Dynamic Generation of Archetype-Based User Interfaces for Queries on Electronic Health Record Databases

Shelly Sachdeva, Daigo Yaginuma, Wanming Chu, and Subhash Bhalla

Graduate Department of Computer and Information Systems,
University of Aizu, Fukushima, Japan
{d8111107,m5151102,w-chu,bhalla}@u-aizu.ac.jp

Abstract. Standardized Electronic Health Records (EHRs) make use of archetypes for representation of data. In combination with terminologies, the archetypes enable powerful possibilities for semantic querying of repository data. Such querying enables longitudinal processing of health data, regardless of the originating system. The semantics of data is better understood by viewing the data in the context of the user interface (UI). The paper demonstrates the feasibility of creating a query interface. It introduces a general purpose database transformation channel. It will shorten the application development process and increase the quality of the software by automatically generating software artifacts that are often made manually (and are prone to errors). It is possible to know the locations of each leaf datum within information conforming to an archetype. The tool helps in the inspection of an archetype in advance, which can yield a set of path fragments. It can be used to query instances which conform to an archetype for intelligent querying.

Keywords: Electronic Health Records, Querying, User-Interfaces, Improving Information Quality, Archetype-Based EHR.

1 Introduction

Electronic Health Records (EHRs) are the paperless solution to healthcare world that runs on a chain of paper files. These former Electronic Medical Records (EMRs) have bad record design and shallow support of user interfaces to fill in and extract data, leading to incomplete and incorrect data records. In contrast, the standardized EHR record design is based on clinical investigator recording process [5]. These ensure Data Quality (DQ). Its two key principles are data accuracy and data validity. To communicate effectively, data must be valid and conform to an expected range of values. To be useful, data must be accurate. As the recording of data is subject to human error, there need to be built-in control measures to eliminate errors, in manual recording and computer entry. DQ also includes reliability, completeness, legibility, timeliness, accessibility, usefulness, confidentiality and security [27]. Data quality is proportionate to the attainment of achievable improvements in health care.

Thus, the proposed Electronic Health Records (EHRs) have a complex structure that may include data of about 100-200 parameters, such as temperature, blood-pressure, heart rate and body mass index. Individual parameters have their own

S. Kikuchi et al. (Eds.): DNIS 2011, LNCS 7108, pp. 109–125, 2011.

contents (Figure 1). In order to serve as an information interchange platform, EHRs use archetypes to accommodate various forms of contents [16]. The components within EHR data have multitude of representations. The contents can be structured, semi-structured or unstructured, or a mixture of all three. These can be plain text, coded text, paragraphs, measured quantities with values and units, date, time, date-time, and partial date/time, encapsulated data (multimedia, parsable content), basic types (such as boolean, state variable), container types (list, set) and uniform resource identifiers (URI).

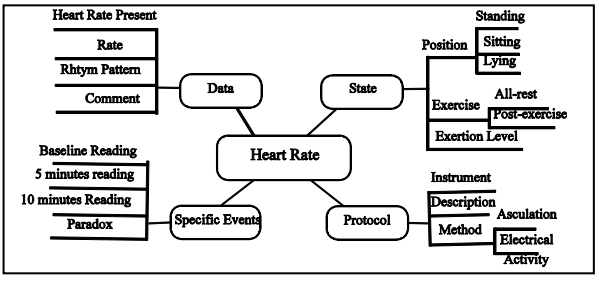

Fig. 1. Heart Rate Archetype

1.1 Archetype-Based EHRs

Standardized EHRs provided by openEHR [24], HL7 [13] and CEN TC/251[10] aims to use archetypes. Archetypes allow describing specific clinical concepts (for example, blood pressure, and ECG measurements) as constraint rules that constrain the possible types, relationships and values of the record components in an ISO 13606 'Composition' [15]. A 'Composition' corresponds to one clinical document. Thus, archetype is an agreed formal and interoperable specification of a re-usable clinical data set which underpins an electronic health record (EHR). It captures as much information about a particular and discrete clinical concept as possible. An example of a simple archetype is WEIGHT, which can be used in multiple places, wherever is required within an EHR. ISO 13606-2 [16] has defined an Archetype Definition Language (ADL) [16], [6], i. e., a formal language that is related to the ISO 13606-1 [15] reference model. Archetypes expressed in this language will be convertible to HL7 Refined Message Information Models (R-MIMs) and Common Message Element Types (CMETs). It is intended to harmonize the ISO/openEHR 'archetype' concept with the HL7 Clinical Document Architecture [14] and HL7 Templates.

An archetype definition basically consists of three parts: descriptive data, constraint rules and ontological definitions (Figure 2). The descriptive data contains a unique identifier for the archetype, a machine-readable code describing the clinical concept modeled by the archetype and various metadata such as author, version, and purpose. The constraint rules are the core of the archetype and define restrictions on the valid structure, cardinality and content of EHR record component instances complying with the archetype (represented by definition section). The ontological part defines the controlled vocabulary (i. e., machine readable codes) that can be used in

specific places in instances of the archetype. It may contain language translations of code meanings and bindings from the local code values used within the archetype to external vocabularies such as SNOMED or LOINC. It may also define additional constraints on the relationship between coded entries in the archetype based on the code value. The ADL for the archetype 'heart rate' is given in Figure 6. ADL is path addressable in a similar way, as data in XML.

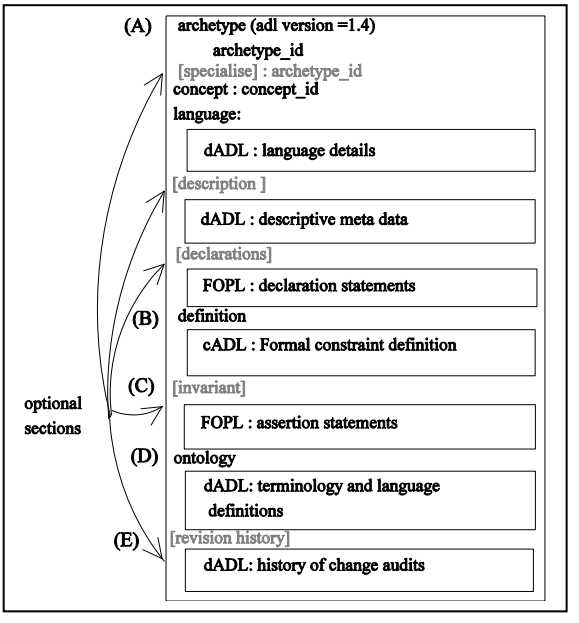

Fig. 2. Structure of Archetype

At the storage level, the EHR components use a Reference Model (RM) [5] for the physical structure. The archetype defines constraints on the structure, types and values of instances of the RM. It enables archetypes to have different granularity levels according to the different classes defined in the RM. Thus, there are different categories of archetypes such as 'Composition', 'Section', 'Entry' and 'Item-Structure'. Each category may have a different structure. Archetypes are stored in common agreed repositories that allow both sides, source and receptor, to identify the semantic links and relationships between the concepts included within an EHR Extract. EuroRec[1] requirements include recommendations for repositories [11]. There are the public archetypes repositories [9] across the world such as openEHR Clinical Knowledge Manager (CKM), NHS repository, Minas Gerais repository and Swedish CKM.

The current research indicates the need for building user interfaces (UI) on the top of qualitatively designed archetypes, thus helping to obtain correct and

[1] The European Institute for Health Records or EuroRec Institute [11] is a non-profit organization that promotes the quality of EHR systems.

complete records. The rest of paper is organized as follows. Section 2 describes the context of archetypes related to improving quality aspects. Section 3 describes querying EHR data with emphasis on high-level user-interfaces for semi-skilled users. The dynamic generation of a user-interface based on archetype is discussed in Section 4. Section 5 describes the system configurations. Section 6 presents related work and discussions. Finally, section 7 presents the summary and conclusions.

2 User- Interfaces for Improving Information Quality

2.1 Motivation and Background

Often, the data quality problem is actually a data misinterpretation problem [21]. The data source may not have any "error," within the data that it provided. However, the content may not convey the meaning that the receiver expected. The issue is about how data in one context can be used in a different context. This is a desirable goal. It is sought through standardization efforts [21]. The current research addresses the problem of semantic heterogeneity in EHR domain. The archetypes created through domain knowledge governance and developing technologies, provide data that is consistent with receiver preference (service layer), thereby improving the data quality at the receiver end. A prototype query interface has been implemented at the service layer for the EHRs that follow the openEHR standard.

2.2 Improving Data and Information Quality

Table 1 gives various DQ requirements [37] and their aspects in archetypes. High quality archetypes with high quality clinical content are the key to semantic interoperability of clinical systems [12]. They aid in decision support. Archetypes may define compositional relationships to other archetypes by using 'Slots'. A 'Slot' sets constraints in the archetype nodes to define which archetypes can be allowed or excluded. These increase the reusability of archetypes because these follow the definition of hierarchical structures between archetypes. Terminologies also help in achieving semantic interoperability. Terms within archetypes are linked (bound) to external terminologies like SNOMED-CT [32]. Thus, archetypes have been chosen in current study for resolving the problem of semantic heterogeneity in EHR domain.

3 Intelligent Querying through Archetypes

Considering the challenges faced by today's health record systems (the need to record more data, the need to analyse more data and the need to share more data), querying plays a vital role in enhancing the information quality of EHR systems. The hindrances to querying are detailed knowledge of schema and specialized knowledge of query language.

Table 1. Data Quality Requirements and Aspects

Data Quality Requirements	Aspects in Archetypes
Accuracy and Validity	Business rules defined in archetypes
Believability	Archetypes developed by domain experts
Accessibility	Sharable Archetypes (through common repository)
Timeliness	New and modified archetypes are developed as clinical knowledge expands
Completeness	Standardized data definitions, content and structure
Interpretability	Fine granularity of data in archetypes and Linkage to terminology standards
Ease of Understanding	Translatable to different languages without language primacy
Concise Representation	Rich health data definitions
Consistency	Ontology-based archetype transformation process (e.g. openEHR archetypes to HL7 CDA archetypes or CEN 13606 archetypes)

The Agency for Healthcare Research and Quality (AHRQ) has found low EHR adoption rates for physician groups [2]. Need of high-level query language interface (as a need) has been identified for improving information quality gains and ease of access in EHR domain [29]. In order to achieve information quality, the current research identifies the use of generated user interfaces for the purpose of querying. It allows querying data at the finest level of granularity (providing flexibility).

3.1 Schema Analysis

EHRs allow multiple representations [5]. In principle, EHRs can be represented as relational structures (governed by an object/relational mapping layer), and in various XML storage representations.

The schema of the EHR consists of set of archetypes along with their parameters and the relationships they have with one another (Figure 3). Archetypes in our data model correspond to an entity set. The parameters of the archetype include not only simple and multi-valued attributes, but also complex-typed attributes. Unlike ER model, our model does not support relationship attributes nor does it distinguish between strong and weak attributes.

EHR has a hierarchical structure. An archetypable data instance in an openEHR standard based EHR is either a ‘composition’, or a ‘section’ (which must be contained in a composition or section), or an ‘entry’ (which must be contained in a composition or section), or an ‘item structure’ (which must be contained within an entry or item structure). The hierarchical structure of EHR and categories of archetypes based on openEHR standard is shown in Figure 4.

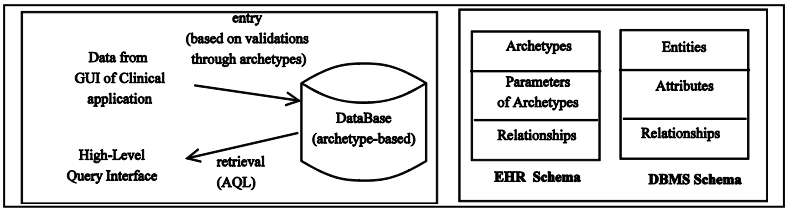

Fig. 3. Querying archetype-based EHR based on analogy of EHR Schema and DBMS Schema

The archetype paradigm (containing categories) is more flexible and easily scalable because it provides the means to handle the knowledge evolution. This technology avoids reimplementation of systems, migrating databases and allows the creation of future-proof systems. Also, these categories are based on the clinical investigator recording process [5], hence improving data quality improvement.

In order to create a data instance of a parameter of EHR, we need different archetypes in ADL, and also these archetypes may belong to different categories of archetypes. For example, to create a data instance for Heart Rate, we need two different archetypes-namely, encounter and heart rate. These archetypes belong to different categories viz., 'composition' and 'observation' (having different structure). At the time of query, a user faces this problem- which archetypes must be included in querying?

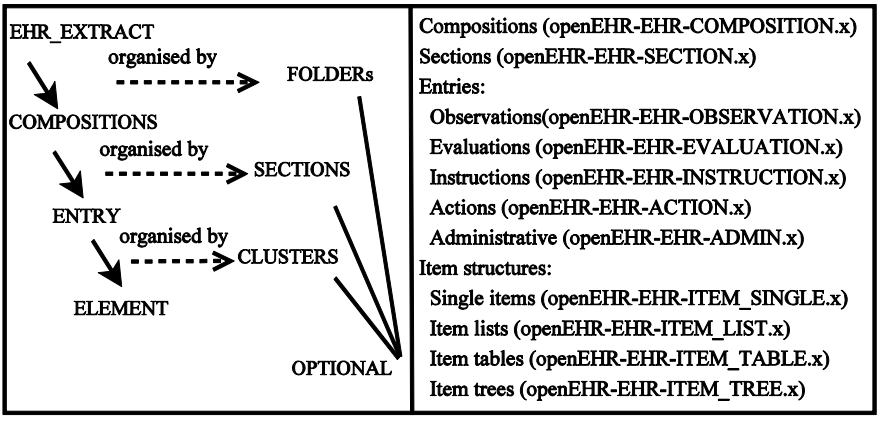

Fig. 4. Hierarchical structure of EHR and categories of archetypes

3.2 Data Analysis

Relatively few archetypes are used to construct quite large slabs of data; a clinical example would be 'ECG results', where one archetype corresponds to 10 leads' worth of time-series data, with possibly hundreds of samples. Consider a simple EHR of health encounter consisting of SOAP (subjective, objective, assessment, plan), which

further consists of blood_pressure and heart_rate. The instance data of such an EHR is very large. The archetype map is much smaller than its data as shown in Figure 5. It is a suitable basis for optimized querying [7].

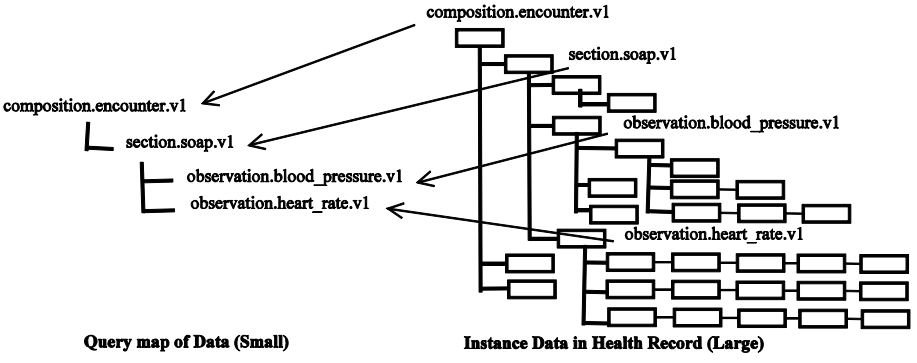

Fig. 5. Data analysis of an EHR transaction [based on 7]

4 Dynamic Generation of User-Interfaces

Recent research emphasizes that good graphical user interfaces that support customization and data validation play a decisive role for user acceptance and data quality [31]. It shows the feasibility of generation of user interfaces from openEHR archetypes. However the use of Mozilla XML User Interface Language (XUL) revealed some problems (e.g., implementation bugs, inconsistencies, lack of modern development environment). Thurston [35] mentions that the presentation of EHR data must be flexible to support a variety of access needs and should be extensible to support the presentation/viewing needs of various clinical and administrative institutions. The opereffa project is the real model of practical EHR use (which is based on openEHR standard) [26]. It makes use of archetype-based user interfaces for the purpose of data entry and validation. Our proposed study implements the tool for generating user interfaces. It shows its usage for the purpose of intelligent querying, and for improving information quality aspects. This is the first initiative taken in this regard.

4.1 User-Interfaces for EHRs

The generated user interfaces records the maximal data about the physiological measurements (Figure 6). For example, blood pressure is higher if a person is lying down than if they are sitting or standing. The reading itself has little meaning without detailed information about the context in which it was taken. This leads to data quality enhancements. Further, the generated UI provides runtime validation of data input - thus improving data entry quality. For example, ADL for archetype ‘heart rate’ (Figure 6 (left) , line 20) defines the constraint that rate should be greater than 0.

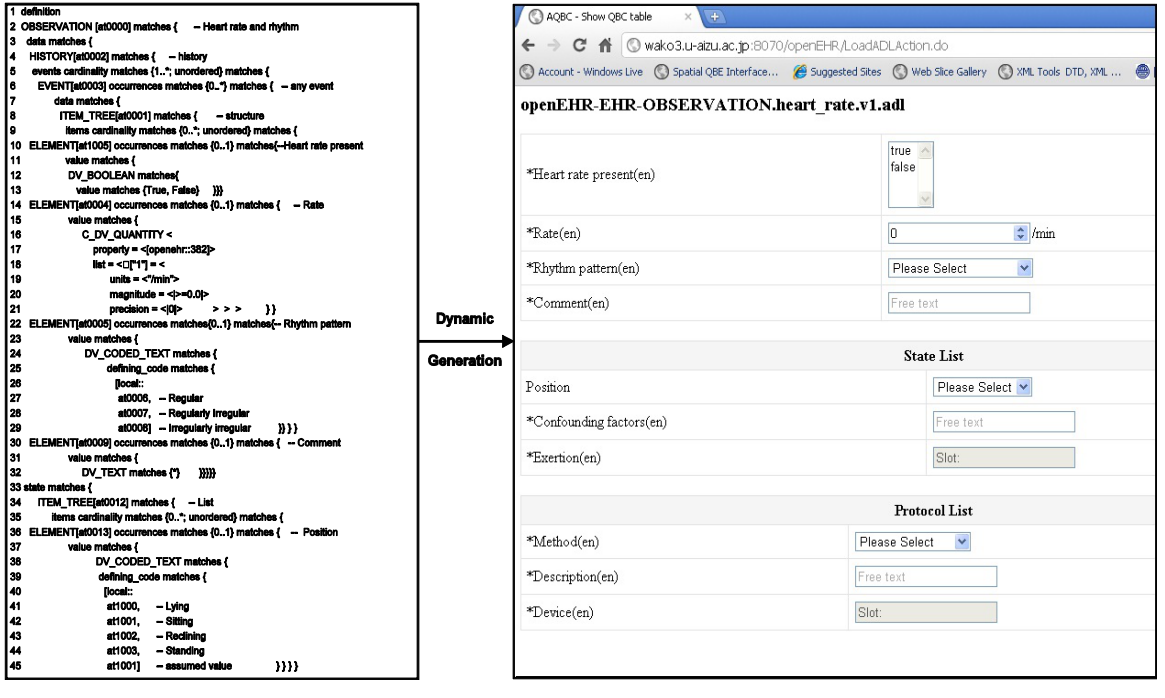

Fig. 6. Dynamic Generation of User Interface for archetype 'heart rate'

This constraint has been represented as shown under 'rate' field in the generated UI (second graphical widget in Figure 6 (right) shows value of '0' and permits the data value only greater than it).

4.1.1 Challenges Encountered

The challenges during automatic generation of user-interfaces are dealing with:

(i) Multitude of representations.
(ii) Different categories of archetypes having different structures.

The prototype deals with the multitude representations. A little difficulty is being faced while dealing with the link type ('reference') object and multimedia ('media-type') object. For example, in case of link type object, the module needs first to search for finding the corresponding object (it may be quantity, text, date), followed by returning the link path. Consider another example, in case of 'media-type' object; it is described by referring to a URL according to openEHR Java API. However, URL is mentioned but is stated through codes. ADL has codes such as '[425, 426, 427,…]', which imply that data should be displayed as [image/cgm, image/gif, image/tiff,…]. The software has been coded with a conversion table (for instance, change of '425' to 'image/cgm').

Archetypes belong to any one among following categories (composition, section, entry and item-structure). There are five sorts of entries (observation, evaluation, action, instruction, administrative) and four types of item-structures - single entities (e.g. weight, height), lists (e.g. blood test results), tables (e.g., visual acuity results) and trees (e.g., biochemistry results) (Figure 4). During implementation, it has been observed that 'action' category completely has another structure. A different presentation model according to each category of archetype has been being built. Thus, all the concepts (279 developed till date) can be represented by using these small number of presentation models.

For current research, the archetypes have been downloaded from openEHR CKM [23]. The corresponding ADLs are stored as an archetype repository. The process of user interface generation involves the following:

(i) 'LoadADL file' module loads the ADL archetype.
(ii) The category of the archetype is being checked.
(iii) The ADL archetype is read using openEHR Java ADL Parser reference implementation which generates Archetype Object Model representation of the archetype.
(iv) These are further pushed to the browser in the form of HTML, using HTML, JavaScript and JavaScript libraries (jQuery and jQuery UI), which turns simple components into more capable components.
(v) User sees a capable, dynamic HTML based UI.

Our approach is dynamic in the sense that whenever clinical knowledge expands, a new archetype is being developed by domain expert without changing the underlying schema. So, we automatically generate a new user interface based on the newly developed archetype. These can be used further for the purpose of intelligent querying, leading to quality enhancements for end-users.

4.2 Querying EHRs

The current research is an effort towards the archetype-based high-level query interface (Figure 3). In contrast to a traditional setting, where users express queries against the database schema, we assert that the semantics of data can often be understood by viewing the data in the context of the user interface (UI) of the software tool used to enter the data. The conceptual model of the user interface of a data-entry application may bear little or no resemblance to the schema of the underlying database. User's view of data in the application is heavily influenced by how the data appears in the user interface. Our goal is to allow domain experts with little technical skill to understand and query data.

We propose that intelligent querying is possible through the use of

(i) Archetype user-interface (Figure 6) and
(ii) Archetype query maps (Figure 5).

When data is committed, its "archetype query map" [7] is computed, and stored separately to aid efficient querying. The map is simply a list of archetypes used to construct the data, keyed by the actual paths in the data where they are used (Figure 5).

The query interface for heart rate is defined by the contents of its archetype. The availability of automatic generation of user-interface on the top of archetypes is effectively a service interface for data. The tool helps in the inspection of an archetype in advance, which can yield a set of path fragments which can be used to query instances which conform to an archetype. The detailed design specifications with sample query examples are a topic for future consideration. The current research presents the implementation issues and details. The study considers QBE (Query-by-Example) [38] as a model which can support many levels of user skills and many types of functional requirements. It can provide an interface that accepts user's intent and communicates well-formed formulas (W.F.Fs) for computations. The proposed approach is implemented on the basic SQL style data operations for queries (relational algebraic operations/set-theoretic operations). The various query examples have been implemented against the functionality, which includes the basic operations - project, select, rename, negation and join (Table 2). It can be further enhanced to include full query language operations. Most of the healthcare worker's needs are met by the single table style queries (i.e., single archetype based). However, the approach is simple for queries involving multiple archetypes, as it is facilitated by the archetype query map. QBE is a relationally complete language [38]. The proposed approach is a relationally complete language as it is built as an extension to QBE. Thus, QBE can be used for archetype-based EHR data. For querying, it provides archetype as a view which provides user-defined subset of a large database. The implementation snapshots for the following query example are shown in Figure 7. In future, the authors propose to test the functionalities of querying through the described approach for semi-skilled users.

Example: Get the heart rate values where the rate is equal to 120 beats per minute and the heart rhythm is regular.

AQL Expression:
SELECT obs/data[at0002]/events[at0003]/data[at0001]/items[at0004]/value[at0017],
FROM EHR [ehr_id/value=$ehrUid]

CONTAINS COMPOSITION [openEHR-EHR-COMPOSITION.encounter.v1]
CONTAINS OBSERVATION obs openEHR-EHR-OBSERVATION.heart_rate.v1]
WHERE
obs/data[at0002]/events[at0003]/data[at0001]/items[at0004]/value[at0017]=120 AND
obs/data[at0002]/events[at0003]/data[at0001]/items[at0005]/value = 'regular'

Query interface:

Step 1. The concept 'heart_rate' is known and selected by the health worker for a specified patient. (Figure 7(a)).

Step 2. The required archetypes connected with 'heart_rate', that is, 'encounter' and 'heart_rate' are prompted to the user in the form of tabular data. (Figure 7(b)).

(archetype description of 'heart_rate' presented by the system)
(archetype description of 'encounter' presented by the system)
[Restrict] and [project] single patient data

The system support facilities for specifying restrict operation (rate=120 AND rhythm= 'regular'). (Figure 7(c)).

5 System Configurations and Architecture

Client-server architecture has been used as shown in Figure 8. The prototype system has been implemented using Java 6 [17], Struts 1.3.10 [33]. A tomcat 6.0 [36] server has been used. The libraries from openEHR Java API 1.0.1 [25] have been incorporated. The other components used are JavaScript (JS) libraries such as, jQuery 1.6.2 [19] and jQuery UI 1.8.16 [20]. jQuery UI provides abstractions for low-level interaction and animation, advanced effects and high-level, themeable widgets, built on top of the jQuery JavaScript Library, that you can use to build highly interactive web applications.

The current prototype has been implemented based on standardized openEHR specifications. To standardize the representation of an international electronic health record, the abstract specifications are defined using the UML notation and formal textual class specifications [5]. The prototype makes use of the archetypes downloaded from a public repository named openEHR CKM [23]. It has 279 numbers of archetypes developed through domain-knowledge governance till date. The prototype has been tested by generation of user-interfaces for a selected list of archetypes. A sample is presented in Figure 9. It consists of examples taken from each category of the archetypes.

Requirement of specific user interface elements and screen design for medical applications have been studied in a wider extend recently. There are many research initiatives by Microsoft such as, Microsoft Health Common User Interface (MS CUI) [22] trying to produce UI elements which would serve better than the usual text box, combo box and other well known UI elements. However, investing into strong UI technologies does not solve the problem, because standardized EHR implementations (openEHR) heavily rely on models, in other words composition of RM classes described by ADL [34]. Joining the UI to these models is an architectural challenge.

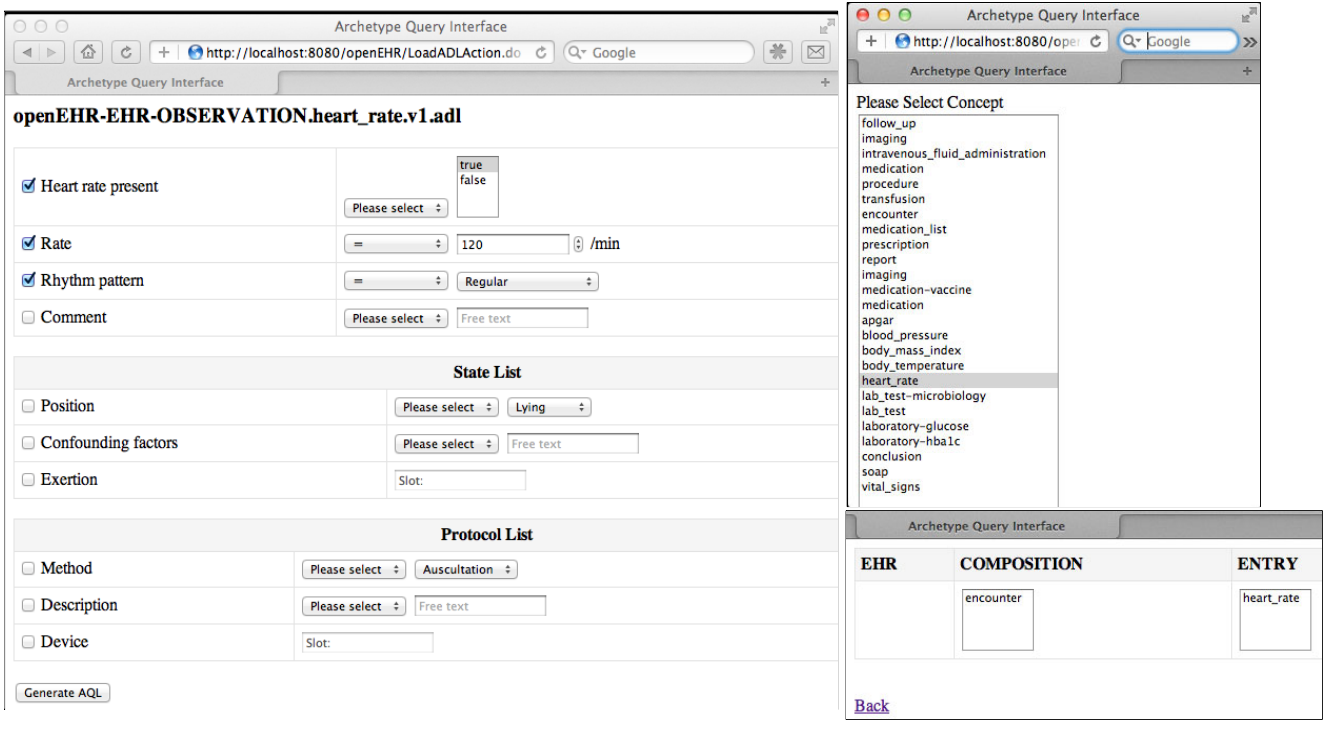

Fig. 7. Snapshots for query example. (a) Snapshot for selecting clinical concept (top-right), (b) Based on chosen concept by user in (a) the related concepts for querying presented by the system (below-right), and (c) Interface for specifying query conditions.

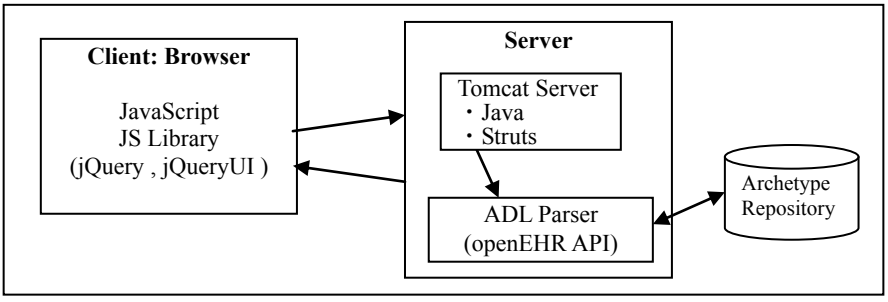

Fig. 8. System Configurations for Generation of User-Interfaces

openEHR-EHR-COMPOSITION.report.v1
openEHR-EHR-COMPOSITION.prescription.v1.adl
openEHR-EHR-EVALUATION.problemdiagnosis.v1
openEHR-EHR-OBSERVATION.laboratory-hba1c.v1
openEHR-EHR-OBSERVATION.blood_pressure.v1
openEHR-EHR-OBSERVATION.body_mass_index.v1
openEHR-EHR-OBSERVATION.heart_rate.v1
openEHR-EHR-OBSERVATION.lab_test-microbiology.v1.adl
openEHR-EHR-ITEM_TREE.medication-vaccine.v1
openEHR-EHR-ITEM_TREE.imaging.v1
openEHR-EHR-ACTION.procedure.v1.adl

Fig. 9. Sample List of Archetypes

Table 2. Relational operations for querying

Query Languages→ QL Functionality↓	**SQL**	**AQL**	**Proposed approach**
Project	√	√	√
Restrict	√	√	√
Rename	√	√	√
Negation	√	√	√
Join	√	√	√
Relational operators/ Boolean Operators	√	√	√
Rename	√	√	√
Disjunction	√	√	√
Union	√	√	√
Difference	√	√	√

6 Related Work and Discussions

Archetype Query Language (AQL) [3] has been developed by openEHR for querying archetype-based EHRs. AQL is used by developer level users (skilled users). Its syntax is complex. It requires more skills than SQL and XQuery, which are at application level. It also requires the knowledge of ADL path. Ocean informatics provides query builder tool for building AQL queries [28]. It often limits the query expressiveness. A common challenge of many query building tools is a case of a complex query. The limitation may be because the chosen graphical metaphor or the tool's native modeling paradigm cannot support all necessary query criteria; or there is limited capability to combine interim solution layers within the tool (e.g., output of one query criterion is input for another criterion). In comparison, the EHR system must have an appealing and responsive high-level query interface that provides a rich overview of data and an effective query mechanism for patient data. It should support semi-skilled users at clinics or hospitals.

All of the form-based query interfaces share two characteristics in common, besides their form-based nature. First, they are all intended to hide some complex query operations from the user. In most cases, the complex query operation in question is the join operator. Second, these techniques typically expose only a subset of the data available in a database. For instance, an application may have a search form for building complex queries to find medical providers in a database, but that form was custom-built by a developer anticipating a certain class of query; one cannot then use the same query form to search for patients. In recent research, one of the metrics that they use to evaluate their generated search forms is coverage of schema elements (tables and foreign keys) because they generate forms only for the most frequently accessed tables and the most frequently issued queries (to keep the number of generated forms low) [18].

Jayapandian and Jagadish [18] proposed tools that generate forms-based query interfaces based on the schema of the database (and the profiles of executed queries). The query interface generation in the current study is through the EHR schema, presented in section 3.1. It uses the bottom-up approach for the query interface generation. In our proposal, the user's view is incorporated despite the use of actual database schema tables, which is more user-friendly. The current research treats the user interface itself as a view schema and constructs the query interface. It does not require users to specify complex operators such as joins. The query interface exposes all the data that is available in the user interface of the application in a similar way, as the QBE approach. It generates a query interface from the conceptual model. The interface poses limitation in the kinds of queries that it can express relative to SQL.

FoxQ [1] and EquiX [8] help end users rather than form developers to build forms to query a database. These approaches provide graphical view of the database schema and allow users to specify predicates on schema attributes to build queries incrementally. We provide a tabular view of archetype map (archetypes involved in querying) and allow users to specify predicates on archetype attributes to build queries incrementally.

Earlier, the use of XQBE (XQuery-by-Example) interface directly over the XML description of archetype has been examined [30]. The XQBE interface requires some knowledge of tree or graphs on the part of users, whereas QBE interface is quite

simple and intuitive to use. In the current approach, the archetype query map in combination with archetype user interfaces provide the information of what items will be available in any actual data, enabling complex queries to be served easily.

6.1 Information Quality Requirements

The keyword and form-based interfaces have limitations for EHR domain, where obtaining accurate and qualitative data is very important for medical users. Suitable query language capable interfaces enhance information quality. Many previous studies have identified the need of high-level query language interface for improving information quality gains in EHR domain [29] [32].

Various information and data quality issues have been reported in data integration systems [4]:

(i) Technological heterogeneity
(ii) Schema heterogeneity due to
- Different data model and
- Different data representations, and

(iii) Instance-level heterogeneity (caused by different conflicting data values provided by distinct sources for the same objects). These are caused by quality errors such as accuracy, consistency, completeness and currency errors.

The recent research studies highlight the handling of these issues through the use of abstract specifications (defined using UML), adherence to a single Reference Model and adoption of international set of archetypes developed through domain knowledge governance. Thus, this paper contributes in following ways. First, it demonstrates that information and data quality can be included as an integral part during the user-interface generation. Secondly, it offers a perspective for the migration from today's focus on the EHR domain towards a broader concern for enhancing information quality. The research highlights the need for Quality of the content being accessed from EHR repositories. It is related to topics like "structuring" the data, information sources, modeling and encoding, and consolidating redundant data. It focuses on how the method of data capture (through archetypes) plays an important role in shaping what is possible to measure in an electronic environment.

The generated user-interfaces specify the design of the clinical data that a health care professional needs to store in a computer system. Thus, we show how the generation of UI can be used to capture context knowledge and improve information access and its quality by automatically reconciling semantic differences between the sources and the receivers. The proposed interface meets a key requirement for semantic interoperable EHR systems which is the key to information and data quality in healthcare domain.

7 Summary and Conclusions

Keeping in mind, the ultimate goal of data engineering, which is to put high quality data in the hands of users, this study has explored the generation of user interfaces

from archetypes (which are powerful representations of clinical knowledge). The current research treats the user interface itself as a view schema and constructs the query interface against it that may be more user-friendly. The efforts are undergoing for implementation to further test the approach with sample set of queries for all database operations with medical users. The graphical interface would complement efforts by the openEHR foundation, Microsoft, ISO 13606 and CEN/TC251 (European committee for standardization), which are working on semantic interoperable EHRs, making use of archetype-based EHRs. Thus, querying in an environment, where all sources and all receivers of data always have the same meanings, will reduce the problem of semantic heterogeneity.

In contrast to the old message paradigm, the archetype paradigm is more flexible and easily scalable because it provides a means to handle knowledge evolution. This technology avoids reimplementation of systems, migrating databases and allows the creation of future-proof systems. The study explores the aspects of archetypes considering the DQ attributes of wholeness, discrete, specialization, simple, generic and re-usable.

References

1. Abraham, R.: FoXQ – XQuery by Forms. In: Proceedings 2003 IEEE Symposium on Human Centric Computing Languages and Environments (2003)
2. Agency for Healthcare Research and Quality (AHRQ). Research Finds Low Electronic Health Record Adoption Rates for Physician Groups. Press release (September 14, 2005)
3. Archetype Query Language Description, http://www.openehr.org/wiki/display/spec/Archetype+Query+Language+Description
4. Batini, C., Scannapieco, M.: Data Quality: Concepts, Methodologies, and Techniques. Springer, Berlin (2006)
5. Beale, T., Heard, S.: openEHR Architecture: Architecture Overview in The openEHR release 1.0.2. In: Beale, T., Heard, S. (eds.) openEHR Foundation (2008)
6. Beale, T., Heard, S.: The openEHR Archetype Model-Archetype Definition Language ADL 1.4. openEHR release 1.0.2 (December 12, 2008)
7. Beale, T.: Archetypes Constraint-based Domain Models for Future proof Information Systems. The OpenEHR Foundation (August 21, 2001)
8. Cohen, S., Kanza, Y., Kogan, Y., Sagiv, Y., Nutt, W., Serebrenik, A.: EquiX—A search and query language for XML. Journal of the American Society for Information Science and Technology 53, 454–466 (2002)
9. Conde, A.M.: Towards best practice in the Archetype Development Process. Master thesis, Trinity College Dublin, Center for Health Informatics, Department of Computer Science (September 2010)
10. European committee for Standardization, Technical committee on Health informatics, Standard for EHR communication, http://www.cen.eu
11. EUROREC. European Record Institute (2010), http://www.eurorec.org/
12. Garde, S., Hovenga, E.J.S., Gränz, J., Foozonkhah, S., Heard, S.: Managing Archetypes for Sustainable and Semantically Interoperable Electronic Health Records. Electronic Journal of Health Informatics (2007)
13. HL7 (Health Level 7), http://www.hl7.org (retrieved November 10, 2010)
14. HL7 CDA (Clinical Document Architecture), Release 2, http://www.hl7.org/v3ballot/html/infrastructure/cda/cda.htm

15. ISO 13606-1, Health informatics - Electronic health record communication - Part 1: Reference Model (2008)
16. ISO 13606-2, Health informatics - Electronic health record communication - Part 2: Archetype interchange specification (2008)
17. Java 6, http://www.oracle.com/technetwork/java/javasebusiness/downloads/java-archive-downloads-javafx-419431.html#jdk-6u23-javafx-1.3.1-oth-JPR
18. Jayapandian, M., Jagadish, H.V.: Automated Creation of a Forms-based Database Query Interface. In: VLDB 2008, Auckland, New Zealand, August 25-28, pp. 695–709 (2008)
19. jQuery 1.6.2, http://jquery.com/
20. jQueryUI 1.8.16, http://jqueryui.com/
21. Madnick, S., Zhu, H.: Improving Data Quality through Effective Use of Data Semantics. Journal of Data & Knowledge Engineering - Special issue: WIDM 2004 59(2) (November 2006)
22. Microsoft Health Common User Interface (MS CUI), http://www.mscui.net/
23. openEHR Clinical Knowledge Manager version 1.0.5, http://www.openehr.org/knowledge
24. openEHR Foundation, http://www.openehr.org
25. openEHR Java API, http://www.openehr.org/wiki/display/projects/Java+Project+Download
26. Opereffa Project, http://opereffa.chime.ucl.ac.uk/introduction.jsf
27. Orfanidis, L., Bamidis, P.D., Eaglestone, B.: Data Quality Issues in Electronic Health Records: An Adaptation Framework for the Greek Health System. Health Informatics Journal 10(1), 23 (2004)
28. Query builder, Ocean Informatics, http://www.oceaninformatics.com/Solutions/ocean-products/Clinical-Modelling/Ocean-Query-Builder.html (accessed April 4, 2011)
29. Sachdeva, S., Bhalla, S.: Electronic Health Record- A Framework for Standardization and Semantic Interoperability, Technical Report, University of Aizu, Technical Report 2010-001 (November 29, 2010)
30. Sachdeva, S., Bhalla, S.: Implementing High-Level Query Language Interfaces for Archetype-Based Electronic Health Records Database. In: International Conference on Management of Data (COMAD), pp. 235–238 (December 2009)
31. Schuler, T., Garde, S., Heard, S., Beale, T.: Towards automatic generation of GUIs from archetypes. Studies in Health Technology and Informatics 124, 221–226 (2006)
32. SNOMED (Systematized Nomenclature of Medicine) Clinical Terms, http://www.snomed.org/documents/snomed_overview.pdf
33. Struts 1.3.10, http://struts.apache.org/download.cgi
34. Technology and architecture challenges in UI implementation, http://www.openehr.org/wiki/display/projects/Technology+and+architecture+challenges+in+UI+implementation
35. Thurston, L.M.: Flexible and Extensible Display of Archetyped Data: The openEHR Presentation Challenge. In: Proceedings of HIC 2006 and HINZ 2006, Brunswick East, Vic.: Health Informatics Society of Australia, pp. 28–36 (2006)
36. Tomcat 6.0, http://tomcat.apache.org/download-60.cgi
37. Wang, R.Y., Zaid, M., Lee, Y.W.: Book on “Data Quality”. Kluwer Academic publishers (2001)
38. Zloof, M.M.: Query-By-Example. In: AFIPS 1975: Proceedings of the National Computer Conference and Exposition, May 19-22 (1975)