A Model-driven Approach for Representing Clinical Archetypes for Semantic Web Environments

Catalina Martínez-Costa[1], Marcos Menárguez-Tortosa[1],

Jesualdo Tomás Fernández-Breis[1], José Alberto Maldonado[2]

[1]*Departamento de Informática y Sistemas, Facultad de Informática, Universidad de Murcia, CP 30071 Murcia, Spain*

[2]*Biomedical Informatics Group (IBIME), ITACA Institute, Technical University of Valencia, Valencia, Spain.*

Address for correspondence:

Jesualdo Tomás Fernández-Breis

Departamento de Informática y Sistemas

Facultad de Informática

Campus de Espinardo

Universidad de Murcia

CP 30100 Murcia

Spain

Email: jfernand@um.es

Fax: +34968364151

Phone: +34968364613

Abstract— The lifelong clinical information of any person supported by electronic means configures his Electronic Health Record (EHR). This information is usually distributed among several independent and heterogeneous systems that may be syntactically or semantically incompatible. There are currently different standards for representing and exchanging EHR information among different systems. In advanced EHR approaches, clinical information is represented by means of archetypes. Most of these approaches use ADL to specify archetypes. However, ADL has some drawbacks when attempting to perform semantic activities in Semantic Web environments. In this work, Semantic Web technologies are used to specify clinical archetypes for advanced EHR architectures. The advantages of using OWL instead of ADL are described and discussed in this work. Moreover, a solution combining Semantic Web and Model-Driven Engineering technologies is proposed to transform ADL into OWL for the CEN ENV13606 EHR architecture.

KEYWORDS:

Biomedical Informatics, Electronic Healthcare Records, Archetypes, Semantic Web, Ontology, Model-Driven Engineering

1 Introduction

One of the basic needs for healthcare professionals is to be able to access the clinical information of patients in an understandable and normalized way. If that information is supported by electronic means, the Electronic Healthcare Record (EHR) concept arises. This information is usually distributed among several independent and heterogeneous systems that may be syntactically or semantically incompatible. EHR systems, as pointed out in [4], must support life-long EHR, be technology and data format independent, facilitate sharing of EHRs via interoperability at data and knowledge levels, integrate with any/multiple terminologies, support for clinical data structures and prioritize the patient/clinician interaction. As stated in [25], not only is medicine domain big, but also open-ended because new information, finer grained details or new relationships are always being discovered or becoming relevant. In the health domain there are O(100.000) –O(1.000.000) terms and probably 200.000 – 300.000 concepts. As a consequence, no fixed enumerated list of medical concepts can ever be complete. This implies that a traditional information model will never be completely adapted to the clinical requirements and its continuous evolution (Beale, 2001). Given this situation, advanced standards and architectures [4;9] for representing and communicating electronic healthcare records make use of an architecture based on the dual model approach. This architecture defines two conceptual levels [2]: (1) reference model; and (2) archetype model. In this work, special attention will be paid to this second level, archetype model, where archetypes define distinct domain-level concepts in the form of structured and constrained combinations of the classes contained in the reference model. A basic benefit of the archetype approach is that they are shareable and reusable. One of such architectures is the CEN ENV 13606 standard, proposed by the CEN/TC251, Technical Committee 251 of the Normalization European Committee [41], on which this research work is focused.

Clinical activities also need the exploitation of the clinical information represented by means of archetypes, which are typically represented by using the Archetype Definition Language (ADL) [50]. The exploitation of clinical information requires carrying out a set of activities, such as comparisons, classifications, integration of clinical information coming from different systems, based on different

EHR architectures and so on. These activities are related to the semantic management and interoperability of clinical systems and information. Unfortunately, the syntactic orientation and limitations of ADL makes it insufficient to achieve these goals. Hence, providing a representation of clinical archetypes and information suitable for performing such semantic operations is a critical issue. In this sense, the advances in the Semantic Web community make it a candidate technology for supporting such knowledge intensive tasks related to archetypes and EHR systems. The goal of this work is then to provide mechanisms for representing archetypes in Semantic Web- manageable manner, and the proposed solution combines Semantic Web and Model-driven Engineering technologies in the context of Technological Spaces to obtain a Semantic Web manageable representation of clinical information.

Finally, the structure of this work is described next. First, Section 2 includes a brief introduction to EHR standards and archetypes, and the technologies used in this work. Section 3 contains a discussion on the suitability of ADL and OWL for representing and exploiting clinical archetypes. Later, the proposed solution is described in Section 4. Finally, Section 5 will contain the discussion and the main conclusions drawn from this work.

2 Methods

This section describes the main concepts and the motivation of this work. There is an introduction about clinical standards, focusing on the CEN ENV13606 specification. Later, descriptions of clinical archetypes and the Archetype Description Language (ADL) are provided, as well as a discussion of the limitations of this language. Next, the software technologies used in our methodological solution are presented.

2.1 Electronic Healthcare Records (EHR) standards

Nowadays, there are different advanced standards and architectures [4] for representing and communicating electronic healthcare records, such as HL7 [46], OpenEHR [50] and the CEN

ENV13606 [41] standard. Each one defines its own information models and manages the information in a particular way. This implies that clinical information systems of different clinical organizations might differ in how electronic healthcare records are managed. The last two mentioned standards follow a dual model architecture approach [2]. This architecture is based on the meta-modeling of healthcare records, which distinguishes two conceptual levels: (1) reference model, and (2) archetypes. The reference model represents the global features of the annotations of healthcare records, how they are aggregated and the context information required to meet the ethical, legal, etc requirements. This model defines the set of classes that forms the generic building blocks of the electronic healthcare record and it contains the non volatile features of the electronic healthcare record. An archetype models the common features of types of entities and, therefore, it defines the valid domain structures in terms of taxonomic ("is a class of") and partonomic ("is a part of") components. Archetypes restrict the business objects, which can be considered descriptors of domain ontological levels, defined in a reference model. Archetypes bridge the generality of business concepts defined in the reference model and the variability of clinical practice, thus becoming a standard tool to represent this issue. The second principle is that the information system is based on the Reference Model, and the valid healthcare records extracts are instances of this reference model.

In this work, the standard CEN ENV 13606 [41] is addressed. The CEN ENV 13606 specification is proposed by the CEN/TC251, Technical Committee 251 of the Normalization European Committee, has recently become an ISO standard. This standard intends to support interoperability between systems and to provide components for interaction with EHR services. For this purpose, it defines the following five parts, three of which are relevant for this work:

- Reference model: Generic information model for communicating the electronic healthcare record of any one patient.
- Archetype exchange specification: Generic information model and language for representing and communicating the definition of individual instances of archetypes.

- Reference archetypes and term lists: A range of archetypes reflecting a diversity of clinical requirements and settings, as a “starter set” for adopters and to illustrate how other clinical domains might similarly be represented.

2.2 Archetypes

As it has been previously mentioned, archetypes model the common features of types of entities and, therefore, they define the valid domain structures in terms of taxonomic (“is a class of”) and partonomic (“is a part of”) components, which conforms the particular structure of an archetype. These are structured models of domain content. In clinical setting, they refer to clinical concepts. An example of a clinical archetype might be a genetic condition defined by a clinician. The definition of this clinical archetype might contain the following information: the name of the genetic condition, the date of manifestation, the age of manifestation, the severity, the clinical description, the date of clinical recognition, the location, the complications, the date and age of resolution, and references and web links about this genetic condition. Each information item can be either simple (such as the clinical description) or complex (such as the complications, each described by the complicating problem and the clinical description). When defining a clinical archetype, each information item has a set of restrictions associated. For example, “the severity can take values from the range {mild, moderate, severe}” or “dates can be specified by using only the year and the month”.

Clinical archetypes contain specific clinical knowledge, so that they are usually built by domain experts, and they define valid data configurations. In fact, they are an attempt to standardize clinical practice. They can be used to control and validate the data obtained by clinicians and to guide the processing of clinical queries. In summary, their primary purpose is to provide a reusable, interoperable way of managing data creation, validation and querying, by ensuring that data conform to particular structures and semantic constraints.

Furthermore, archetype construction is also related to issues such as versioning, specialization, and composition. First, the clinical context is a dynamic environment, which has continuous research clinical results so that how to perform an activity is likely to evolve over time. Therefore, archetype

construction approaches must take into account this fact and allow for defining and managing versions. Second, reusability is obviously positive in order to save efforts and time and increasing productivity. Some archetypes might be defined as extensions or specializations of existing ones. For instance, the definition of a genetic condition can be viewed as the specialization of an archetype for generic problems. This is another issue that archetype management approaches must consider. Finally, some archetypes might be structural parts of other archetypes. In this case, mechanisms for managing this partonomic structure must also be provided.

2.2.1 Archetype Modeling in ADL

The Archetype Definition Language (ADL) is a formal language for expressing archetypes. ADL documents are structured text files, whose structure is independent from any particular standard or domain. Generally speaking, ADL is not a language for clinical domains. It can be used for defining any type of archetype. However, we consider it in this work as a language for specifying clinical archetypes.

Figure 1 shows an extract of an ADL archetype for Cholesterol according to the CEN standard. There, the different ADL sections can be identified: header, description, definition and ontology. The header includes the name of the archetype, specialization information and so on. In this example, this header includes the name of the archetype (*CEN-EHR-ENTRY.Cholesterol.v1*) and the language it is written in. Concerning the name it is a formatted string which includes the EHR standard (*CEN-EHR*), the clinical data structure which is built (*ENTRY*), the name of the clinical concept (*Cholesterol*) and the version identifier (v1). The description section includes audit information, such as original author, lifecycle status or purpose. The definition section contains the structure and restrictions associated to the clinical concept defined by the archetype. In this example, it can be noticed that cholesterol is defined by an entry, which has a list of items. In this case, only one item has been defined, the element whose value is in the range [0,1000], and is measured in mg/ml. Finally, the ontology section includes the terminological definition and bindings. Here, the meaning for at0000 and at0001 is provided as well as a binding for at0001 in the external terminology LOINC

[51]. They refer to *Cholesterol* in all cases. ADL is very flexible, since the same structure can be used for specifying archetypes for different reference models. However, we are talking about the same syntactic structure, but not semantic.

```
archetype

    CEN-EHR-ENTRY.Cholesterol.v1
concept
    [at0000]
description
    author = <"Unknown">
    status = <"draft">
    description ("en") = <
        purpose = <"CEN test">
    >
definition

ENTRY[at0000] occurrences matches {1..1} matches {
        items cardinality matches {0..1 ;ordered;unique} matches {
            ELEMENT[at0001] occurrences matches {0 ..1} matches {
                value matches {
                    PQ occurrences matches {1..1} matches {
                        units matches {
                            CS matches {
                                codeValue matches {"mg/dl"}
                            }
                        }
                        value matches {|0.0..1000.0|}
                    }}}}}
ontology
    primary_language = <"en">
    languages_available = <"en",...>
    term_definitions("en") = <
        items("at0000") = <
            text = <"Cholesterol">
            description = <"*">
        >
        items("at0001") = <
            text = <"Cholesterol">
            description = <"*">
        >
    >
    term_binding("en") = <
        ["LOINC"] = <
            items = <
                ["at0001"] = <[LOINC::12531-0]>
            >
        >
    >
    constraint_definitions("en") = < >
    constraint_binding("en") = < >
```

Figure 1. Extract of an ADL archetype

ADL archetypes are built on top of the Archetype Object Model (AOM). A partial view of AOM classes is shown in Figure 2. Two of such classes, namely, archetype ontology, and

C_Complex_Objects are the most relevant for our work, since they contain the information of the clinical concept. Hence, when processing an ADL archetype, a collection of AOM objects is obtained.

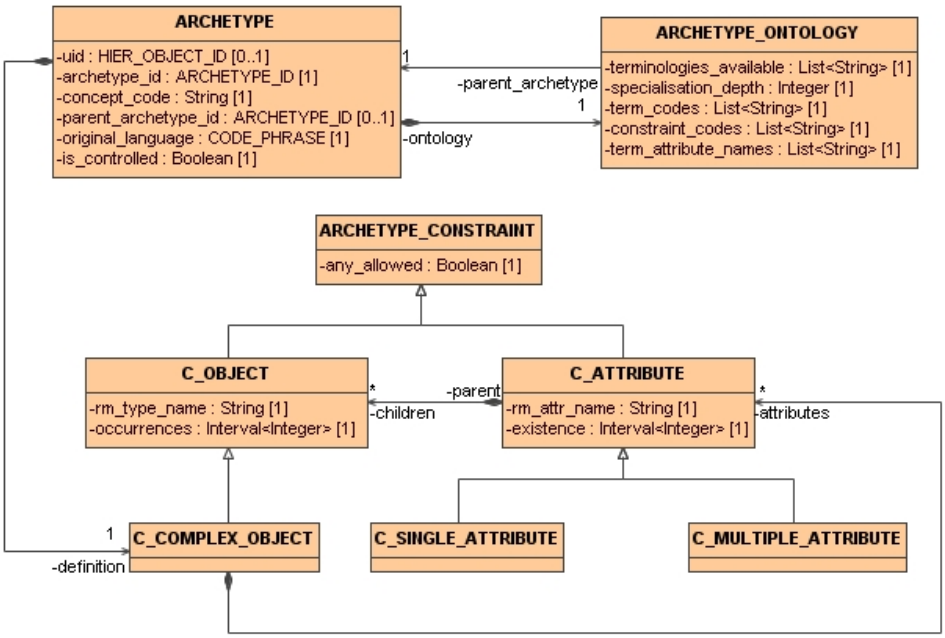

Figure 2. An overview of the Archetype Object Model

2.3 Semantic Web

The Semantic Web [3] is a vision of the future Web in which information is given explicit meaning, making it easier for machines to automatically process and integrate information available on the Web. Generally speaking, Semantic Web technologies promise to be capable of facilitating the management of knowledge and promote semantic interoperability between systems, so they might be helpful for the aforementioned tasks. There are different basic technologies for the success of the Semantic Web, amongst which the cornerstone technology is the ontology.

In literature, multiple definitions for ontology can be found (see for instance [14;35]). An ontology represents a common, shareable and reusable view of a particular application domain, and they give meaning to information structures that are exchanged by information systems [6]. An ontology can be seen as a semantic model containing concepts, their properties, interconceptual relations, and axioms related to the previous elements. Furthermore, ontology has a standard reference model to integrate

information known a knowledge sharing, and it is technology and data independent. In practical settings, ontologies have become widely used due to the advantages they have (see for instance [10]). On the one hand, ontologies are reusable, that is, a same ontology can be reused in different applications, either individually or in combination with other ontologies. On the other hand, ontologies are shareable, that is, their knowledge allows for being shared by a particular community. In a context of integration and interoperability, they facilitate the human understanding of the information, the access based on information and the integration of information of very different information systems. In this sense, ontologies allow for differentiating among resources, and this is especially useful when there are resources with redundant data.

The use of ontologies to represent biomedical knowledge is not new, since ontologies have been widely used in biomedical domains for the last years with different purposes. Medical concepts have been formalized by using ontologies (see for instance [33;29]). One of the most significant advances in bioinformatics was linked to ontologies, the development of the Gene Ontology [1]. In fact, the amount of bio-ontologies and related projects (e.g., the Open Biological Ontologies project [37]) is increasing. All these apportions reveal the usefulness of ontologies to represent biomedical knowledge, which is reinforced for our purpose by the use of ontologies related to EHR management (see for instance [19;27;32]). In addition to this, the EU-funded projects such as ARTEMIS represents an effort to provide semantically enriched web-services-based interoperability across OpenEHR and HL7 systems (see for instance [8]). More recently, the European project Semantic Health [30] also considers basic the use of Semantic Web technologies for representing clinical knowledge for achieving interoperability. Moreover, ontologies have also been used in biomedical domains for integration and interoperability (see [24;31;36;38]).

In this work, ontologies are proposed to represent clinical information semantics. The ontologies will be modeled by using the Ontology Web Language (OWL) [53], which is the recommendation of the W3C for the exchange of semantic content on the web. In particular, OWL-DL (where DL stands for “Description Logics”) is used, because of its decidability and computability nature. It offers enough expressiveness and the possibility of reasoning over the information that it describes.

2.4 Model-Driven Engineering

Model-Driven Engineering (MDE) is a software development discipline whose key element is the model. A model describes a physical, abstract or hypothetical reality, containing the information that allows the achievement of specific goals such as code generation, applications integration and interoperability, and so on. Moreover, models allow to increase the degree of abstraction of such tasks and to make them automatic.

On the other hand, MDE allows the formal definition of a modeling language. This allows saving time and resources in software maintenance, development of new applications and new features of existing software. Using metamodels implies the usage of a metamodeling architecture. The Object Management Group (OMG) [49] defines a four-level architecture [21]. Each level allows for distinguishing among the different conceptual levels taking part in the modeling of a system. These four levels are:

(M0) Instance, which represents the real system;

(M1) Model, which corresponds to the model of the application. Its concepts define the classifications of M0, whose elements are instances of the elements of M1.

(M2) Metamodel, which corresponds to the modeling languages. M2 and M1 are related in the same way as M1 and M0.

(M3) Metametamodel, which defines the elements of the modeling languages. This is the most abstract level, in which languages such as MOF [21] or Ecore [42] can be found. These languages provide the constructors and mechanisms for describing metamodels for modeling languages, such as UML, OWL or ADL.

Model transformations can be established between metamodels defined in the same metamodeling language. The QVT specification [20] allows for defining model transformations but at present there not exist tools that implements the specification. In recent years, several model transformation languages have been defined. RubyTL [28] is a rule-based hybrid transformation language for

defining the transformation rules in both declarative and imperative ways and includes significant features such as the organization of the rules in phases.

In MDE, software artifacts such as programs, ontologies or XML documents can play the role of model. They are defined in specific working areas named technical spaces (TS) [16]. A technical space is usually associated to a user community sharing concepts, knowledge and tools, and is defined by a pair of concepts such as Program/Grammar (Grammarware), Ontology/Top-level ontology (Semantic Web), Document/Schema (XML), Model/Metamodel (MDE) or Data/Schema (Databases).

Bridges can be established between MDE and several technical spaces so that the artifacts defined in such technical spaces can be represented as models. For instance, Grammarware and MDE are related by means of tools such as xText [44] for generating metamodels from grammars and parsers which instantiate models conforming to these metamodels. In addition, ontologies can be transformed into metamodels by following the ODM standard [22].

3. ADL and OWL for Supporting Knowledge-intensive Clinical Activities

In previous sections, it has been stated that ADL is currently the language to describe clinical archetypes. It has also been stated that knowledge intensive activities have to be performed in clinical settings, having clinical archetypes as an important clinical knowledge unit. The need for such activities makes it necessary to analyze whether ADL can facilitate them or the usage of a knowledge-oriented language such as OWL is recommended. In this section, the limitations of ADL and the benefits of OWL for supporting semantic processing and activities are described

3.1 ADL Limitations

An ADL clinical archetype has to be written for a particular information model. Thus, archetypes are built for CEN, OpenEHR, and so on. An ADL parser obtains objects from an abstract Archetype Object Model (AOM), so it has no information about particular reference models. In this way, the parsing process returns a collection of syntactic objects, which cannot be used as such to perform any

semantic activity. As a parsable syntax, ADL models are considered to have a formal relationship with structural models such as those expressed in UML. Given its genericity, the language does not provide any component that guarantees the consistency of clinical information. It can only offer consistency at archetype level, that is, the conformance of ADL/AOM principles. Therefore, in order to process ADL content, there is a need for two elements: an ADL parser in order to capture AOM objects and the parser of the particular information model to guarantee the clinical correctness of the information. Hence, if we want to perform a semantic processing of an ADL archetype, the document must identify the reference model. This is done in the identification of the archetype (see the first line of the ADL archetype shown in Figure 1).

The ontology section of ADL archetypes contains attributes such as terminologies_available, term_codes, term_attribute_names and constraint_codes which are modeled as lists of strings. In the case of *C_Object*, the type of object (from the reference model) is also defined by a string, as well as the name of the attribute in *C_Attribute*. However, most of these strings refer to classes of the reference or archetype models, so that this representation does not structure this information semantically. It would be more appropriate to model this reference through a relation between the corresponding classes. An example is the non-existence of explicit, semantic links between all the information concerning an archetype term. An archetype term is defined by its term definitions, term bindings, constraint definitions, and constraint bindings. These elements are not semantically or formally modeled and related to the corresponding elements. Moreover, this drawback is also applicable to the type of archetype term, since there is no explicit, semantic link between the archetype term and its corresponding clinical data structure in the reference model.

Let us briefly describe how the ADL archetype shown in Figure 1 would be parsed. The parser returns an *Archetype* object, which contains one property for each part. In this case we will focus on the part definition of the ADL shown in such figure. The definition of this *ENTRY* would be parsed as a *C_COMPLEX_OBJECT* which is defined through a set of properties, amongst which three are relevant for this discussion: (1) *rm_type_name*: String; (2) *node_id*: String; and (3) *attributes*: Set of *C_ATTRIBUTE*. The first property establishes the name of the type in the reference model (in this

case *ENTRY*).; *Node_id* would stand for at0000, and *attributes* refer to the constrained defined for the attributes included in the reference model type for *rm_type_name* . A *C_ATTRIBUTE* is also defined by an *rm_attribute_name*, that is the name of the attribute in the reference model, and has associated a set of constraints (*C_OBJECT*) on such attribute. The parsing of our ADL example would produce two main *C_COMPLEX_OBJECT* nodes, having the following values for the triples (*rm_type_name*, *node_id*, *attributes*): (1) ("ENTRY", "at0000", {items}); (2) ("ELEMENT", "at0001",{value}). The *C_ATTRIBUTE* would have other *C_OBJECT* associated to define the constraints on codeValue ("mg/dl") and value [0.0,1000.0]. The AOM graphical representation of this archetype shown in Figure 3.

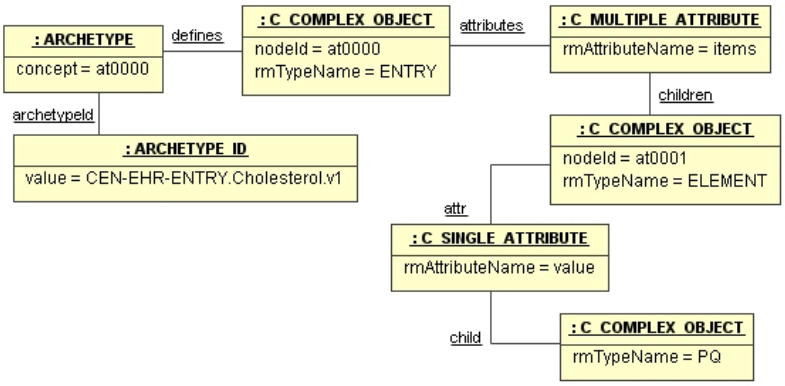

Figure 3. Extract of the Archetype Object Model of the Cholesterol archetype

Hence, the ADL parser produces a set of AOM objects with no explicit, semantic relations between them. The semantics is unknown for the parser and only the association between elements from the definition and ontology section might be ideally done by the parser by string matching. Unfortunately, it can be aware of the existence of objects and constraints from the definition section, but it does not know what *constraint_codes* or *term_codes* means.

Hence, the possibilities of reasoning over ADL are currently very limited, as well as the availability of tools to use and manage ADL content is reduced. Consequently, particular reasoning frameworks for each information model are needed.

3.2 OWL Benefits

The benefits of OWL can be discussed from two different perspectives: (1) the activities that can be better performed in OWL, and (2) the representation of knowledge. This section begins by discussing the first perspective. Archetypes can be designed by healthcare professionals in different ways, as it happens with ontologies. Hence, there is a clear need for management mechanisms. The Semantic Web community has been working for long in methodologies and tools for comparing different ontologies, merging ontologies, identifying inconsistencies and so on. We do not refer to non-compliance to the information model but to knowledge inconsistencies between different archetypes descriptions used in different healthcare institutions. Hence, these activities can be performed over OWL content in a more generic, easier and more efficient way than over ADL content, since OWL is the "de facto" knowledge representation language for Semantic Web environments.

Exchanging archetypes is a common task in archetype-oriented architectures. A particular system can receive unknown archetypes, which have to be classified in the particular archetype library. These classifications cannot be done by using current ADL technologies, whereas it becomes possible using OWL, because semantic similarity measurement techniques are available in the Semantic Web community. In fact, the archetype community is aware of the usefulness of Semantic Web technologies for classification. For instance, in [11] an OWL Archetype Ontology provides the necessary meta-information on archetypes for Domain Knowledge Governance. However, this approach uses the ontology with organizational purposes, whereas the proper archetype content might be used to automatically suggest such classifications by using OWL-based metrics.

On the other hand, terminologies are very important in biomedical domains and in archetype modeling. In fact, any clinical concept included in the archetype can be related to different terminologies. The most important terminologies, such as SNOMED [52], are currently in the process of adapting their representation to Semantic Web environments, so that OWL models for them are appearing. Having the representation of both clinical and terminological information in the same formalism would facilitate better clinical knowledge management. There are also a few approaches in

the Semantic Web community for mapping and merging different ontologies, so that, more complete archetypes can be built.

Another advantage of OWL against ADL is the large research community working on its development. OWL 1.0 was produced in 2004, OWL 1.1 [54] is available now, and different technologies and languages for querying, defining rules and exploiting OWL content are in progress.

Concerning the representation of knowledge, OWL allows for defining detailed, accurate, consistent, sound, and meaningful distinctions among the classes, properties, and relations. Moreover, an OWL-based archetype construction approach might guarantee the consistency of the knowledge, which cannot be granted by ADL. The first issue to address is whether OWL has enough expressivity to model clinical archetypes. OWL ontologies are structured through language primitives (i.e., subclass of) and user-defined properties. Restrictions over archetypes can also be established by using OWL restrictions or defining the appropriate elements. Archetype modeling implies specializations, versioning and composition. These issues, as they are understood in archetype modeling, can also be addressed in OWL. Hence, OWL seems to be appropriate to represent clinical archetypes and information about electronic healthcare records.

There are other differences between the ADL and OWL representations, such as how information is parsed and processed. OWL modeling brings all the information concerning a particular term together (code, name, binding, translations, constraints) so that a particular information item can be accessed and analyzed in its context. Moreover, the processing of the OWL document does both the parsing of the OWL and the capture of the consistent clinical information.

4. The POSEACLE Approach

According to the previous sections, it seems sensible to represent and manage clinical archetypes in OWL. Provided that ADL is currently the language used for such purpose, mechanisms for transform ADL archetypes into OWL ones are needed. In this section, the process of transforming the syntactic content of an ADL archetype into its semantic expression in OWL is described. This solution is comprised of the following steps: (1) Creation of syntactic models representing ADL content; (2)

Transforming syntactic models to semantic models conforming to CEN standard; and (3) Instantiation of OWL archetypes. In order to perform such steps, there is a need for an OWL ontology for representing clinical archetypes. The construction of this ontology is described first (see 4.1). Provided that the MDE technical space is used as the pivotal space for performing the transformation, bridges from ADL and OWL to MDE have to be built (see 4.2). At this point, the transformation process can be performed (see 4.3). Finally, a process-oriented vision of the approach is provided (see 4.4).

4.1 OWL Archetype Model

The representation of archetypes in OWL requires the semantic interpretation of clinical archetypes. For this, different steps have to be made. First, the CEN ENV13606 reference and archetype models were analyzed to make a semantic interpretation of its information. In fact, the standards' specifications are not expressed in formal manner, and this was a difficulty to achieve our goal. In our semantic interpretation, referential semantics is modeled through semantic relations between the concepts. The semantic interpretation of the Archetype Object Model, whose result is an OWL archetype model, is described next.

Two main entities can be pointed out in archetype semantics: the archetype itself and the archetype terms:

- Archetype: Each archetype represents a clinical concept (i.e., blood pressure). So, this clinical concept has to be defined in the conceptual definition of the archetype. This clinical concept will be of a specific type of clinical item depending on the underlying information model. In our case, the type is one of the included in the CEN reference model. This clinical concept may be built by refining an already existing one. In this case, the new archetype is considered a specialization of the already existing one.
- Archetype terms: The clinical concept has an internal structure. This internal structure is defined by the type of clinical concept defined by the archetype. Each component of this internal structure is represented as a term by the different standards. Each archetype term has a proper internal

structure, so that, it may be composed of different archetype terms as well. In fact, the clinical concept is also an archetype term.

Furthermore, the ontological archetype also contains general information as it is specified in the specification and described in previous sections: Auditory details, archetype description, assertions, translations, and terminological bindings. In this approach, translations are defined at term level, so that, an archetype translation to a specific language is comprised of the set of translations of archetype terms to such language. Therefore, each archetype term has a set of translations associated. Terminologies are basic in biomedical domains, and they represent different ways of coding, representing and classifying biomedical terms. Managing multiple terminologies in the archetypes allow archetype builders for using their own terms without diminishing the standardization of the content by means of terminological links to standardized terminologies such as MEDCIN [47] or SNOMED, or to medical vocabulary resources such as the UMLS metathesaurus [48]. In our OWL modelling, terminological bindings are also defined at archetype term level.

Figure 4 shows the graphical representation of this part of the Archetype Ontology Model. This figure has been obtained by using the Ontoviz Plug-in for Protégé [39] and it shows the context of the concept archetype.

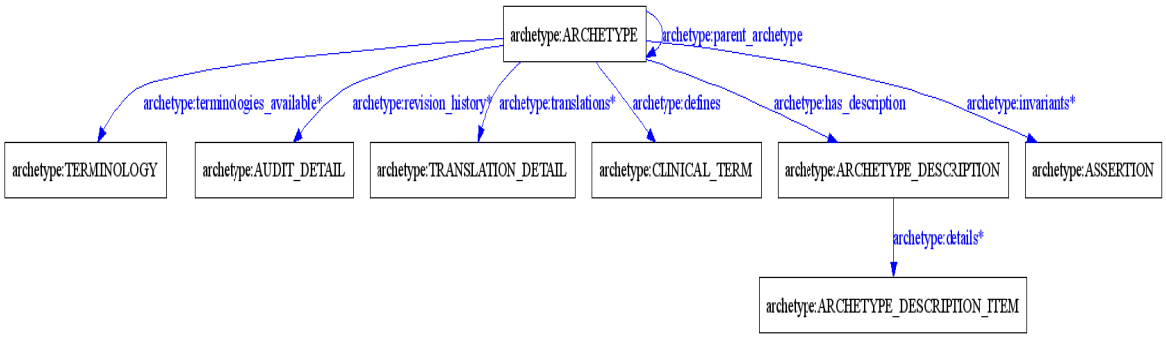

Figure 4.The concept archetype and its context

Let us go into deeper details concerning archetype terms. Archetype terms can refer to restrictions and conceptual entities, so having constraint terms and clinical terms. Like Figure 4, Figure 5 shows the ontological representation of the concept archetype term and its context. This context includes the term definitions, the term translations, the terminological bindings, and the type of clinical item:

Cluster, Element, Section, Entry, Folder, Item or Composition. Constraint terms are also types of constraints. There are other types of constraints accounting for the cardinality of terms having lists of values, the existence of a particular term, and the number of occurrence. Cardinality constraints are only compulsory for terms of types such as lists or sets, whereas every ontology term has an occurrence constraint associated. Each clinical term have also an occurrence constraint associated, accounting for the occurrence of this type of node in the data under the owning term, that is, in the context of its parent archetype term.

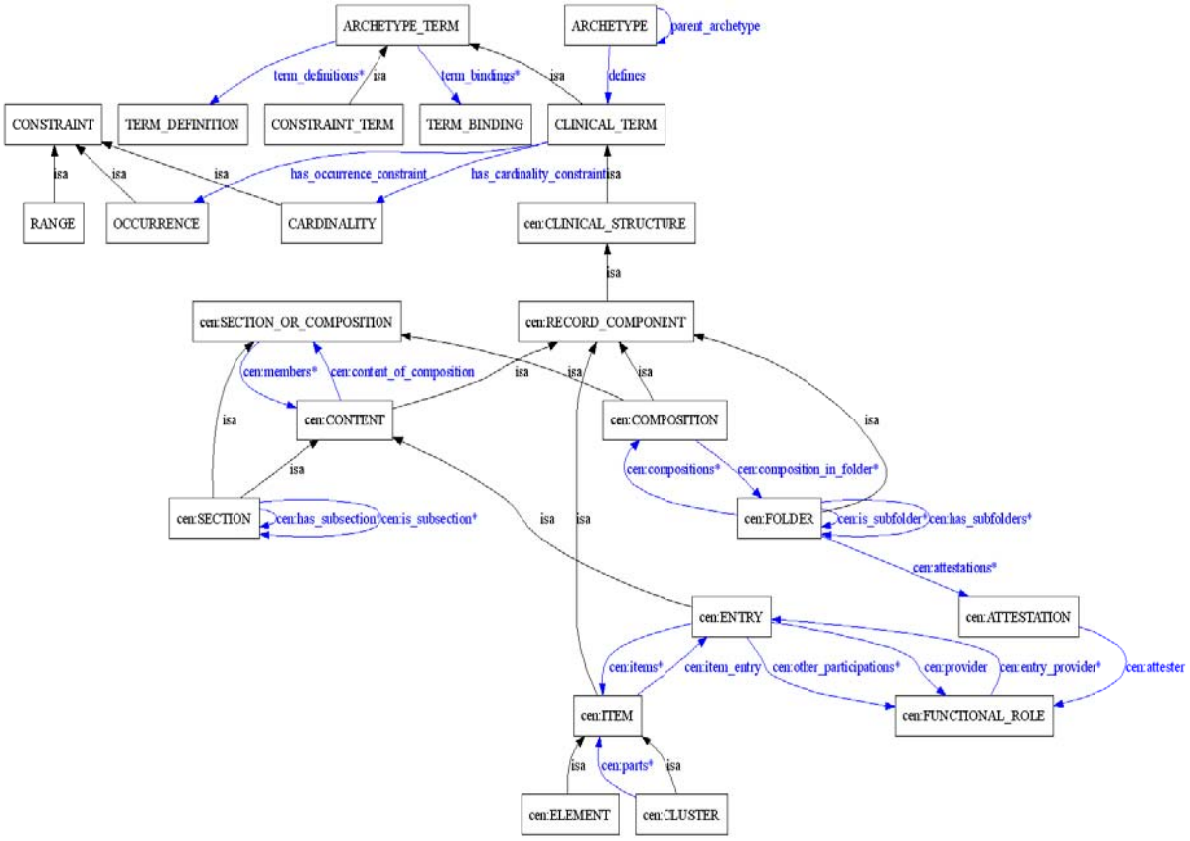

Figure 5. The clinical knowledge in the archetype model ontology: Archetype term

Figure 5 contains some concepts which are not defined in the Archetype Ontology but they are used from a different one. This is the case of all the concepts belonging to the *cen* namespace. All these concepts have been defined in the CEN reference model ontology, which is used by the CEN archetype model ontology. In fact, the development of the ontologies for both the reference and archetype models for CEN has produced three ontologies, which are described next. The reference model contains the definition of business concepts and the building blocks for defining clinical

concepts. The blocks for building clinical concepts, that is, the clinical data structures and data types, are contained in the *CEN-SP* ontology. The complete reference model is contained in the *CEN-RM* ontology. This ontology reuses *CEN-SP* and defines the business objects. The archetype model is defined in the *CEN-AR* ontology, which also reuses *CEN-SP* to build the particular clinical concepts. Table 1 shows the metrics of these ontologies in terms of OWL primitives, that is, classes, datatype properties, object properties and restrictions. These ontologies are available at http://klt.inf.um.es/~poseacle.

Table 1. Metrics of the CEN ontologies

Ontology	**Classes**	**Datatype**	**Object**	**Restrictions**
CEN-SP	64	33	70	124
CEN-RM	80	37	113	146
CEN-AR	120	83	115	272

4.2 Representing ADL and the OWL Archetype Model in MDE

ADL archetypes are written in ADL, which is EHR standard-independent. In order to represent ADL archetypes as models, a metamodel of the abstract syntax of the language is required. The textual concrete syntax of ADL is processed by the xText tool, which is part of the oAW toolkit [44]. This tool implements a bridge between the Grammarware and MDE technical spaces. In the one hand, a metamodel representing the concrete syntax is derived from the grammar. This metamodel is referred in this work as *eADL*. On the other hand, xText generates a parser capable of creating models conforming to the *eADL* metamodel.

The transformation process to OWL instances has to be as generic as possible to easily deal with other EHR standards, like OpenEHR. So, an intermediate representation of the archetype, common to all these standards, could be useful. AOM plays this role in the specifications of the CEN ENV 13606 standard. Its metamodel is formally defined as an XML Schema and is expressed as an Ecore metamodel by means of the Eclipse Modeling Framework (EMF) [42]. The resulting metamodel will be referred in this work as *eAOM*. Thus, a model transformation process is required for expressing

models in *eAOM*. The transformation rules in charge of instantiating the *eAOM* metamodel are written in the model transformation language RubyTL. Figure 6 shows partly the cholesterol archetype represented as an *eAOM* model.

The *eAOM* model provides an intermediate archetype representation to describe the archetype information. It allows to represent the specific elements of a CEN archetype in a generic way and makes possible the following step in our proposed approach, transforming syntactic models to semantic models conforming to CEN ENV13606. In order to establish the mapping between AOM and *CEN-AR*, the ontology has also to be expressed as an Ecore metamodel, *eCEN-AR*. The ODM standard defines the semantics of the transformation of OWL ontologies to models. The Protégé environment implements this transformation, from OWL ontologies to Ecore metamodels.

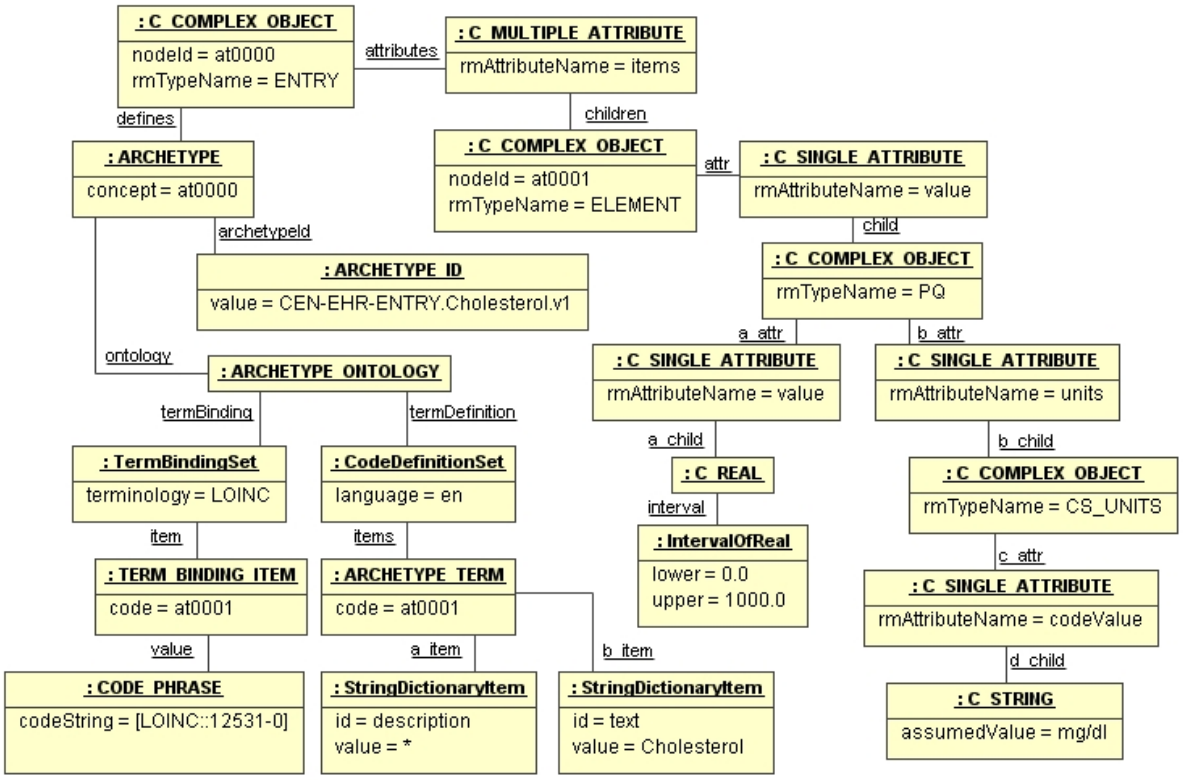

Figure 6. *eAOM* model of the cholesterol archetype.

4.3 The Transformation Process

Once the metamodels have been obtained, correspondences can be defined between *eAOM* and *eCEN-AR* in order to transform ADL content into OWL. The transformation process would be then completed by instantiating OWL archetypes. Let us describe next such stages.

4.3.1 Correspondences eAOM and eCEN-AR metamodels

The AOM representation of archetypes (*eAOM* metamodel) is mapped to the CEN standard representation (*eCEN-AR* metamodel). Translating archetypes from AOM to CEN means to add the specific features of the CEN standard representation to archetypes. Let us briefly describe some of the main correspondences between *eAOM* and *eCEN-AR* through some fragments of the previously mentioned Cholesterol example. A partial representation of both models is shown in Figures 7 and 8.

The root concept in both models is *Archetype*. Let us focus first on its definitional part. In AOM, objects are represented as *C_COMPLEX_OBJECT* and their attributes as *C_MULTIPLE_ATTRIBUTE* or *C_SINGLE_ATTRIBUTE*. For instance, the parsing of the ADL content shown in Figure 1 would produce the following partial *eAOM* model:

- Four *C_COMPLEX_OBJECT* nodes, having the following values for the pair (rmTypeName, nodeId): (1) (“ENTRY”, “at0000”); (2) (“ELEMENT”, “at0001”); (3)(“PQ”,””); (4)(“CS”,””).
- One *C_MULTIPLE_ATTRIBUTE* object and four *C_SINGLE_ATTRIBUTE* objects having the value for (:rmAttributeName): (1)(“items”); (2)(“value”); (3)(“units”); (4)(“codeValue”); (5)(“value”).

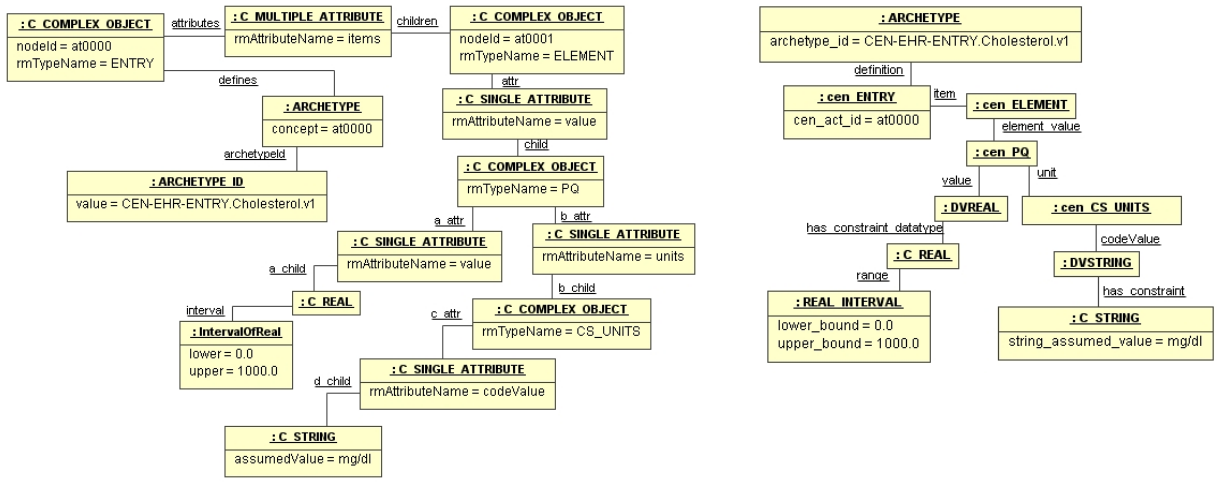

Figure 7. Left. Fragment of *eAOM* model of cholesterol archetype, Right. Fragment of *eCEN-AR* model of cholesterol archetype.

The generic nature of AOM makes it no possible to make explicit the semantics of these objects, and it is embedded into string matching using the attributes *rmTypeName* and *rmAttributeName*. By analyzing the value of these properties, the following mappings to the *eCEN-AR* model can be defined:

- The four *C_COMPLEX_OBJECT* are converted into the following specific elements from the CEN reference model: (1)(cen_ENTRY); (2)(cen_ELEMENT); (3)(cen_PQ); (4)(cen_CS_UNITS).
- The five *C_ATTRIBUTE* are converted into specific attributes of the previous mentioned types from the reference model. A *cen_ENTRY* object has the attribute *cen_items*, a *cen_ELEMENT* the attribute *cen_element_value*, a *cen_PQ* the attributes *cen_units* and *cen_value_real*, and a *cen_CS_UNITS* the attribute *cen_codeValue*.

Let us analyze now the ontology part of the archetype. In this section, the terminological information is provided. There are four major parts in an archetype ontology: term definitions, term bindings, constraint definitions, and constraint bindings. In Figure 8, the former two are shown. Such Figure also includes the ontology section of the cholesterol archetype in both *eAOM* and *eCEN-AR* models.

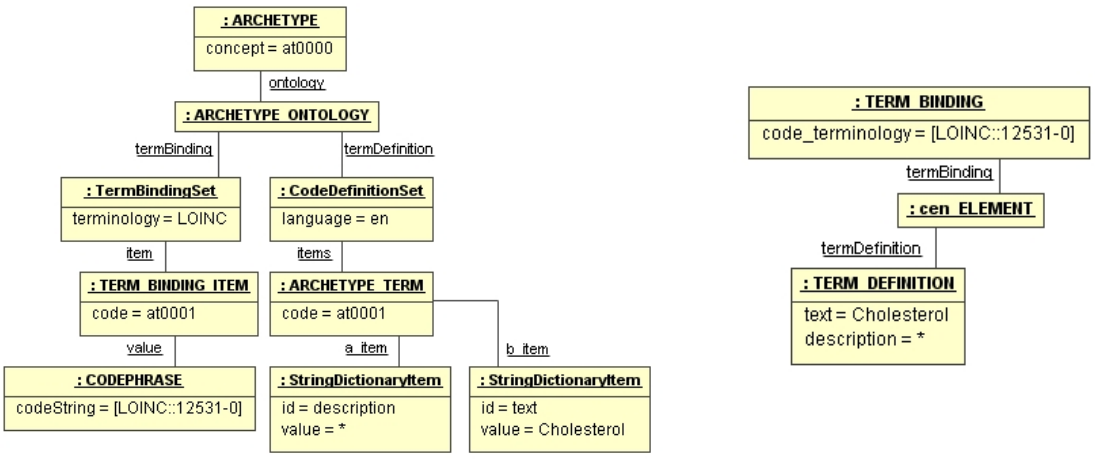

Figure 8: Left. *eAOM* model fragment for the ontology section of the cholesterol archetype, Right: *eCEN-AR* model fragment for the ontology part of the cholesterol archetype.

An archetype has an association, called *ontology*, with the concept *ARCHETYPE_ONTOLOGY*, which has a term definition and a term binding associations. The binding is contained in a *TermBindingSet* as a *TERM_BINDING_ITEM* and the definition in a *CodeDefinitionSet* as an *ARCHETYPE_TERM*. Both are indexed with a unique identifier, which is used within the archetype definition body. In this case these define the meaning and the binding in an external terminology, e.g., the *ELEMENT* at0001. Again, there is no explicit relation between the element and its definition and binding. Such relation must be established by string processing and matching. By mapping *eAOM* concepts to the *eCEN-AR* model, the *cen_ELEMENT* at0001 has a direct association with its definition (*TERM_DEFINITION*) and binding (*TERM_BINDING*).

RubyTL has been used to define these correspondences. This language permits to define a set of transformation rules, that establish the correspondence between objects of the *eAOM* and the *eCEN-AR* metamodels by means of bindings. A binding is a kind of assignment that allows to declare what and not how, needs to be transformed. This language also provides helpers. A helper is a kind of function that allows to define code outside rules, making clearer the code. For instance, Figure 9 shows a fragment of a rule that defines the transformation of a *C_COMPLEX_OBJECT* into an *ELEMENT* object of the *eCEN-AR* metamodel. This rule contains some bindings, which transform *eAOM* into *eCEN-AR* objects and some helpers that return the correct *eAOM* object in the model. In

line 10 of the code of this Figure, getProperty is a helper, and for the cholesterol example, it would return a *C_COMPLEX_OBJECT*(rmTypeName:PQ), in that way the binding will be created between a *C_COMPLEX_OBJECT* and a *Cen_PQ*.

```
(1)  rule 'C_COMPLEX_OBJECT2Cen_ELEMENT' do
(2)  from   eAom::C_COMPLEX_OBJECT
(3)  to      eCen-AR::Cen_ELEMENT
(4)
(5)  filter { |aom| aom.is_ELEMENT? }
(6)  mapping do |aom,cen|
(7)     cen.has_occurrence_constraint = aom.occurrences()
(8)     cen.has_cardinality_constraint = aom.cardinality()
(9)     cen.cen_node_id = aom.nodeId
(10)    cen.cen_value_element = aom.getProperty ("value")
(11)    cen.term_definitions = aom.term_definitions()
(12)    cen.term_bindings = aom.term_bindings()
(13)    ...
(14)  end
(15) end
```

Figure 9. RubyTL transformation rule for cen_ELEMENT

Figure 10 depicts the *eCEN-AR* model for the Cholesterol archetype example as the result of applying the transformation rules. The generic terms of the cholesterol *eAOM* example are now specific terms of the CEN standard, a *C_COMPLEX_OBJECT* is changed now to a *cen_ENTRY*, *cen_ELEMENT*, *cen_PQ* and so on.

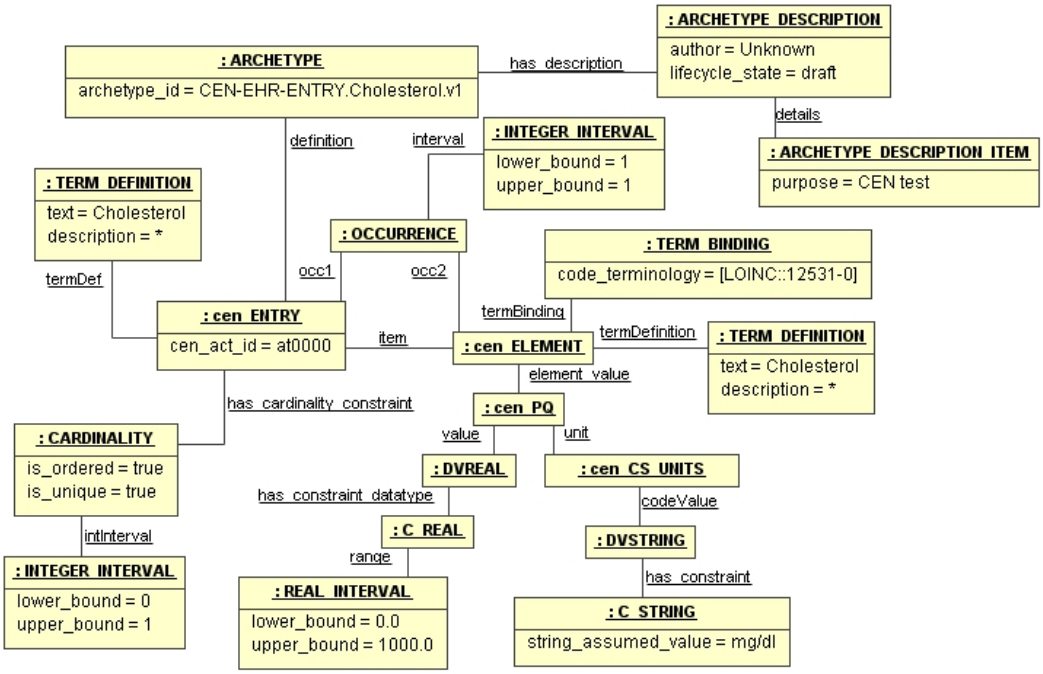

Figure 10. *eCEN-AR* model of the cholesterol archetype.

4.3.2. Instantiation of OWL archetypes

A model conforming to the *eCEN-AR* metamodel conveys the semantic interpretation of the ADL archetype according to CEN standard, but the metamodeling formalism does not allow the semantic exploitation of ADL content. So it is necessary to express models as OWL ontologies. That is the third step in our approach, the instantiation of OWL archetypes. A model-to-text transformation language has been used to generate OWL content from models. The transformation is written in MOFScript template language [43] due to the integration with Eclipse and EMF, and the alignment to the Model2Text OMG standard. The MOFScript language is used to obtain the OWL code from an *eCEN-AR* model. This language allows to define a set of rules, in which static text and imperative sentences can be combined. The imperative constructions allow to control the code generation and to invoke other rules. In our approach a rule for each metaclass in the *eCEN-AR* metamodel is defined. Figure 11 shows the extract of a MOFScript rule that generates the OWL code for the cen_ELEMENT object from the *eCEN-AR* model. As we can noticed in this Figure, there is static text and some sentences invoking the toOwl() rule, lines 5, 10 or 15. This rule has different effects depending on the object it is applied on due to polymorphism. Let us consider line 5 of Figure 11. In case of having an OCCURRENCE object, the corresponding code according to the cholesterol example will be generated as it is shown in Figure 12, lines 10 to 12. As it can be noticed, different invocations generate the different terms of the *CEN-AR* ontology.

```
(1)  ec.cen_ELEMENT::toOwl (){
(2)        '<cen:ELEMENT rdf:ID="'self.cen_node_id'">'nl
(3)        if(self.has_occurrence_constraint != null){
(4)           '<cen-archetype:has_occurrence_constraint>'nl
(5)            self.has_occurrence_constraint.toOwl()
(6)           '<cen-archetype:has_occurrence_constraint>'nl
(7)        }
(8)         if (self.cen_element_value != null){
(9)            '<cen:element_value>'nl
(10)           self.cen_element_value.toOwl()
(11)           '</cen:element_value>'nl
(12)       }
(13)       self.term_definitions->forEach(c:ec.TERM_DEFINITION){
(14)          '<cen-archetype:term_definitions>'nl
(15)           c.toOwl(self.cen_node_id)
(16)         '</cen-archetype:term_definitions>'nl
(17)       }
(18)    ...
(19)       '</cen:ELEMENT>'nl
(20)  }
```

Figure 11. Extract of MOFScript generation rule for cen_ELEMENT

```
(1) <rdf:RDF
(2)     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
(3)     ...
(4)   <owl:Ontology rdf:about="">
(5)       <owl:imports rdf:resource="http://klt.inf.um.es/~poseacle/CEN-AR-v1.0.owl"/>
(6)   </owl:Ontology>
(7)    ...
(8)      <cen:ELEMENT rdf:ID="at0001">
(9)          <cen-archetype:has_occurrence_constraint>
(10)             <cen-archetype:OCCURRENCE rdf:ID="OCCURRENCE_0_1">
(11)                <cen-archetype:interval rdf:ID="#INTEGER_INTERVAL_0_1"/>
(12)             </cen-archetype:OCCURRENCE>
(13)       </cen-archetype:has_occurrence_constraint>
(14)         <cen:element_value>
(15)           ...
(16)       </cen:element_value>
(17)       <cen-archetype:term_definitions>
(18)          ...
(19)      </cen-archetype:term_definitions>
(20)     </cen:ELEMENT>
(21)        ...
(22)      </rdf:RDF>
```

Figure 12. Extract of the resulting OWL cholesterol archetype

4.4 The process-oriented vision of the approach

In the previous subsection, the different steps for transforming ADL content into OWL have been described. Now, the whole process is considered. Figure 13 depicts the global structure of the process of transforming ADL to OWL. The transformation process has an ADL archetype as input and an OWL archetype as output, so starting the process with a syntactic content and obtaining a semantic content. In Figure 13, the internal process is shown into a box, and the workflow is divided into four main steps: (1) obtention of the eADL model for the ADL archetype; (2) obtention of the eAOM model for the ADL model; (3) obtention of the eCEN-AR model for the archetype; and (4) obtention of the OWL archetype. Hence, the input is an ADL archetype belonging to the Grammar technical space and is transformed into an archetype belonging to the Semantic Web technical space through a transformation process in the MDE technical space.

In the figure, the separation of the different conceptual layers can be observed. On the one hand, the internal process is done at metamodel level. At this level the relations Grammar/Metamodel and Ontology/Metamodel can be identified. On the other hand, in the transformation from ADL

archetypes to OWL ones, the interactions are mostly at model level, being clear then the relations between the corresponding models and metamodels.

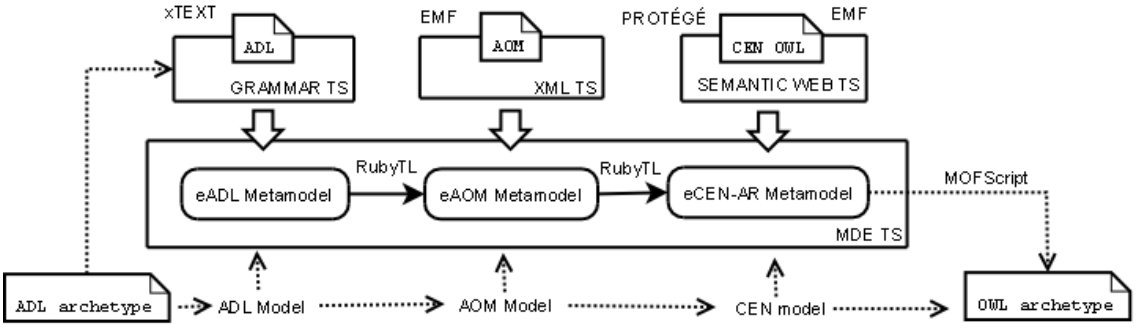

Figure 13. Architecture of the solution

5. DISCUSSION AND CONCLUSIONS

Archetypes facilitates the definition of a semantic layer for common understanding and mutual communication of clinical data structured as a formal clinical concept definition decided by health domain experts, achieving at the same time semantic interoperability among clinical Information Systems. But archetypes are also a valid approach for upgrading already deployed systems in order to make them compatible with an EHR standard, considering the archetypes as clinical data integration components.

Provided that archetypes are considered an important element towards the consecution of semantic interoperability among EHR systems, it seems sensible to compare archetypes and ontologies as representation technologies to discuss whether they can be considered functionally equivalent for such purpose. This discussion does not intent to make a correspondence between archetypes and a particular type of ontology (i.e., top-level, domain, application, and so on) because, in the context of this research, ontologies are more generic than archetypes. Archetypes attempt to harmonize, unify and guide clinical practice by containing consensus knowledge, so containing universally valid content. On its hand, an ontology ideally contains all the existing consensus knowledge of a particular domain, being this knowledge recognized and accepted by the community, so playing both technologies a similar role. It has proven complicated to agree on standard ontologies since different

experts have different points of view on a particular reality, and it is likely to be a problem for archetypes [11].

There have been different proposals for representing shareable clinical information in the last decade, such as the OpenGalen project [56] or GLIF [5]. GALEN proposed a technology to represent shareable clinical information in way capable of reconciling the diversity of needs for terminology, bridging the gap between information required for patient care and for statistical, management and research purposes. GALEN provides, amongst other components, a reference model and a terminology server. The reference model provides a taxonomic classification of medical concepts, as well as its structural information although the dual model approach is more generic. This kind of information is part of the information needed to define clinical archetypes, and it can be useful to define the terminological component of archetypes. The GALEN knowledge is formalized by using the GRAIL language [26], which was developed for formal representation of medical terminologies, allowing for defining concepts and their essential properties, and is a Description Logics language. However, GRAIL has not become a standard such as OWL. On the other hand, GLIF provides a shareable representation of clinical guidelines, including the corresponding workflow, although the purpose of archetypes is not being used as clinical guidelines, as these are currently understood.

This work has been developed in the context of a research project aiming at developing and applying semantic web technologies for managing electronic healthcare records. Therefore, it is expected to integrate this top-down perspective to build archetypes with complementary bottom-up approaches for populating the archetypes built from information currently stored in relational databases. This work is also been carried out in this research project (see for instance [18]). The combination of these works will allow for building interoperable and semantically manageable archetypes and populating them from existing databases. Both works will provide interfaces to different worlds: public external information (OWL archetypes) and internal information (databases). The semantic publication of the contents of the archetypes would be in line with the objectives of the development of the Semantic Web, which targets accessible web contents for both humans and computers so that applications might interoperate semantically in an efficient way. Given the importance of interoperability in the

health domain, having access to the Semantic Web and Semantic Web Services should be considered as necessary.

When representing archetypes in OWL, different decisions have to be made. On the one hand, archetypes can be modeled as classes, because they are models themselves. On the other hand, archetypes can be modeled as instances, because they are an instantiation of the archetype model. The modeling decision depends then on the actual use of the archetype, since both are built from the same reference model by specializing or instantiating a model concept (i.e., cluster, element and so on). Here, the latter approach has been followed, since our main goal is to perform semantic activities at archetype level, so the archetypes are our individuals.

In this work, a methodology for transforming ADL into OWL for CEN archetypes has been followed. This transformation mechanism is different from the proposed in [15]. There, ADL archetypes are mapped into OWL. However, the authors do not perform a semantic interpretation of archetypes but translate ADL expressions into OWL. In [40], the OpenEHR standard has been modeled in OWL, but without making the semantic interpretation.

The solution proposed in this work for representing ADL content in OWL is based on a transformation process through three technical spaces. On the one hand, ADL archetypes are defined by means of a grammar and the result of parsing ADL files is an abstract syntax tree represented as AOM object models. On the other hand, the archetype ontology expressing the semantic interpretation of the CEN standard is represented in OWL. So, the syntactic representation of ADL in Grammarware technical space needs to be transformed into an OWL ontology in Semantic Web technical space. This transformation is performed in the context of MDE technical space due to the availability of mature transformation frameworks and languages to carry out this task.

Bridging Grammarware and MDE technical spaces can be approached from two different perspectives. Grammar-based tools start with a grammar and automatically obtain a metamodel and a parser capable of instantiating models conforming to such a metamodel. This approach is implemented by xText tool [44] using the EMF modeling framework and the Eclipse platform. On the other hand, metamodel-based approaches require the metamodel of the abstract syntax of the

language (i.e. AOM for ADL) and allow for defining a textual concrete syntax for the metamodel. These tools automatically generate a parser to process the concrete syntax of the language and create models conforming to the metamodel. The most remarkable tool in this group is TCS [45] which also works with EMF and Eclipse. The grammar-based approach is appropriate if the grammar of the language is provided and the metamodel of the abstract syntax are not defined, and the metamodel-approach is suitable when the grammar of the language is not established. In our work both the grammar (i.e. ADL) and the metamodel of the abstract syntax (i.e. AOM) are defined. In this case, the maintenance and development cost have been the criteria for using xText against TCS due to the size of the grammar and the metamodel. So, it has been easier the implementation a model transformation from the metamodel obtained from xText to AOM than defining the ADL concrete syntax of the AOM metamodel.

.

The relation of the technical spaces of Semantic Web and MDE is defined in the Ontology Definition Metamodel (ODM) specification [22], supported by OMG. ODM defines the mapping between OWL ontologies and MOF/Ecore metamodels. EODM is an ongoing project intended to implement the ODM specification in EMF. Unfortunately, the EODM implementation is not mature enough to process our archetype ontology. So, alternatives to EODM are necessary to bridge MDE and Semantic Web technical spaces. Protégé allows for generating Ecore metamodels using OWL ontologies, since it implements the mappings defined in the ODM specification. On the other hand, the transformation of models into OWL content needs to be addressed. A first option might be to use existing APIs for building ontologies, such as Jena [55]. However, this approach has not very good maintenance properties and requires a high implementation effort. In this work, the MOFScript template language has been used to generate the XML-like textual representation of OWL from models. The development and maintenance cost of using templates are lower than the API approach. Finally, this is not the first work linking OWL with MDE by using technical spaces, since in [12], a formal approach to make closer MDA (a flavor of MDE) and OWL using the idea of technological spaces can be found. The goal of that work was to contribute to find a suitable MDA-based technique

for the Semantic Web ontologies, so that ontology development process would be closer to software engineers.

Finally, this work is one of the first steps towards the consecution of semantic interoperability among clinical information systems. An advantage of using semantic approaches for such purpose is that they do not require to replace current integration technologies, databases and applications, but to add a new layer that takes advantage of the already existing infrastructure [17;23;34]. We aim at representing clinical information from different EHR architectures in OWL, so that, a semantic interoperability infrastructure can be built by using this semantic layer. Semantic Web technologies have been used not only in medical domains for facilitating interoperability. Recent examples can be found in [7] for obtaining a semantic interoperability infrastructure for E-Government Services and in [13] where many approaches using ontologies for interoperability purposes can be found. In this sense, on-going work is focused on the application of the methodology here presented to OpenEHR and to define the ontological mappings between the CEN and OpenEHR clinical data structures and data types to facilitate the transformation of CEN content into OpenEHR content and viceversa, by using the semantic context provided by the corresponding OWL models.

ACKNOWLEDGEMENTS

This work has been possible thanks to the Spanish Ministry for Science and Education through the projects TSI2007-66575-C02-01, TSI2007-66575-C02-02, FIT-350300-2007-31 and to the Séneca Foundation through the project 05738/PI/07.

REFERENCES

[1] Ashburner, M. C. A. Ball, J.A. Blake, D. Botstein, H. Butler, J. M. Cherry, A.P. Davis, K. Dolinski, S.S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J.C. Matese, J.E. Richardson, M. Ringwald, G. M. Rubin & G. Sherlock (2000) Gene Ontology: tool for the unification of biology. Nature Genetics 25, 25 - 29

[2] Beale, T. (2001). “Archetypes, Constraint-based Domain Models for Future-proof Information Systems”. Available: http://www.deepthought.com.au/it/archetypes/ archetypes.pdf

[3] Berners-Lee, T., Hendler, J., Lassila, O. (2001) The Semantic Web. The Scientific American, May.

[4] Blobel BG (2006) Advanced EHR architectures--promises or reality. Methods of Information in Medicine.45(1):95-101.

[5] Boxwala, A.A., Peleg, M., Tu, S., Ogunyemi, O., Zeng, Q.T., Wang, D., Patel, V.L., Greenes, R.A., Shortliffe, E.H. (2004) GLIF3: a representation format for shareable computer-interpretable clinical practice guidelines. Journal of Biomedical Informatics 37:147-161.

[6] Brewster, C., O’Hara, K., Fuller, S., Wilks, Y., Franconi, E., Musen, M.A., Ellman, J., Buckinham Shum, S. (2004) Knowledge Representation with Ontologies: The Present and Future. IEEE Intelligent Systems vol 19(1): 72-81.

[7] Della Valle, E., Cerizza, D., Celino, I., Estublier, J., Vega, G., Kerrigan, M., Ramírez, J., Villazón, B., Guarrera, P., Zhao, G., Monteleone, G. (2007) SEEMP: A Semantic Interoperability Infrastructure for e-Government Services in the Employment Sector. Lecture Notes in Computer Science 4519: 220-234.

[8] Dogac, A., Laleci, G., Kirbas S., Kabak Y., Sinir S., Yildiz A. Gurcan Y. (2006) Artemis: Deploying Semantically Enriched Web Services in the Healthcare Domain", Information Systems Journal, 31(4-5): 321-339.

[9] Eichelberg,M., Aden, T., Riesmeier,J., Dogac, A., Laleci, G.B. (2006). Electronic Health Record Standards- A brief overview. 4th International Conference on Information and Communications Technology, Cairo, Egypt, December 2006

[10] Fernández-Breis, J.T., Martínez-Béjar,R. (2002) A Cooperative Framework for Integrating Ontologies. International Journal of Human-Computer Studies, 56(6):662-717

[11] Garde S, Knaup P, Hovenga EJS, Heard S. (2007) Towards semantic interoperability for electronic health records. Methods of information in medicine 46(2):332-343.

[12] Gasevic, D. Djuric,D. Devedzic, V., Damjanovic, V (2004) Approaching OWL and MDA Through Technological Spaces, Third Workshop in Software Model Engineering-WiSME2004.

[13] Gonçalves, R.J.; Müller, J.P.; Mertins, K.; Zelm, M. (Eds.). (2007) Enterprise Interoperability II: New Challenges and Approaches. Springer

[14] Gruber, T. R. (1993). A translation approach to portable ontology specifications. Knowledge Acquisition Vol. 5, pp.199-220.

[15] Kilic, O., Bicer, V., Dogac, A. (2005) Mapping Archetypes to OWL. Technical Report

[16] Kurtev, I., Bzivin, J., Aksit,M. (2002) *Technological Spaces: an Initial Appraisal*, CoopIS, DOA'2002.

[17] Linthicum, D. (2004) Leveraging Ontologies: The Intersection of Data Integration and Business Intelligence Part I . DMR Review Magazine, June.

[18] Moner, D., Maldonado, J.A., Bosca, D., Fernández-Breis, J.T., Angulo,C., Crespo, P., Vivancos-Vicente, P.J., Robles,M. (2006) Archetype-Based Semantic Integration and Standardization of Clinical Data. 28th Anual International Conference IEEE Engineering in Medicine and Biology, New York, EEUU.

[19] Nardon FB, Moura LA. Knowledge sharing and information integration in healthcare using ontologies and deductive databases.Medinfo. 2004;11(Pt 1):62-6.

[20] Object Management Group: MOF QVT Final Adopted Specification, 2005. http://www.omg.org/docs/ptc/05-11-01.pdf

[21] OMG (2006) Meta Object Facility (MOF) 2.0 Core Specification, OMG Document formal/2006-01-01, http://www.omg.org/cgi-bin/doc?formal/2006-01-01.

[22] Ontology Metamodel Definition Specification (2006) . OMG Documentformal/2006-05-01. http://www.omg.org/cgi-bin/doc?ad/2006-05-01.pdf

[23] Partridge, C. (2002) The Role of Ontology in Semantic Integration

[24] Paterson, G.I. (2004) Semantic Interoperability for Decision Support Using Case Formalism and Controlled Vocabulary. Health'04.

[25] Rector, A.L. (1999). Clinical Terminology: Why is it so hard?. Methods of Information in Medicine 6, pp. 245:251.

[26] Rector AL, Bechhofer S, Goble CA, Horrocks I, Nowlan WA, Solomon WD. The GRAIL concept modelling language for medical terminology. Artif Intell Med. 1997 Feb; 9(2): 139-71.

[27] Rose JS, Fisch BJ, Hogan WR, Levy B, Marshal P, Thomas DR, Kirkley D. (2001) Common medical terminology comes of age, Part One: Standard language improves healthcare quality. Journal of Healthcare Information Management, Fall;15(3):307-18.

[28] Sánchez-Cuadrado, J., García-Molina, J, Menárguez-Tortosa, M. (2006) RubyTL: A Practical, Extensible Transformation Language. ECMDA-FA 2006. http://rubytl.rubyforge.org/

[29] Schulz S, Hahn U (2005) Part-whole representation and reasoning in formal biomedical ontologies. Artificial Intelligence in Medicine 34 (3) : 179-200

[30] Semantic Health (2006) Semantic Health Final Report. http://www.semantichealth.org/DELIVERABLES/SemanticHEALTH_D1_1_finalC.pdf

[31] Semantic Interoperability Community of Practice (2005). White Paper Series Module 1: Introducing Semantic Technologies and the Vision of the Semantic Web

[32] Smith B, Ceusters W. (2005) An ontology-based methodology for the migration of biomedical terminologies to electronic health records.AMIA Annual Symposium Proceedings:704-708.

[33] Smith, B. (2006) “From Concepts to Clinical Reality: An Essay on the Benchmarking of Biomedical Terminologies”, Journal of Biomedical Informatics, 39(3): 288-298.

[34] Stuckenschmidt, H., Wache, H., Visser, U., Schuster, G. (2001) Methodologies for Ontology-based Semantic Translation. ECIMF.

[35] Van Heijst, G., A. T. Schreiber, & B. J. Wielinga (1997) ‘Using explicit ontologies in KBS development’. International Journal of Human-Computer Studies, 45, 183-292.

[36] http://colab.cim3.net/cgi-bin/wiki.pl?SICoP

[37] http://obo.sourceforge.net

[38] http://ontolog.cim3.net/

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

[40] http://trajano.us.es/~isabel/

[41] http://www.centc251.org

[42] http://www.eclipse.org/emf/

[43] http://www.eclipse.org/gmt/mofscript/

[44] http://www.eclipse.org/gmt/oaw

[45] http://www.eclipse.org/gmt/tcs/

[46] http://www.hl7.org

[47] http://www.medicomp.com

[48] http://www.nlm.nih.gov/research/umls/

[49] http://www.omg.org

[50] http://www.openehr.org

[51] http://www.regenstrief.org/medinformatics/loinc/

[52] http://www.snomed.org/

[53] http://www.w3.org/TR/owl-ref/

[54] http://www.webont.org/owl/1.1/

[55] http://jena.sourceforge.net/

[56] http://www.opengalen.org/