A model-driven approach for representing clinical archetypes for Semantic Web environments

Catalina Martínez-Costa, Marcos Menárguez-Tortosa, Jesualdo Tomás Fernández-Breis, José Alberto Maldonado

* Corresponding author. Fax: +34 968364151.
E-mail address: jfernand@um.es (J.T. Fernández-Breis).

ARTICLE INFO
Article history:
Received 14 March 2008
Available online 23 May 2008

Keywords:
Biomedical informatics
Electronic Healthcare Records
Archetypes
Semantic Web
Ontology
Model-driven Engineering

ABSTRACT
The life-long 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 the Archetype Definition Language (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 the Ontology Web Language (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 EN13606 EHR architecture.

© 2008 Elsevier Inc. All rights reserved.

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 [26], not only is medicine domain big, for example, SNOMED-CT [60] contains around 350,000 atomic concepts, but also open-ended because new information, finer grained details or new relationships are always being discovered or becoming relevant.

As a consequence, the list of medical concepts can never be complete. This implies that a traditional information model will never be completely adapted to the clinical requirements and its continuous evolution [2]. 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 ENV13606 standard, proposed by the CEN/TC251, Technical Committee 251 of the Normalization European Committee [48], 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) [57]. 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. The syntactic orientation and limitations of ADL makes the achievement of such goals more difficult, as it will be described in Section 3. 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. This work has been done in the context of the POSEACLE research project, and it aims at providing mechanisms for representing archetypes in Semantic Web-manageable manner, and the proposed solution combines Semantic Web and Model-driven Engineering technologies 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 Record (EHR) standards

Nowadays, there are different advanced standards and architectures [4,9] for representing and communicating Electronic Healthcare Records, such as HL7 [53], OpenEHR [57] and the CEN ENV13606 [48] 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) archetype model. 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 valid information 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 ENV13606 [48] is addressed. The CEN ENV13606 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 information 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 settings, 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. This would account for all the information a medical doctor should include when (s)he is evaluating a genetic condition of a patient. 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 are usually built by domain experts, so they are based on clinical knowledge, 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 medical domain is a dynamic environment, which has continuous research clinical results, so 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.
Fig. 1 shows an extract of an ADL archetype for Cholesterol according to the CEN standard. It should be noted that this archetype does not define the concept Cholesterol, but how this must be measured for a patient. In the figure, 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 the measurement of 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 [0.0,1000.0], and is measured in mg/ml. Finally, the ontology section includes the terminological definition and bindings. Here, the linguistic expressions associated to at0000 and at0001 are provided as well as a binding for at0001 in the external terminology LOINC [59]. They are the term Cholesterol in the three cases. For instance, the link to the LOINC term means that the ELEMENT is related to the medical concept Cholesterol defined in such terminology. This section might lead to confusion, since no ontology is really defined in it. 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.

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

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 avail-

![Fig. 1. Extract of an ADL archetype.](image-url)
able 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, among which the cornerstone technology is the ontology.

In the literature, multiple definitions for ontology can be found (see for instance [15,38]). 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, conceptual relations, and axioms related to the previous elements. In practical settings, ontologies have become widely used due to the advantages they have (see for instance [11]). 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 [32,36]). One of the most significant advances in bioinformatics was the development of the Gene Ontology [1]. In fact, the amount of bio-ontologies and related projects (e.g., the Open Biomedical Ontologies project [41]) is increasing. All these applications 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 [20,30,35]). 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 [33] 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 representing clinical information (see [25,34,39,42]).

In this work, ontologies are proposed to represent clinical information semantics. The ontologies will be modeled by using the Ontology Web Language (OWL) [61], 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. Models allow for saving time and resources in software maintenance and development, increasing 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. The Object Management Group (OMG) [56] defines a four-level meta-modeling architecture [22]. 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 [22] or Ecore [49] 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 meta-modeling language. The QVT specification [21] allows for defining model transformations but at present there are not mature tools that implement the specification. In recent years, several model transformation languages have been defined. RubyTL [31] is a rule-based hybrid transformation language for defining transformation rules in both declarative and imperative ways and includes significant features such as the organization of 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 [17]. 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 [51] for generating metamodels from grammars and parsers which instantiate models conforming to these metamodels. In addition, ontologies can be transformed into models by following the Ontology Definition Metamodel standard [23].

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, playing clinical archetypes an important role. 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, such as CEN. 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 reference model to guarantee the clinical correctness of the ADL content. 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 Fig. 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 attribute name 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 Fig. 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, among 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 constrain 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 is shown in Fig. 3.

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 sections 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 reference model but to knowledge inconsistencies between different archetypes descriptions used in different healthcare institutions.

Archetypes are used for guiding clinical practice, so that, they are also a tool for supporting the staff. Semantic Web management approaches might help to find the appropriate archetype for a particular situation. This may also apply to the selection of the set of archetypes to be used for building a particular system. Hence, activities such as comparison, selection, classification and consistency checking 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 [28,29]. In fact, the archetype community is aware of the usefulness of Semantic Web technologies for classification. For instance, in [12] 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-CT [60], are currently in the process of adapting their representation to Semantic Web environments, so that OWL-based archetype construction approach might guarantee the consistency of the knowledge, which cannot be granted by ADL.

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 Section 4.1). Provided that the Model-driven Engineering technical space is used as the pivotal space for performing the transformation, bridges from ADL and OWL to Model-driven Engineering have to be built (see Section 4.2). At this point, the transformation process can be performed (see Section 4.3). Finally, a process-oriented vision of the approach is provided (see Section 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 specifications of the standard 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:

- **Archetype**: Each archetype represents a clinical concept (i.e., measurement of 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 [54] or SNOMED [60], or to medical vocabulary resources such as the UMLS metathesaurus [55]. In our OWL modeling, terminological bindings are also defined at archetype term level.

Fig. 4 shows the graphical representation of this part of the Archetype Ontology Model, and it shows the context of the concept archetype.

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 Figs. 4 and 5 show 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.

Fig. 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 clinical data structures and data types used for building such concepts 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. The current release contains necessary conditions, whereas necessary and sufficient conditions will be included in the next one. This will be helpful to support reasoning processes. These ontologies are available at http://klt.inf.um.es/~poseacle.

4.2. Representing ADL and the OWL archetype model in the Model-driven Engineering technical space

In order to represent ADL archetypes as models, a metamodel of the abstract syntax of the language is required. The grammar of ADL is processed by the xText tool, which is part of the oAW toolkit [51]. This tool implements a bridge between the Grammar and Model-driven Engineering technical spaces. In the one hand, an
Ecore 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 ENV13606 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) [49]. 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. Fig. 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 for representing 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 Ontology Definition Metamodel [23] standard defines the semantics of the transformation of OWL ontologies to models. The Protégé environment [45] implements this transformation, from OWL ontologies to Ecore metamodels.

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 between 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

![Fig. 5. The clinical knowledge in the archetype model ontology: archetype term.](image)
and eCEN-AR through some fragments of the previously mentioned Cholesterol example. A partial representation of both models is shown in Figs. 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 Fig. 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", "at0002"); (4) ("CS", "at0003").

Fig. 6. eAOM model of the cholesterol archetype.

Fig. 7. (Left) Fragment of eAOM model of cholesterol archetype, (right) Fragment of eCEN-AR model of cholesterol archetype.
By analyzing the value of these properties, the following mappings between the element and its definition and binding. The binding is contained in a TermBindingSet as a bindingSet as a ConstraintBindingSet as an ARCHETYPE_ONTOLOGY as a TERM_BINDING.

- 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").

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, interpreting here ontology as in ADL, that is, the terminological information. There are four major parts in an archetype ontology: term definitions, term bindings, constraint definitions, and constraint bindings. In Fig. 8, the former two are shown. Such figure also includes the ontology section of the cholesterol archetype in both eAOM and eCEN-AR models.

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, Fig. 9 shows a fragment of a rule that defines the transformation of a C_COMPLEX_OBJECT into a cen_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, helper 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 C_PQ.

Fig. 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; in this example, C_COMPLEX_OBJECTs have been transformed into cen_ENTRY, cen_ELEMENT, cen_PQ and so on.

(1) rule 'C_COMPLEX_OBJECT?cen_ELEMENT' do (2) from eAOM::C_COMPLEX_OBJECT (3) to eCEN-AR::Cen_ELEMENT (4) (5) filter { lambda { 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.termBindings = aom.term_bindings() (13) ... (14) end (15) end

Fig. 9. RubyTL transformation rule for cen_ELEMENT.

Fig. 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.
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 meta-modeling 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 [50] due to the integration with the Eclipse platform and the Eclipse Modeling Framework, 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 for defining 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. Fig. 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 Fig. 11. In case of having an OCCURRENCE object, the corresponding code according to the cholesterol example will be generated as it is shown in Fig. 12, lines 10–12. As it can be noticed, different invocations generate the different terms of the CEN-AR ontology.

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. Fig. 13 depicts the global structure of the process of transforming ADL to OWL. The rectangles show the different technical spaces that are involved. In the bigger one, Model-driven Engineering, the transformation is performed. The process has an ADL archetype as input and an OWL archetype as output. In this way, the process starts with syntactic content and obtains semantic content. The broken lines point out the path followed by the instance of the ADL archetype until being transformed into OWL. The workflow is divided into four main steps:

(1) Obtention of the eADL model through the xText tool.
Obtention of the eAOM model. For this purpose, the eAOM metamodel has been generated with the Eclipse Modeling Framework (EMF) and the model transformation carried out by mean of the RubyTL language.

Obtention of the eCEN-AR model. First, the eCEN-AR metamodel is created with Protégé and the Eclipse Modeling Framework. Next, the corresponding RubyTL transformations are applied.

Obtention of the OWL archetype through the model to text transformation language, MOFScript. Hence, the input is an ADL archetype belonging to the Grammar technical space and it is transformed into an archetype belonging to the Semantic Web technical space through a transformation process in the Model-driven Engineering 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.

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 intend 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...
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 [12].

There have been different proposals for representing shareable clinical information in the last decade, such as the OpenGalén project [58] or GLIF [5]. GALEN proposed a technology to represent shareable clinical information in a 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, among 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 [27], 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 [5] 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 such 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 has also been carried out in this research project (see for instance [19]). 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 one proposed in [16]. 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 [47], 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 an Archetype Object Model (AOM). 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 the Grammar technical space needs to be transformed into an OWL ontology in the Semantic Web technical space. A first option might be to use an ADL parser [46] and existing APIs for building ontologies, such as Jena [40]. However, this approach has poor maintenance properties and requires a high implementation effort. In this work, this transformation is performed in the context of Model-driven Engineering technical space due to two main reasons: (1) the availability of mature transformation frameworks and languages to carry out this task, reducing the implementation effort and simplifying maintenance; and (2) the availability of bridges to other technical spaces, such as Grammareware and Semantic Web.

Bridging Grammar and Model-driven Engineering 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 [51] using the Eclipse 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 [52] which also works with the Eclipse platform. 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 has 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 Model-driven Engineering is defined in the Ontology Definition Metamodel (ODM) specification [23], supported by OMG. ODM defines mappings between OWL ontologies and several metamodels, such as UML. EODM is an on-going project intended to implement the ODM specification in the Eclipse Modeling Framework. Unfortunately, the EODM implementation is not mature enough to process our archetype ontology. So, alternatives to EODM are necessary to bridge Model-driven Engineering and Semantic Web technical spaces. Protégé [45] allows for generating Ecore metamodels using OWL ontologies. On the other hand, the transformation of models into OWL content needs to be addressed. In this work, the MOF-Script template language has been used to generate the XML-like textual representation of OWL from models. Finally, this is not the first work linking OWL with Model-driven Engineering by using technical spaces, since in [13], a formal approach to make closer Model-driven Architecture (MDA), a flavor of Model-driven Engineering, and OWL using the idea of technical 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.

Providing an OWL representation for archetypes allows for carrying out semantic activities, such as comparison, classification, selection, and consistency checking more efficiently. This representation is used in the semantic archetype management system that is currently being developed in the context of the POSEACLE project. This system will allow for annotating archetypes and perform...
different types of semantic searches on virtual archetype repositories.

The availability of reasoners for OWL-DL is an advantage of such representation. To date, reasoners such as Pellet [44] and FACT++ [43] have been used by us to check for the correctness and consistency of the OWL archetypes obtained as a result of the transformation process. They will also be used for supporting classification mechanisms in the above-mentioned management system.

To date, manually constructed ADL archetypes for CEN have been used, these being adaptations of existing ones for OpenEHR. The OpenEHR standard is also dealt in our research project, which also pursues the semantic interoperability between CEN and OpenEHR-based 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 [18,24,37]. 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 [14] 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 vice versa, by using the semantic context provided by the corresponding OWL models.

Acknowledgments

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 Science and Education through the projects TSI2007-66575-C02-02-02, FIT-350300-2007-31 and to the Séneca Foundation through the project 05738/P1/07.

References

Available from: http://www.opengalen.org/.
Available from: http://www.snomed.org/.
Available from: http://www.w3.org/TR/owl-ref/.
Available from: http://www.webont.org/owl/1.1/.