOnto#
owl:            http://www.w3.org/2002/07/owl#
rdf:            http://www.w3.org/1999/02/22-rdf-syntax-ns#
xml:            http://www.w3.org/XML/1998/namespace
xsd:            http://www.w3.org/2001/XMLSchema#
rdfs:           http://www.w3.org/2000/01/rdf-schema#

[SourceDeclaration]
sourceUri       datasource1
connectionUrl   jdbc:mysql://localhost:3306/dbagora
usernameroot
passwordXxxxX
driverClass     com.mysql.jdbc.Driver

[MappingDeclaration] @collection [[
mappingId       UserCMapp
target          :user/{userid}/ a :User ; :userName {username} .
source          SELECT DISTINCT ra."userid" as "userid",
CONCAT('user',ra."userid") as "username" FROM "mdl_role_assignments" ra
WHERE ra."roleid" in (3,4,5)

mappingId       EventCMapp
target          :event/{userid}/{id}/{datetimeId}/ a :Event ; :name
{name} ; :startDate {startDate} ; :endDate {endDate} .
source          SELECT lo."userid" as "userid", lo."contextinstanceid" as
"id", lo."timecreated" as "datetimeId", lo."eventname" as "name",
from_unixtime(lo."timecreated") as "startDate",
from_unixtime(lo."timecreated") as "endDate" FROM
"mdl_logstore_standard_log" lo WHERE lo."userid" IN
(48645,115221,1152786,1152353,1152366,11521046,1152304,1152352,1152310,115
2338,1152294,1152297,1152359,1152283,1152308,11521061,1152300,1152339,1152
783,115234,1152345,1152344,1152291,1152306,1152781,1152303,1152292,1152349
,1152314,1152309,1152348,1152333,1152287,1152305,1152296,1152288,1152313,1
152329,1152143,1152356,1152319,1152284,1152299,115250,1152205,1152226,1152
190,1152211,1152245,1152255,1152183,1152231,1152234,1152181,1152195,115225
2,1152197,115271,1152222,1152247,1152257,1152732,1152191,1152213,1152187,1
152267,1152265,1152198,1152784,1152290,11521081,1152358,1152311,1152342,11
52354,1152282,1152320,1152298,1152369,1152361,11521044,1152373,1152289,115
2346,1152331,15185001,15186661,39926821)

mappingId       UserInteractionCMapp
target          :event/{userid}/{id}/{datetimeId}/ a :UserInteraction ;
:action {action} ; :info {info} ; :activityTime {activityTime} ;
:timeUntilAccess {timeUntilAccess} .
source          SELECT lo."userid" as "userid", lo."contextinstanceid" as
"id", lo."timecreated" as "datetimeId", CASE WHEN lo."action" = 'viewed'
AND (lo."component" = 'mod_assign'  AND lo."eventname" LIKE
'%submission_form%') THEN 'initialized' WHEN lo."action" = 'viewed' THEN
'opened' WHEN lo."action" = 'uploaded' THEN 'attached' WHEN (lo."action" =
'updated' OR lo."action" = 'created') AND lo."component" =
'assignsubmission_file' THEN 'delivered' WHEN lo."action" = 'submitted'
THEN 'submitted' WHEN lo."action" = 'created' AND lo."eventname" LIKE
'%comment_created%' THEN 'commented' ELSE 'interacted' END as "action",
CONCAT(lo."eventname",'.',lo."objectid",'.',lo."objecttable") as
"info",(lo."timecreated"+ 1 - lo."timecreated") as "activityTime", 0 as
"timeUntilAccess" FROM "mdl_logstore_standard_log" lo WHERE lo."userid" IN
(48645,115221,1152786,1152353,1152366,11521046,1152304,1152352,1152310,115
2338,1152294,1152297,1152359,1152283,1152308,11521061,1152300,1152339,1152
783,115234,1152345,1152344,1152291,1152306,1152781,1152303,1152292,1152349
,1152314,1152309,1152348,1152333,1152287,1152305,1152296,1152288,1152313,1
152329,1152143,1152356,1152319,1152284,1152299,115250,1152205,1152226,1152
190,1152211,1152245,1152255,1152183,1152231,1152234,1152181,1152195,115225
2,1152197,115271,1152222,1152247,1152257,1152732,1152191,1152213,1152187,1
152267,1152265,1152198,1152784,1152290,11521081,1152358,1152311,1152342,11
52354,1152282,1152320,1152298,1152369,1152361,11521044,1152373,1152289,115
2346,1152331,15185001,15186661,39926821)

mappingId       hasOMapp
target          :userid/{userid}/ :has
:event/{userid}/{id}/{datetimeId}/ .
source          SELECT distinct lo."userid" as "userid",
lo."contextinstanceid" as "id", lo."timecreated" as "datetimeId" FROM
"mdl_logstore_standard_log" lo, "mdl_role_assignments" ra WHERE
ra."userid"=lo."userid" AND lo."userid" IN
(48645,115221,1152786,1152353,1152366,11521046,1152304,1152352,1152310,115
2338,1152294,1152297,1152359,1152283,1152308,11521061,1152300,1152339,1152
783,115234,1152345,1152344,1152291,1152306,1152781,1152303,1152292,1152349
,1152314,1152309,1152348,1152333,1152287,1152305,1152296,1152288,1152313,1
152329,1152143,1152356,1152319,1152284,1152299,115250,1152205,1152226,1152
190,1152211,1152245,1152255,1152183,1152231,1152234,1152181,1152195,115225
2,1152197,115271,1152222,1152247,1152257,1152732,1152191,1152213,1152187,1
152267,1152265,1152198,1152784,1152290,11521081,1152358,1152311,1152342,11
52354,1152282,1152320,1152298,1152369,1152361,11521044,1152373,1152289,115
2346,1152331,15185001,15186661,39926821)

mappingId       CreativeWorkCMapp
target          :creativework/{id}/ a :CreativeWork ; :name {name} ;
:dateCreated {dateCreated} ; :dateModified {dateModified} ; :language
{language} ; :interactivityType {interactivityType} ; :isLearningObject
{isLearningObject} .
source          SELECT cm."id" as "id", a."name" as "name",
from_unixtime(cm."added") as "dateCreated",
from_unixtime(a."timemodified") as "dateModified", 'català' as "language",
'Mixed' as "interactivityType",'true' as "isLearningObject" FROM
"mdl_assign" a JOIN "mdl_course_modules" cm ON a."id" = cm."instance" JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" in
('assign','assignment') UNION SELECT cm."id" as "id", fo."name" as "name",
from_unixtime(cm."added") as "dateCreated",
from_unixtime(fo."timemodified") as "dateModified", 'català' as
"language", 'Expositive' as "interactivityType",'true' as
"isLearningObject" FROM "mdl_folder" fo JOIN "mdl_course_modules" cm ON
fo."id" = cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('folder') UNION SELECT cm."id" as "id", foro."name" as
"name", from_unixtime(cm."added") as "dateCreated",
from_unixtime(foro."timemodified") as "dateModified", 'català' as
"language", 'Mixed' as "interactivityType",'true' as "isLearningObject"
```

```sql
FROM "mdl_forum" foro JOIN "mdl_course_modules" cm ON foro."id" =
cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name"
IN ('forum') UNION SELECT cm."id" as "id", j."name" as "name",
from_unixtime(cm."added") as "dateCreated", 0 as "dateModified", 'català'
as "language", 'Mixed' as "interactivityType",'true' as "isLearningObject"
FROM "mdl_jclic" j JOIN "mdl_course_modules" cm ON j."id" = cm."instance"
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('jclic')
UNION SELECT cm."id" as "id", q."name" as "name",
from_unixtime(cm."added") as "dateCreated",
from_unixtime(q."timemodified") as "dateModified", 'català' as "language",
'Active' as "interactivityType",'true' as "isLearningObject" FROM
"mdl_quiz" q JOIN "mdl_course_modules" cm ON q."id" = cm."instance" JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('quiz') UNION
SELECT cm."id" as "id", r."name" as "name", from_unixtime(cm."added") as
"dateCreated", from_unixtime(r."timemodified") as "dateModified", 'català'
as "language", 'Mixed' as "interactivityType",'true' as "isLearningObject"
FROM "mdl_resource" r JOIN "mdl_course_modules" cm ON r."id" =
cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name"
IN ('resource') UNION SELECT cm."id" as "id", u."name" as "name",
from_unixtime(cm."added") as "dateCreated",
from_unixtime(u."timemodified") as "dateModified", 'català' as "language",
'Mixed' as "interactivityType",'true' as "isLearningObject" FROM "mdl_url"
u JOIN "mdl_course_modules" cm ON u."id" = cm."instance" JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('url') UNION
SELECT cm."id" as "id", w."name" as "name", from_unixtime(cm."added") as
"dateCreated", from_unixtime(w."timemodified") as "dateModified", 'català'
as "language", 'Mixed' as "interactivityType",'true' as "isLearningObject"
FROM "mdl_wiki" w JOIN "mdl_course_modules" cm ON w."id" = cm."instance"
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('wiki')
UNION SELECT cm."id" as "id", b."name" as "name",
from_unixtime(cm."added") as "dateCreated",
from_unixtime(b."timemodified") as "dateModified", 'català' as "language",
'Expositive' as "interactivityType",'true' as "isLearningObject" FROM
"mdl_book" b JOIN "mdl_course_modules" cm ON b."id" = cm."instance" JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('book') UNION
SELECT cm."id" as "id", g."name" as "name", from_unixtime(cm."added") as
"dateCreated", from_unixtime(g."timemodified") as "dateModified", 'català'
as "language", 'Expositive' as "interactivityType",'true' as
"isLearningObject" FROM "mdl_glossary" g JOIN "mdl_course_modules" cm ON
g."id" = cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('glossary') UNION SELECT cm."id" as "id", l."name" as "name",
from_unixtime(cm."added") as "dateCreated",
from_unixtime(l."timemodified") as "dateModified", 'català' as "language",
'Expositive' as "interactivityType",'true' as "isLearningObject" FROM
"mdl_label" l JOIN "mdl_course_modules" cm ON l."id" = cm."instance" JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('label') UNION
SELECT cm."id" as "id", s."name" as "name", from_unixtime(cm."added") as
"dateCreated", from_unixtime(s."timemodified") as "dateModified", 'català'
as "language", 'Active' as "interactivityType",'true' as
"isLearningObject" FROM "mdl_survey" s JOIN "mdl_course_modules" cm ON
s."id" = cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('survey') UNION SELECT cm."id" as "id", wk."name" as "name",
from_unixtime(cm."added") as "dateCreated",
from_unixtime(wk."timemodified") as "dateModified", 'català' as
"language", 'Mixed' as "interactivityType",'true' as "isLearningObject"
FROM "mdl_workshop" wk JOIN "mdl_course_modules" cm ON wk."id" =
cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name"
IN ('workshop') UNION SELECT cm."id" as "id", sc."name" as "name",
from_unixtime(cm."added") as "dateCreated",
from_unixtime(sc."timemodified") as "dateModified", 'català' as
"language", 'Undefined' as "interactivityType",'true' as
"isLearningObject" FROM "mdl_scorm" sc JOIN "mdl_course_modules" cm ON
sc."id" = cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('scorm') UNION SELECT cm."id" as "id", p."name" as "name",
from_unixtime(cm."added") as "dateCreated",
from_unixtime(p."timemodified") as "dateModified", 'català' as "language",
'Mixed' as "interactivityType",'true' as "isLearningObject" FROM
"mdl_page" p JOIN "mdl_course_modules" cm ON p."id" = cm."instance" JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('page') UNION
SELECT cm."id" as "id", d."name" as "name", from_unixtime(cm."added") as
"dateCreated", 0 as "dateModified", 'català' as "language", 'Mixed' as
"interactivityType",'true' as "isLearningObject" FROM "mdl_data" d JOIN
"mdl_course_modules" cm ON d."id" = cm."instance" JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('data') UNION SELECT cm."id" as
"id", h."name" as "name", from_unixtime(cm."added") as "dateCreated",
from_unixtime(h."timemodified") as "dateModified", 'català' as "language",
'Active' as "interactivityType",'true' as "isLearningObject" FROM
"mdl_hotpot" h JOIN "mdl_course_modules" cm ON h."id" = cm."instance" JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('hotpot') UNION
SELECT cm."id" as "id", q."name" as "name", from_unixtime(cm."added") as
"dateCreated", from_unixtime(q."timemodified") as "dateModified", 'català'
as "language", 'Active' as "interactivityType",'true' as
"isLearningObject" FROM "mdl_questionnaire" q JOIN "mdl_course_modules" cm
ON q."id" = cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id"
WHERE m."name" IN ('questionnaire') UNION SELECT cm."id" as "id", g."name"
as "name", from_unixtime(cm."added") as "dateCreated",
from_unixtime(g."timemodified") as "dateModified", 'català' as "language",
'Active' as "interactivityType",'true' as "isLearningObject" FROM
"mdl_geogebra" g JOIN "mdl_course_modules" cm ON g."id" = cm."instance"
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN
('geogebra')

mappingId       interactsOMapp
target          :event/{userid}/{id}/{datetimeId}/ :interacts
:creativework/{id}/ .
source          SELECT distinct lo."userid" as "userid",
lo."contextinstanceid" as "id", lo."timecreated" as "datetimeId" FROM
"mdl_course_modules" cm, "mdl_logstore_standard_log" lo WHERE
lo."contextinstanceid"=cm."id" AND lo."userid" IN
(48645,115221,1152786,1152353,1152366,11521046,1152304,1152352,1152310,115
2338,1152294,1152297,1152359,1152283,1152308,11521061,1152300,1152339,1152
783,115234,1152345,1152344,1152291,1152306,1152781,1152303,1152292,1152349
,1152314,1152309,1152348,1152333,1152287,1152305,1152296,1152288,1152313,1
152329,1152143,1152356,1152319,1152284,1152299,115250,1152205,1152226,1152
190,1152211,1152245,1152255,1152183,1152231,1152234,1152181,1152195,115225
2,1152197,115271,1152222,1152247,1152257,1152732,1152191,1152213,1152187,1
152267,1152265,1152198,1152784,1152290,11521081,1152358,1152311,1152342,11
52354,1152282,1152320,1152298,1152369,1152361,11521044,1152373,1152289,115
2346,1152331,15185001,15186661,39926821)

mappingId       SubjectCMapp
target          :subject/{idSubject}/ a :Subject ; :title {title} .
source          SELECT  "id" as "idSubject", "fullname" as "title" FROM
"mdl_course"

mappingId       UnitCMapp
target          :unit/{idUnit}/ a :Unit ; :title {title} ; :unitNumber
{unitNumber} .
```

```
source          SELECT "id" as "idUnit", IFNULL("name",'NULL VALUE')as
"title", "section" as "unitNumber" FROM "mdl_course_sections"

mappingId       InstructorCMapp
target          :user/{userid}/ a :Instructor ; :isLecturer {isLecturer}
.
source          SELECT DISTINCT ra."userid" as "userid", 'true' as
"isLecturer" FROM "mdl_role_assignments" ra WHERE ra."roleid" in (4)

mappingId       LearnerCMapp
target          :user/{userid}/ a :Learner ; :forumAccess {forumAccess} ;
:curiosityRate {curiosityRate} ; :deliveryRate {deliveryRate} ;
:learningObjectAccessed {learningObjectAccessed} ; :numberOfAccesses
{numberOfAccesses} ; :percentageLearningObjectsAccessed
{percentageLearningObjectsAccessed} ; :isStudent {isStudent} .
source          SELECT DISTINCT ra."userid" as "userid", 0 as
"forumAccess", 0 as "curiosityRate", 0 as "deliveryRate", 0 as
"learningObjectAccessed", 0 as "percentageLearningObjectsAccessed", 0 as
"numberOfAccesses", 'true' as "isStudent" FROM "mdl_role_assignments" ra
WHERE ra."roleid" in (5)

mappingId       ContentDevCMapp
target          :user/{userid}/ a :ContentDeveloper .
source          SELECT  DISTINCT ra."userid" as "userid" FROM
"mdl_role_assignments" ra WHERE ra."roleid" in (3)

mappingId       LearningObjectCMapp
target          :creativework/{id}/ a :LearningObject ;
:interactivityLevel {interactivityLevel} ; :learningObjectType
{learningObjectType} ; :available {available} .
source          SELECT cm."id" as "id",'high' as "interactivityLevel",
'activity' as "learningObjectType",'true' as "available" FROM
"mdl_course_modules" cm JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" in ('assign','assignment') UNION SELECT cm."id" as "id",'medium'
as "interactivityLevel", 'activity' as "learningObjectType",'true' as
"available" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('jclic') UNION SELECT cm."id" as
"id", 'high' as "interactivityLevel", 'activity' as
"learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('quiz')
UNION SELECT cm."id" as "id", 'high' as "interactivityLevel", 'activity'
as "learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('survey')
UNION SELECT cm."id" as "id", 'high' as "interactivityLevel", 'activity'
as "learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN
('workshop') UNION SELECT cm."id" as "id", 'medium' as
"interactivityLevel", 'activity' as "learningObjectType",'true' as
"available" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('scorm') UNION SELECT cm."id" as
"id", 'medium' as "interactivityLevel", 'activity' as
"learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('data')
UNION SELECT cm."id" as "id", 'high' as "interactivityLevel", 'activity'
as "learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('hotpot')
UNION SELECT cm."id" as "id", 'high' as "interactivityLevel", 'activity'
```

```
as "learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN
('questionnaire') UNION SELECT cm."id" as "id", 'medium' as
"interactivityLevel", 'activity' as "learningObjectType",'true' as
"available" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('geogebra') UNION SELECT cm."id"
as "id", 'low' as "interactivityLevel", 'resource' as
"learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('folder')
UNION SELECT cm."id" as "id", 'high' as "interactivityLevel", 'resource'
as "learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('forum')
UNION SELECT cm."id" as "id", 'low' as "interactivityLevel", 'resource' as
"learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN
('resource') UNION SELECT cm."id" as "id", 'low' as "interactivityLevel",
'resource' as "learningObjectType",'true' as "available" FROM
"mdl_course_modules" cm JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('url') UNION SELECT cm."id" as "id", 'medium' as
"interactivityLevel", 'resource' as "learningObjectType",'true' as
"available" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('wiki') UNION SELECT cm."id" as
"id", 'VeryLow' as "interactivityLevel", 'book' as
"learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('book')
UNION SELECT cm."id" as "id", 'low' as "interactivityLevel", 'resource' as
"learningObjectType",'true' as "available" FROM "mdl_course_modules" cm
JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN
('glossary') UNION SELECT cm."id" as "id", 'low' as "interactivityLevel",
'resource' as "learningObjectType",'true' as "available" FROM
"mdl_course_modules" cm JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('label') UNION SELECT cm."id" as "id", 'medium' as
"interactivityLevel", 'resource' as "learningObjectType",'true' as
"available" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('page')

mappingId       ActivitytCMapp
target          :creativework/{id}/ a :Activity ; :activityType
{activityType} .
source          SELECT cm."id" as "id", 'exercise' as "activityType" FROM
"mdl_course_modules" cm  JOIN "mdl_modules" m ON cm."module" = m."id"
WHERE m."name" in ('assign','assignment') UNION SELECT cm."id" as "id",
'exercise' as "activityType" FROM "mdl_course_modules" cm JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('jclic') UNION
SELECT cm."id" as "id", 'test' as "activityType" FROM "mdl_course_modules"
cm  JOIN "mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN
('quiz') UNION SELECT cm."id" as "id", 'test' as "activityType" FROM
"mdl_course_modules" cm JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('survey') UNION SELECT cm."id" as "id", 'exercise' as
"activityType" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('workshop') UNION SELECT cm."id"
as "id", 'exercise' as "activityType" FROM "mdl_course_modules" cm JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('scorm') UNION
SELECT cm."id" as "id", 'exercise' as "activityType" FROM
"mdl_course_modules" cm JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('data') UNION SELECT cm."id" as "id", 'test' as
"activityType" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('hotpot') UNION SELECT cm."id" as
"id", 'test' as "activityType" FROM "mdl_course_modules" cm JOIN
```

```
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN
('questionnaire') UNION SELECT cm."id" as "id", 'exercise' as
"activityType" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('geogebra')

mappingId       ExerciseCMapp
target          :creativework/{id}/ a :Exercise .
source          SELECT cm."id" as "id" FROM "mdl_assign" a JOIN
"mdl_course_modules" cm ON a."id" = cm."instance" JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" in ('assign','assignment') UNION
SELECT cm."id" as "id" FROM "mdl_jclic" j JOIN "mdl_course_modules" cm ON
j."id" = cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('jclic') UNION SELECT cm."id" as "id" FROM "mdl_scorm" sc
JOIN "mdl_course_modules" cm ON sc."id" = cm."instance" JOIN "mdl_modules"
m ON cm."module" = m."id" WHERE m."name" IN ('scorm') UNION SELECT
distinct cm."id" as "id" FROM "mdl_geogebra" g JOIN "mdl_course_modules"
cm ON g."id" = cm."instance" JOIN "mdl_modules" m ON cm."module" = m."id"
WHERE m."name" IN ('geogebra')

mappingId       AssignmentCMapp
target          :creativework/{id}/ a :Assignment .
source          SELECT cm."id" as "id" FROM "mdl_course_modules" cm  JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" in
('assign','assignment')

mappingId       TestCMapp
target          :creativework/{id}/ a :Test .
source          SELECT cm."id" as "id" FROM "mdl_course_modules" cm  JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('quiz') UNION
SELECT cm."id" as "id" FROM "mdl_course_modules" cm  JOIN "mdl_modules" m
ON cm."module" = m."id" WHERE m."name" IN ('survey') UNION SELECT cm."id"
as "id" FROM "mdl_course_modules" cm  JOIN "mdl_modules" m ON cm."module"
= m."id" WHERE m."name" IN ('hotpot') UNION SELECT cm."id" as "id" FROM
"mdl_course_modules" cm  JOIN "mdl_modules" m ON cm."module" = m."id"
WHERE m."name" IN ('questionnaire')

mappingId       QuizCMapp
target          :creativework/{id}/ a :Quiz .
source          SELECT cm."id" as "id" FROM "mdl_course_modules" cm  JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" in ('quiz')

mappingId       ResourceCMapp
target          :creativework/{id}/ a :Resource ; :resourceType
{resourceType} .
source          SELECT cm."id" as "id", 'folder' as "resourceType" FROM
"mdl_course_modules" cm JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('folder') UNION SELECT cm."id" as "id", 'forum' as
"resourceType" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('forum') UNION SELECT cm."id" as
"id", 'resource' as "resourceType" FROM "mdl_course_modules" cm JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('resource')
UNION SELECT cm."id" as "id", 'url' as "resourceType" FROM
"mdl_course_modules" cm JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('url') UNION SELECT cm."id" as "id", 'wiki' as "resourceType"
FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON cm."module" = m."id"
WHERE m."name" IN ('wiki') UNION SELECT cm."id" as "id", 'book' as
"resourceType" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('book') UNION SELECT cm."id" as
"id", 'glossary' as "resourceType" FROM "mdl_course_modules" cm JOIN
```

```
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('glossary')
UNION SELECT cm."id" as "id", 'label' as "resourceType" FROM
"mdl_course_modules" cm JOIN "mdl_modules" m ON cm."module" = m."id" WHERE
m."name" IN ('label') UNION SELECT cm."id" as "id", 'page' as
"resourceType" FROM "mdl_course_modules" cm JOIN "mdl_modules" m ON
cm."module" = m."id" WHERE m."name" IN ('page')

mappingId       DictionaryCMapp
target          :creativework/{id}/ a :Dictionary ; :dictionaryType
{dictionaryType} .
source          SELECT cm."id" as "id", 'wiki' as "dictionaryType" FROM
"mdl_wiki" w JOIN "mdl_course_modules" cm ON w."id" = cm."instance" JOIN
"mdl_modules" m ON   cm."module" = m."id" WHERE m."name" IN ('wiki') UNION
SELECT cm."id" as "id", 'glossary' as "dictionaryType" FROM "mdl_glossary"
g JOIN "mdl_course_modules" cm ON g."id" = cm."instance" JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" IN ('glossary')

mappingId       ForumCMapp
target          :creativework/{id}/ a :Forum .
source          SELECT cm."id" as "id" FROM "mdl_course_modules" cm JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" in ('forum')

mappingId       BookCMapp
target          :creativework/{id}/ a :Book .
source          SELECT cm."id" as "id" FROM "mdl_course_modules" cm JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" in ('book')

mappingId       WikiCMapp
target          :creativework/{id}/ a :Wiki .
source          SELECT cm."id" as "id" FROM "mdl_course_modules" cm JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" in ('wiki')

mappingId       GlossaryCMapp
target          :creativework/{id}/ a :Glossary .
source          SELECT cm."id" as "id" FROM "mdl_course_modules" cm JOIN
"mdl_modules" m ON cm."module" = m."id" WHERE m."name" in ('glossary')

mappingId       hasUnitsOMapp
target          :subject/{idSubject}/ :hasUnits :unit/{idUnit}/ .
source          SELECT c."id" as "idSubject", cs."id" as "idUnit"  FROM
"mdl_course" c  JOIN
                        "mdl_course_sections" cs ON c."id" = cs."course"

mappingId       DateTimeCMapp
target          :datetime/{datetimeId}/ a :DateTime ; :fullDate
{fullDate} ; :date {date} ; :hour {hour} ; :minutes {minutes} ; :seconds
{seconds} .
source          SELECT distinct lo."timecreated" as "datetimeId",
from_unixtime(lo."timecreated") as "fullDate",
DATE_FORMAT(FROM_UNIXTIME(lo."timecreated"), '%d%m%Y') AS "date",
DATE_FORMAT(FROM_UNIXTIME(lo."timecreated"), '%k') AS "hour",
DATE_FORMAT(FROM_UNIXTIME(lo."timecreated"), '%i') AS "minutes",
DATE_FORMAT(FROM_UNIXTIME(lo."timecreated"), '%s') AS "seconds" FROM
"mdl_logstore_standard_log" lo WHERE lo."userid" IN
(48645,115221,1152786,1152353,1152366,11521046,1152304,1152352,1152310,115
2338,1152294,1152297,1152359,1152283,1152308,11521061,1152300,1152339,1152
783,115234,1152345,1152344,1152291,1152306,1152781,1152303,1152292,1152349
,1152314,1152309,1152348,1152333,1152287,1152305,1152296,1152288,1152313,1
152329,1152143,1152356,1152319,1152284,1152299,115250,1152205,1152226,1152
190,1152211,1152245,1152255,1152183,1152231,1152234,1152181,1152195,115225
2,1152197,115271,1152222,1152247,1152257,1152732,1152191,1152213,1152187,1
152267,1152265,1152198,1152784,1152290,11521081,1152358,1152311,1152342,11
52354,1152282,1152320,1152298,1152369,1152361,11521044,1152373,1152289,115
2346,1152331,15185001,15186661,39926821)

mappingId       atOMapp
target          :event/{userid}/{id}/{datetimeId}/ :at
:datetime/{datetimeId}/ .
source          SELECT distinct lo."userid" as "userid",
lo."contextinstanceid" as "id", lo."timecreated" as "datetimeId" FROM
"mdl_logstore_standard_log" lo WHERE lo."userid" IN
(48645,115221,1152786,1152353,1152366,11521046,1152304,1152352,1152310,115
2338,1152294,1152297,1152359,1152283,1152308,11521061,1152300,1152339,1152
783,115234,1152345,1152344,1152291,1152306,1152781,1152303,1152292,1152349
,1152314,1152309,1152348,1152333,1152287,1152305,1152296,1152288,1152313,1
152329,1152143,1152356,1152319,1152284,1152299,115250,1152205,1152226,1152
190,1152211,1152245,1152255,1152183,1152231,1152234,1152181,1152195,115225
2,1152197,115271,1152222,1152247,1152257,1152732,1152191,1152213,1152187,1
152267,1152265,1152198,1152784,1152290,11521081,1152358,1152311,1152342,11
52354,1152282,1152320,1152298,1152369,1152361,11521044,1152373,1152289,115
2346,1152331,15185001,15186661,39926821)

mappingId       disposesOfOMapp
target          :unit/{idUnit}/ :disposesOf :creativework/{id}/ .
source          SELECT cs."id" as "idUnit",  cm."id" as "id" FROM
"mdl_course_sections" cs JOIN "mdl_course_modules" cm ON
cs."course"=cm."course"

mappingId       GroupCMapp
target          :group/{sourceId}/ a :Group ; :sourcedId {sourcedId} ;
:name {name} ; :begin {begin} ; :end {end} .
source          SELECT g."id" as "sourceId", g."id" as "sourcedId",
g."name" as "name", from_unixtime(g."timecreated") as "begin",
from_unixtime(g."timemodified") as "end" FROM "mdl_groups" g

mappingId       isFormedByOMapp
target          :group/{sourcedId}/ :isFormedBy :user/{userid} .
source          SELECT distinct g."id" as "sourcedId", ra."userid" as
"userid" FROM "mdl_role_assignments" ra  JOIN "mdl_groups_members" gm ON
gm."userid" = ra."userid"  JOIN "mdl_groups" g ON gm."groupid"=g."id"
WHERE ra."roleid" in ('5') AND ra."userid" IN
(48645,115221,1152786,1152353,1152366,11521046,1152304,1152352,1152310,115
2338,1152294,1152297,1152359,1152283,1152308,11521061,1152300,1152339,1152
783,115234,1152345,1152344,1152291,1152306,1152781,1152303,1152292,1152349
,1152314,1152309,1152348,1152333,1152287,1152305,1152296,1152288,1152313,1
152329,1152143,1152356,1152319,1152284,1152299,115250,1152205,1152226,1152
190,1152211,1152245,1152255,1152183,1152231,1152234,1152181,1152195,115225
2,1152197,115271,1152222,1152247,1152257,1152732,1152191,1152213,1152187,1
152267,1152265,1152198,1152784,1152290,11521081,1152358,1152311,1152342,11
52354,1152282,1152320,1152298,1152369,1152361,11521044,1152373,1152289,115
2346,1152331,15185001,15186661,39926821)
]]
```