Technical Report 2010-001

Electronic Health Record - Standardization and Semantic Interoperability

Shelly Sachdeva, Subhash Bhalla

November 29, 2010

Graduate School of Computer Science and Engineering
The University of Aizu
Tsuruga, Ikki-machi, Aizu-Wakamatsu City
Fukushima 965-8580, Japan

Technical Report 2010-001

Title:
Electronic Health Record - Standardization and Semantic Interoperability

Authors:

Shelly Sachdeva, Subhash Bhalla

Key Words and Phrases:
Electronic Health Records, Data Quality in Healthcare, Archetype-based EHR, Quality-based EHR, Semantic Interoperability, Standardization in EHR, openEHR.

Abstract:

Different clinics and hospitals have their own information systems to maintain patient data. This hinders the exchange of data among systems (and organizations). There is a need to provide standards for data exchange. In digitized form the individual patient's medical record can be stored, retrieved and shared over a network through enhancement in information technology. Thus, Electronic Health Records (EHRs) should be standardized, incorporating semantic interoperability. A subsequent step requires that healthcare professionals and patients get involved in using the EHRs, with the help of technological developments. This study aims to provide different approaches in understanding some current and challenging concepts in health informatics. Successful handling of these challenges will lead to improved quality in healthcare by reducing medical errors, decreasing costs, and enhancing patient care. The study is focused on the following goals:

1. Understanding the role of EHRs,
2. Understanding the need for Standardization to improve quality,
3. Establishing Interoperability in maintaining EHRs,
4. Examining a framework for Standardization and Interoperability--- The openEHR Architecture,
5. Identifying the role of Archetypes for Knowledge Based Systems, and
6. Understanding the difficulties in querying EHR Data.

Report Date:	Written Language:
11/29/2010	English

Any Other Identifying Information of this Report:
Submitted to ACM Journal on Data and Information Quality, 9 Jan. 2010 (JDIQ-2010-01-0012R1) and revised on 4 September 2010; also intermediate version submitted on 12 February 2010 (JDIQ-2010-02-0022).

Distribution Statement:
First Issue: 5 copies

Supplementary Notes:

Graduate School of Computer Science and Engineering
The University of Aizu
Aizu-Wakamatsu
Fukushima 965-8580
Japan

Electronic Health Record - Standardization and Semantic Interoperability

Shelly Sachdeva, Subhash Bhalla
Graduate School of Computer Science and Engineering
The University of Aizu
Aizu-Wakamatsu, Japan

ABSTRACT

Different clinics and hospitals have their own information systems to maintain patient data. This hinders the exchange of data among systems (and organizations). There is a need to provide standards for data exchange. In digitized form the individual patient's medical record can be stored, retrieved and shared over a network through enhancement in information technology. Thus, Electronic Health Records (EHRs) should be standardized, incorporating semantic interoperability. A subsequent step requires that healthcare professionals and patients get involved in using the EHRs, with the help of technological developments. This study aims to provide different approaches in understanding some current and challenging concepts in health informatics. Successful handling of these challenges will lead to improved quality in healthcare by reducing medical errors, decreasing costs, and enhancing patient care. The study is focused on the following goals:

1. Understanding the role of EHRs,
2. Understanding the need for Standardization to improve quality,
3. Establishing Interoperability in maintaining EHRs,
4. Examining a framework for Standardization and Interoperability--- The openEHR Architecture,
5. Identifying the role of Archetypes for Knowledge Based Systems, and
6. Understanding the difficulties in querying EHR Data.

Categories and Subject Descriptors: A.1 **[Introductory and Survey**]; H.2 **[Database Management**] - Logical Design; J.3 **[Life and Medical Sciences**] - Medical Information Systems.

General Terms: Design, Languages, Standardization

Additional Key Words and Phrases: Electronic Health Records, Data Quality in Healthcare, Archetype-based EHR, Quality-based EHR, Semantic Interoperability, Standardization in EHR, openEHR.

1. INTRODUCTION

Healthcare is an information-intensive activity producing large quantities of data from laboratories, wards, operating theatres, primary care organizations, and from wearable and wireless devices [Simonov et al. 2005]. Thus, the management of information across systems and organizations requires collaboration, portability and data integration. In addition, both patient safety and healthcare cost influence the quality of healthcare. To obtain these, efficient and accurate data capture is needed. In view of these contingencies, EHRs are becoming a method by which physicians are able to electronically capture high-quality data at a fast speed and at low cost.

1.1 Role of Electronic Health Records

An Integrated Care EHR [ISO/TC215 2003] is defined as: "a repository of information regarding the health of a subject of care in computer processable form, stored and transmitted securely, and accessible by multiple authorized users. It has a commonly agreed logical information model which is independent of EHR systems. Its primary

purpose is the support of continuing, efficient and quality integrated healthcare and it contains information which is retrospective, concurrent and prospective". EHRs are made easily accessible through the World Wide Web (WWW). Consequently, this data can be used by clinicians, hospitals, patients, healthcare organizations and decision makers, for a variety of purposes.

A record of the longitudinal health history of each patient is required to improve quality of care. Some of the main concerns in maintaining EHRs are privacy, security, standardization and interoperability. EHRs will play an important role in telemedicine, emergency situations, home-care, epidemiological situations and in creating an e-health environment. The new environment will help to prevent medication errors, reduce duplication, and save time. It will facilitate better coordination of long-term patients' data. Thus, quality of care for patients will improve by the use of standardized records [Øvretveit et al. 2007]. EHRs must be designed to capture relevant clinical data using standardized data definitions and standardized quality measures. These will help in improving preventive care and in increasing physician efficiency [Poissant et al. 2005].

Several organizations are working to create EHR standards, such as openEHR Foundation [openEHR 2009], Consolidated Health Informatics Inititiative (CHI) [EHR Standards 2009], Certification Commission for Healthcare Information Technology (CCHIT) [EHR Standards 2009], Healthcare Information and Management Systems Society (HIMSS) [EHR Standards 2009], International Organisation for Standardisation (ISO), American National Standards Institute (ANSI) [EHR Standards 2009], Canada Health Infoway [EHR Standards 2009], and European Committee for standardization (CEN's TC 251) [CEN/TC251]. CEN/TC251 is a regional Standards Development Organization. The standards created by these organizations are formalized, controlled and documented [Lewis et al. 2008]. Standards provide a definitional basis for interoperability [Atalag et al. 2010]. The single-vendor, closed-data paradigm of commercial development is broken by the open source software development sector promoting shared, universal standards.

Among these, the openEHR Foundation has proposed openEHR standards. These support version-controlled health records. Version control enables all past states of a health record to be investigated in the event of clinical or medico-legal investigations. The openEHR stores the most frequently used information in a separate record for the purpose of fast lookup and querying. In this report, we study the openEHR proposals and use the term 'openEHR' to mean the foundation or the standard, depending on the context. The openEHR Foundation was established by University College London and Ocean Informatics [Ocean Informatics 2009]. It is an international foundation working towards semantic interoperability of EHR and improvement of healthcare. Recently, Microsoft has also adopted the openEHR's approach for EHRs [Microsoft 2009]. Other organizations are also developing standards, such as HL7 version 3 and CEN13606, with similar goals [Eichelberg 2005].

1.2 Data Quality in Electronic Health Records

Data quality (DQ) concerns the correctness, timeliness, accuracy, and completeness that make data appropriate for use [Gendron and D'Onofrio 2001]. EHR data quality is often considered only within the narrow scope of data verification and validation. Data quality should also concern the equally critical aspect of assuring that EHR data are appropriate for use [Orfanidis et al. 2004]. DQ builds trust. The various data issues can be incompleteness (missing information), inconsistency (information mismatch between various sources or within the same EHR data sources) and inaccuracy (non-specific, non-standards-based, inexact, incorrect, or imprecise information). Such inaccuracies in the attribute values of patient records make it difficult to find specific patient records [Mikkelsen and Aasly 2005].

The Relation in Figure 1 describes patients, with P_Name, Doctor, First Encounter Date, Revisits and Last Encounter Date. In Figure 1, the cells with data quality problems are shaded. At first, only the cell corresponding to the p_name of patient 3 seems to be affected by a data quality problem. In fact, there is a misspelling in the p_name, where Smih stands for Smith, thus causing an accuracy problem. Nevertheless, another accuracy problem is related to the exchange of the doctor between patient 1 and 2; Ambica is actually the doctor of patient 2 and Yoku the doctor of patient 1. Other data quality problems are a missing value for the doctor of patient 4, causing a completeness problem, and a 0 value for the number of Revisits of patient 4, causing a currency problem because a revisit of the patient has actually been proposed. Finally, there are two consistency problems: first, for patient 1, the value of Last Encounter Date cannot be earlier than First Encounter Date; second, for patient 4 the value of Last Encounter Date cannot be different from null, because the value of Revisits is 0.

Id	P_Name	Doctor	First Encounter Date	Revisits	Last Encounter Date
1	Jerry	Ambica	20 May, 2010	3	20 Jan, 2010
2	Julia	Yoku	12 Feb, 2010	0	NULL
3	Smih	Indira	06 Apr, 2010	0	NULL
4	Sabrina	null	22 May, 2010	0	20 July, 2010

Fig. 1. Relation for 'Patient' with Data Quality Problems

EMRs have been weakened by bad record design and shallow research on user interfaces to fill in and extract data, leading to incomplete and incorrect data records. In contrast, the EHR record design discussed in the paper is based on clinical investigator recording process. Also, the user interfaces are built on the top of qualitatively designed archetypes, thus helping to obtain correct and complete records.

1.3 Data Quality Problems/ Issues

The various data quality problems and issues are given below.

1. There can be problems in the input of data.
2. There can be various types of errors such as missing values, syntax violation, domain violation (overloaded attribute, misspelling error, ambiguous value), incorrect value, violation of business rule, uniqueness violation, existence of synonyms, violation of functional dependency, inconsistent duplicate tuples, referential integrity violation, incorrect reference, heterogeneity of syntaxes, heterogeneity of measure units, heterogeneity of representation and existence of homonyms.
3. Default assumptions are not appropriate for the entry of clinical data.
4. Coded data always imply a simplification of reality. When people are forced to work from a predetermined list of codes, they may not find correct code and so they may select a code that seems to be closest to, but is not truly representing, the real situation or observation. Additionally, coding problems increase when trying to integrate databases. Creating a common coding system from different sets of codes is complicated because it is probable that different processes for coding and different definitions were used [Hristidis 2009].
5. The differences in protocol of data collection have an important impact on data interpretation.

The remaining part of this paper is organized as follows. Section 2 emphasizes the role of standardization of EHR for improvement of quality. Section 3 describes semantic interoperability with emphasis on a dual-model approach. In Section 4, the standardized openEHR architecture is discussed. Section 5 details enhancement of quality by use of archetypes. It also explains archetype description language, a language rich enough to

capture and model entities within the medical care domain. In Section 6, the real challenge of achieving quality in healthcare systems is addressed. Section 7 explains querying of the EHR data, describing various research challenges in querying and presenting a brief description of archetype query language. Section 8 presents discussions of the challenges in design and implementation. Section 9 presents high-level query language interfaces. Section 10 describes the data quality considerations. Finally, the summary and conclusions for the research study are included in Section 11.

2. STANDARDIZATION OF EHR FOR IMPROVING QUALITY

Data collected in various systems can have quality faults [Miettinen and Korhonen 2008]. It can, for instance, be non-coherent or include contradictory information. The desired data may be completely missing. For example, the unit for temperature may not be entered definitively as degree Celcius or Fahrenheit, or the blood pressure may be entered outside the permissible range. There is a need for a communication format and protocol for the purpose of standardization, since a patient's health information is shared in a multi-disciplinary (shared care) environment. Thus, the development and adoption of national and international standards for EHR interoperability is essential. In the current environment, it is necessary to support interoperability between software from different vendors. Standardization will enhance the quality of EHR systems [Miettinen and Korhonen 2008; Maldonadoa et al. 2007], and in this regard, many research studies discuss different approaches to improve quality [Øvretveit 2003].

It is useful to consider the Internet as an analog in development of standardization. As a result of many years of research, the Internet is based on standards such as TCP/IP, SMTP, UTF-8, XML and HTML. In most cases, a user is able to use any browser on any platform to navigate the World Wide Web. Similarly, there is a need for standardization of EHRs. Healthcare professionals must be able to achieve timely and consistent access to EHRs. Thus, standards are the key for successful implementation of any EHR system.

Currently, there is no single universally-accepted clinical data model that will be adhered to by all [Blobel and Pharow 2008]. Yet, various components within the clinical practice, including terminology systems and EHR systems, need to be in harmony. Figure 2 illustrates why standardization is needed in inter-organization transfer of data. The meaning of information must be preserved across various applications, systems and enterprises. However, the major problem is the huge amount of different (proprietary or standardized) interfaces which are in use [Bott 2004]. For example,

i) Message or interface standards - Health Level 7(HL7), Electronic Data Interchange For Administration, Commerce and Transport (EDIFACT), and Digital Imaging and Communications in Medicine (DICOM);
ii) Content-oriented standards - Logical Observation Identifiers Names and Codes (LOINC), The International Statistical Classification of Diseases and Related Health Problems 10th Revision (ICD-10), International Classification of Procedures in Medicine (ICPM); or
iii) Hybrid standards - CEN 13606 and openEHR.

A recent study provides comparison of the available EHR standards [Blobel and Pharow 2008]. Furthermore, there are mappings being developed among standards. For example, ISO 13606-1 is the model for an EHR Extract, designed to enable data to be shared between different EHR systems [ISO 13606-1 2008]. And a mapping algorithm is being developed that allows a bi-directional transform between openEHR and ISO 13606 [Beale 2010].

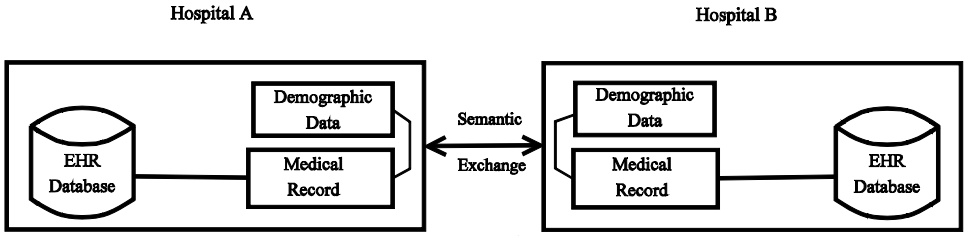


Fig. 2. Standardization of Electronic Health Record

Formally, four layers of standardization have been recognized –content, structure, technological and organizational [Bott 2004]. The content layer and the structure layer are concerned with the standardization of the elements of an EHR.

i) The content layer addresses aspects of coding. It uses terminological systems, such as classifications or controlled vocabularies.

ii) The structure layer focuses on regulations concerning the structure of EHR elements. Its examples include XML-files that are based on standardized DTDs (or XML-Schemas). Several content-oriented aspects, such as a discharge letter, are usually modeled by defining the structure.

iii) The technological layer contains regulations concerning aspects such as software and hardware components, distribution, objects and services, and the Public Key Infrastructure (PKI) for data security.

iv) The organizational layer focuses on changes caused by the usage of an EHR system in an organization. These concern business processes, guidelines, protocols, roles and PKI.

Organizations adopt the standards to achieve interoperability and promote information quality [Lewis et al. 2008]. However, there are problems in reaching agreements on standards. There is a technical problem involving the development of a language sufficiently rich to capture and model the medical domain. Also, there is a human problem involving agreement on what is contained within the domain, and why it is important. This study addresses these problems in Section 5 and Section 6. A brief description of archetype description language and domain knowledge governance is presented there, as a way to overcome these problems.

2.1 Standardized EHRs

In essence, the proposed Electronic Health Records (EHRs) have a complex structure that may include data from about 100-200 parameters, such as temperature, blood-pressure and body mass index. Individual parameters will have their own contents. Each contains an item, such as 'data' (e.g., captured for a blood pressure observation). It offers complete knowledge about a clinical context, (i.e., attributes of data), 'state' (context for interpretation of data), and 'protocol' (information regarding gathering of data), as shown in Figure 3 (depicting completeness).

In order to serve as an information interchange platform, EHRs aim to use archetypes to accommodate various forms of contents [Beale and Heard 2008 a; ISO 13606-1 2008]. The EHR data will have a multitude of representations. The contents may be structured, semi-structured or unstructured, or a mixture of all three. These may 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).

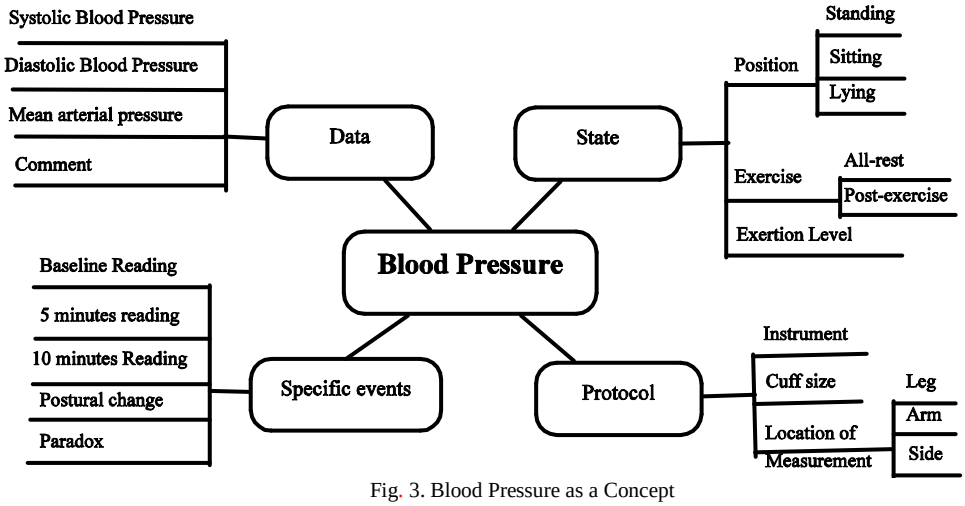


Fig. 3. Blood Pressure as a Concept

3. ESTABLISHING INTEROPERABILITY IN MAINTAINING EHR

Different clinics and hospitals have their own complex information systems to maintain patient data. These may be made up of paper records or electronic medical records (EMRs). EMRs consist of data such as patient demographics, medical history, medicine and allergy lists (including immunization status), laboratory test results, radiology images, billing records and advanced directives. There is redundancy in existing data because of distributed and heterogeneous data resources. This situation hinders the exchange of data among systems and organizations. Professionals cannot use the information provided by other organizations. The same data is sometimes input several times, leading to quality faults.

An additional concern is the complexity of the health domain. It is evolving at a fast rate. Healthcare-related knowledge is becoming broad, deep and rich with time. Thus, there is a need for legacy migration of data. Large scale financial gains in savings can be achieved from interoperability and health information exchange [Walker et al. 2005]. EHRs can be standardized and should incorporate semantic interoperability. National Health Information Network (NHIN) defines semantic interoperability as "the ability to interpret, and, therefore, to make effective use of the information so exchanged" [NHIN 2005]. Similarly, IEEE Standard 1073 defines semantic interoperability as: Shared Data Types, Shared terminologies and Shared codings [Kennelly 1998].

Consequently, standardized terminology is a critical requirement for healthcare applications to ensure accurate diagnosis and treatment. It has led to developing standards, such as Systematized Nomenclature of Medicine - Clinical Terms (SNOMED-CT) [SNOMED 2009]. Shared codings refer to establishing standard encodings to be shared among systems. Such codings refer not only to encoding software functions, but also to encoding medical diagnoses and procedures for claims processing purposes, research, and statistics gathering[1]. Shared data types refer to the types of data exchanged by systems. Interoperability requires that systems share data types on many different levels, including messaging formats (e.g., XML, ASCII), and programming languages (e.g., integer, string). Shared terminologies refer to establishing a common vocabulary for the interchange of information. Semantic interoperability may also require support of ontological mappings

[1] (For example, the International Statistical Classification of Diseases and Related Health Problems - 10th Revision (ICD10) and Current Procedural Terminology. This is a systematic coding system for reporting medical services and procedures performed by physicians).

at the conceptual level (Figure 3 and Section 5).

Most common difficulties in exchange of data arise due to the heterogeneous nature and fragmentation of healthcare organizations. Some of the challenges for achieving interoperability are standardization and reliability [Alexandrou and Mentzas 2009]. Thus, there is a need for precisely defined medical information items to make sure that Semantic Interoperability is assured. The following sections present the model (Section 3) and specifications (Section 4) for achieving interoperability.

3.1 Dual Model Approach

Traditionally, three methodologies have been used for building systems such as EHR systems [Patrick et al. 2006]:

- unstructured approach,
- "BIG" model approach, and
- the generic approach.

The unstructured approach to EHR is simply a warehouse filled with unstructured text. This is problematic, as the system cannot be queried readily, nor reported for management purposes. The "BIG" model approach has a separate table for each clinical concept leading to excessively large schemas. While this proves effective in querying, it leads to errors as few people can completely understand the entire model. Furthermore, the model is brittle over time as it does not suit the volatile, rapidly-developing medical domain. The classic (former) approaches require the details of clinical knowledge to be simultaneously coded into the software. Thus, with the expansion of clinical knowledge the software became unsuitable and outdated. However, the generic model is designed to allow a wide variety of data to be accommodated in a general purpose set of data structures. The model is small (to understand conceptually). It suffers from query difficulties. The stored data is similar to that of the unstructured process.

In order to overcome the problem of lower Data Quality (DQ) from generic modeling, a constraint mechanism must be introduced (for each parameter) to ensure that the stored information is valid in terms of the clinical domain (Figure 4). The mechanism is referred to as the "Archetype Model". This model expresses the character of clinical data attributes and stores this information as data in the database rather than in the database schema. There is a quality improvement as each time there is change in the domain knowledge, the software need not be changed. It was initially developed by the Good Electronic Health Record project [GEHR], and later adopted by the openEHR Foundation. It is well aligned, and makes advances on the CEN 13606 standard [CEN/TC251].

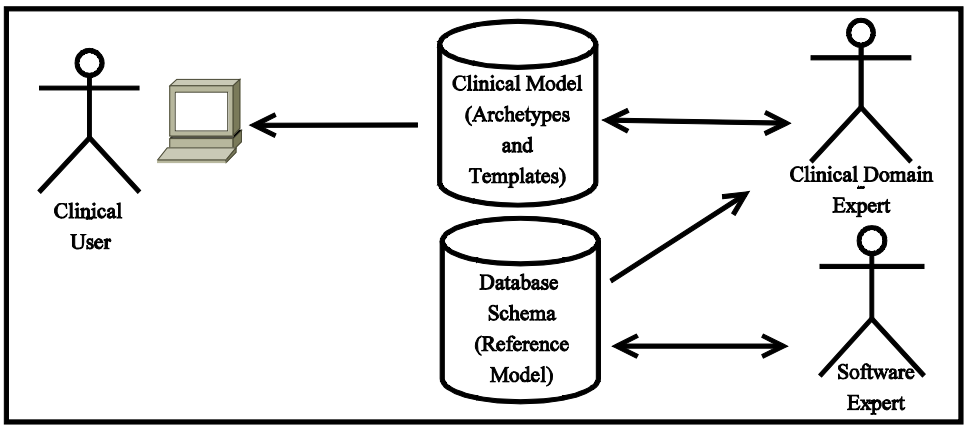


Fig. 4. Two-Level Modeling Approach

Further, in order to achieve interoperability, a two-level modeling approach for

separation of information and knowledge has been used. It is specified in open Electronic Health Record Architecture (EHRA) [Beale and Heard 2008 a; ISO 13606-1 2008]. Examples of Dual Model EHR architecture are CEN/TC251 EN13606 [CEN/TC251] (developed by the European Committee for Standardization) and openEHR. The modeling helps quality improvement by sharing of archetypes via a repository with versioning, by assigning of unique archetype identifiers, and by a widely-accepted underlying reference model.

The two-level approach consists of a Reference Model (RM) and the domain level definitions in the form of archetypes and templates (Figure 3). The concept behind it is the introduction of a level of abstraction between the program logic and the database schema [Beale and Heard 2008 a]. This mechanism provides data independence, similar to the case of conventional Database Management Systems (DBMSs) [Silberschatz et al. 2005]. EHR systems based on this approach have the capability of incorporating new standardized data elements in a timely manner. A domain expert designs archetypes, and the user creates the information item which is mapped to an archetype (Figure 5) [Beale and Heard 2008 a]. The dual model EHRA specifications have already been adopted by Microsoft [Microsoft 2009].

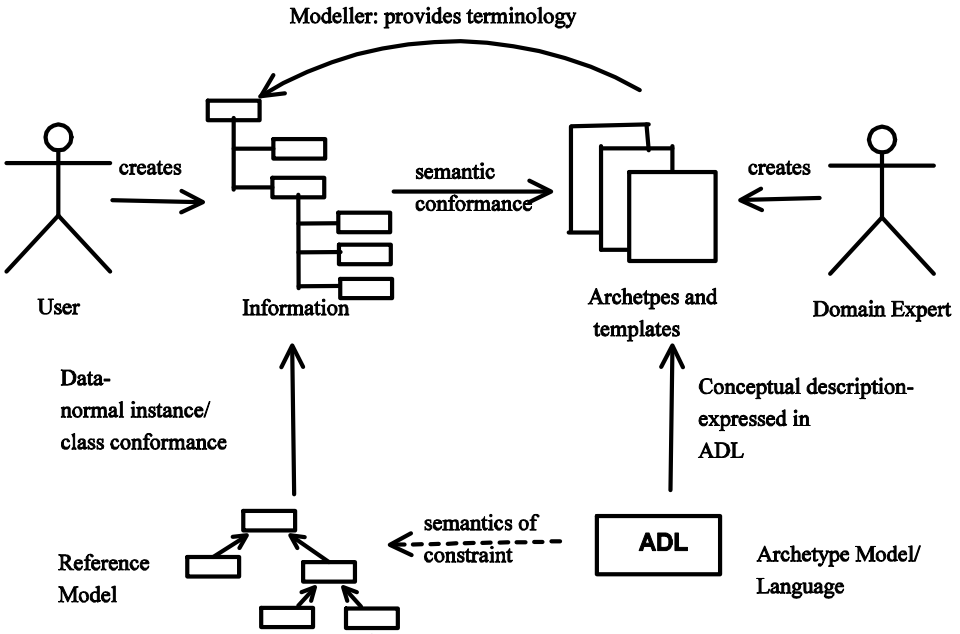


Fig. 5. Expansion of Healthcare Knowledge – Archetype Meta-architecture

3.2 Reference Model and Conceptual Model

At the lower level, Reference Model (RM) is an object-oriented model. It contains basic entities for representing any entry in an EHR. The software and data can be built from RM. Concepts in openEHR RM are invariant. These comprise a small set of classes that define the generic building blocks to construct EHRs. At the upper level, semantic interoperability is achieved by a precise definition of information items called "archetypes". The EHR is based on such archetypes [Beale and Heard 2005]. These are exchanged by the sending and receiving systems. These archetypes are formal definitions of different clinical concepts in the form of structured and constrained combinations of the entities of a RM.

A conceptual definition of data as archetypes can be developed in terms of constraints on structure, types, values, and behaviors of RM classes. These can be created for each

concept in the domain for which the user may have a need. For example, a generic class "PARTY" can represent different domain concepts such as patient, doctor or nurse (Figure 6). Thus, the archetype model (conceptual model) level focuses on individual, self contained, clinical attributes that are independent meta descriptions of clinical information such as 'blood pressure' (Figure 3), 'mode of delivery' and 'birth weight'. This model expresses the character of these clinical data attributes. It stores this information as data in the database rather than in the database schema. Additional software is needed to manage this 'meta' data. A component named as the 'modeller' supports data capture with reference to the archetype model (Figure 5).

Standardization can be achieved in this manner. Whenever there is a change in the clinical knowledge (or requirements), the software need not be changed. The archetypes need to be modified (or added) in conformity with RM. This leads to enhancement in terms of data quality and Information Quality (IQ). The segregation of information from knowledge is shown by the dual levels and directional arrows in Figure 4 and Figure 5. The terminologies contain facts about the real world. The clinical user can enter and access the information through clinical application. The clinical domain expert can record and maintain the clinical model through the modeller. The modeller is software needed to manage the archetypes. The clinical model addresses aspects of coding the content of EHR-Element using terminological systems like classifications or controlled vocabularies.

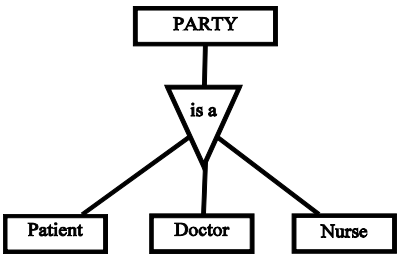


Fig. 6. PARTY – as a Generalization of Concepts

Thus, archetypes have the feature to separate the internal model data from formal terminologies. The internal data is assigned local names which can later be bound or mapped to external terminology codes. This feature eliminates the need to make changes to the model whenever the terminology changes. Matching clinical data to codes in controlled terminologies is the first step towards achieving standardization of data for safe and accurate data interoperability. Archetype Definition Language (ADL) syntax has been proposed by openEHR. It is one possible serialization of an archetype. It is used to describe constraints on data which are instances of the reference model (information model). The Archetype Model structurally expresses the semantics of the ADL (Figure 5). ISO has accepted ADL as a standard language for description of archetypes [ISO 13606-2 2008].

4. STANDARDIZATION AND INTEROPERABILITY (openEHR ARCHITECTURE - IN A DATA QUALITY PERSPECTIVE)

The above dual level system has been developed and implemented by openEHR to improve the data quality. It is a two-level software engineering approach. It separates knowledge from information. The information is generated, stored, manipulated and consumed in a manner which improves data quality. Other organizations with activities close to openEHR are ASTM, CEN, HL7, IHE, ISO and IHTSDO. We have chosen openEHR as a case for study, because the openEHR system puts special emphasis on semantic interoperability to improve the quality of data exchanged among multiple organizations or within a single organization. The openEHR has an advantage over HL7

ver 3.0 and the associated reference information models. It takes into account the clinical workflows and operational contexts, thus ensuring interoperability across enterprises. The openEHR continues to develop standards by implementation testing rather than through committee work. Its standards are supplying both system specifications for data and archetypes for clinical phenomena [Pishev 2006]. Together, these facilitate a stable reference implementation and open source software for the users [openEHR 2009].

With openEHR, clinicians are not just passive users of openEHR-enabled software and systems, but actively determine the possible breadth, depth, and richness of patient data kept in EHR systems. This directly affects the quality of patient care through clinicians' pivotal role in creating, revising, and updating archetypes. Archetypes empower clinicians to influence how their EHRs will function. The openEHR community has a growing library of high-quality authored archetypes for use in clinical care and tools to support their maintenance, governance and release [CKM 2009]. The openEHR based EHR system has been implemented for hospitals (emergency department of Austin Health in Australia, and maternity care in a hospital in Cambodia) [Gok 2008; MOSS8 2009].

4.1 OpenEHR Specifications

The openEHR specifications have been developed to standardize the representation of an international electronic health record. The openEHR project [Beale and Heard 2008 a] deliverables include requirements, abstract specifications, implementation technology specifications (ITS), computable expressions and notations for conformance criteria (Figure 7). The abstract specifications consist of the Reference Model (RM), Archetype Model (AM) and the Service Model (SM) (Figure 8). RM represents the semantics of storing and processing in the system. It contains a set of generic data structures that are flexible enough to model most of the logical structures for knowledge representation (occurring in clinical records). AM contains the knowledge enabling environment by defining domain level structure and constraints on the generic data structures described in the RM. Thus AM describes the semantics of archetypes and templates, and their use within openEHR. At the user level, the openEHR service model includes definitions of basic services in the health information environment, centered around the EHR (Figure 8 and 9).

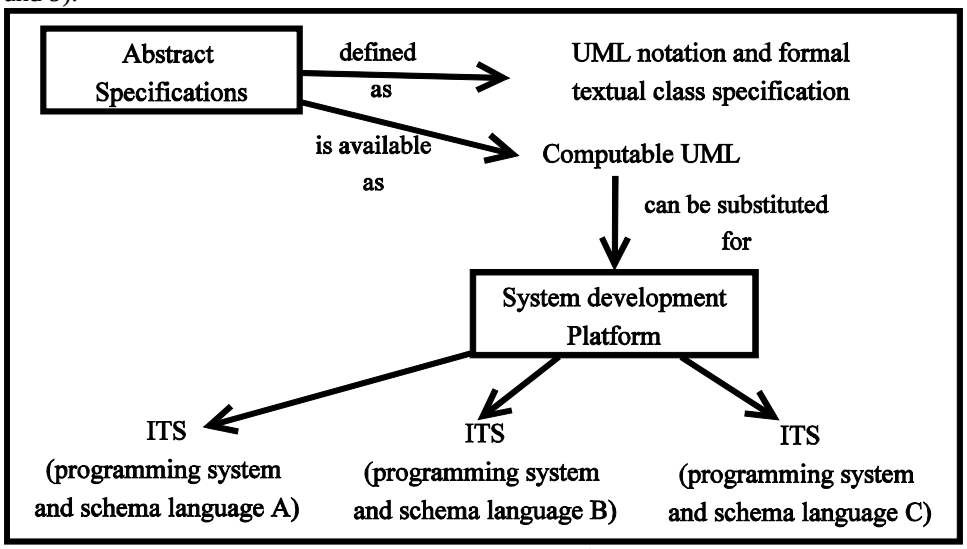


Fig. 7. openEHR Specifications

The above abstract specifications published by openEHR are defined using the UML

notation and formal textual class specifications [Beale and Heard 2008 a]. These are also available in a tool-oriented computable UML format in order to enable development of software and systems [Beale and Heard 2008 a] (Figure 7). The computable expressions for all practical purposes can be assumed as being a lossless representation of the published abstract specifications. The implementation technology specifications (ITS) on the other hand, correspond to the expression of abstract specifications in various programming and schema languages. The approach to implementing any of the openEHR abstract models in a given implementation technology is, to firstly define an ITS for the particular technology, then to use it by formally mapping the abstract models into expressions in that technology.

4.2 EHR Levels of Abstraction

The three levels of abstract specifications in EHR architecture are similar to those in Database Management System (DBMS) architecture [Silberschatz et al. 2005] (Figure 8 and Figure 20).

i) Physical Level: The lowest level of abstraction describes the details of reference model, such as identification, access to knowledge resources, data types and structures, versioning semantics, support for archetyping and semantics of enterprise level health information types.

ii) Logical Level: The conceptual level describes the clinical concepts that are to be stored in the system. These are represented in the form of archetypes and templates. A user of logical level does not need to be aware of their complexity. Clinical domain experts use logical level.

In common with object model classes, archetypes can be specialized, as well as composed (i.e., aggregated) [Beale and Heard 2005]. An example of composition archetype is openEHR-EHROBSERVATION.laboratory.v1. An archetype is a specialization of another archetype if it mentions that archetype as its parent, and only makes changes to its definition, such that its constraints are 'narrower' than those of the parent. (as in openEHR-EHROBSERVATION.laboratory-glucose.v1).

iii) View Level: The highest level of abstraction describes only a part of the entire EHR architecture depending upon the need. This corresponds to the service model. Several views may be defined. Users can see these views. In addition to hiding details of logical level for simplicity, the views also provide a security mechanism to prevent users from accessing certain parts of EHR architecture.

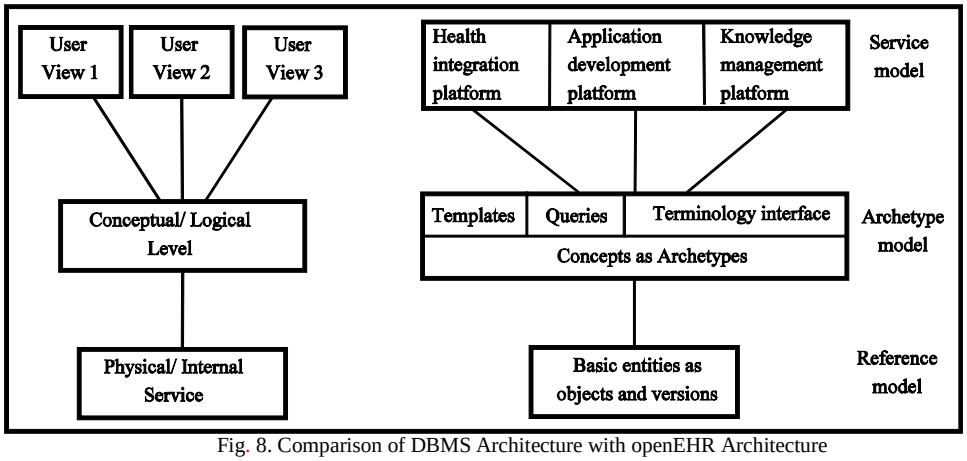


Fig. 8. Comparison of DBMS Architecture with openEHR Architecture

The architecture is similar to the DBMS environment. To obtain any information, there must be a significant amount of pre- and post- processing to decompose and

reconstruct information from the generic data structures. For example, to insert an instance of EHR information (e.g., a Blood Pressure reading) during runtime, the software layer must query and construct its corresponding archetype from an archetype repository, and then perform a comparison to make sure the data instance adheres to all constraints and rules imposed by the archetype.

4.3 The openEHR Model

The three level abstractions provide a macro level view of the EHR architecture. The design aim of openEHR is to provide a coherent, consistent and re-usable type system for health computing. The components within each level are described further. The 'core' of the Reference Model (RM) provides various common design patterns that can be reused ubiquitously in the upper layers of the RM, as well as in the Archetype Model (AM) and Service Model(SM) layers. Figure 9 illustrates the relationships between RM, AM and SM packages. Dependencies only exist from higher packages to lower packages.

Reference Model (RM):

As mentioned earlier, RM provides identification, access to knowledge resources, data types and structures, versioning semantics, and support for archetyping. The components within the RM are organized into three packages: core, pattern and domain (Figure 9). The core group package is generic. It is used by all openEHR models and in all the outer packages. The packages in the patterns and domain group define the semantics of enterprise-level health information types, including the EHR and demographics. These are described in the following paragraphs.

Core: The main component in this group is support. It consists of the 'definitions', 'identification', 'terminology' and 'measurement' packages. The semantics defined in 'support' allows all other models to use identifiers and to have access to knowledge services like terminology and other reference data. The use of standardized data types enhances the interoperability of low-level data semantics across systems.

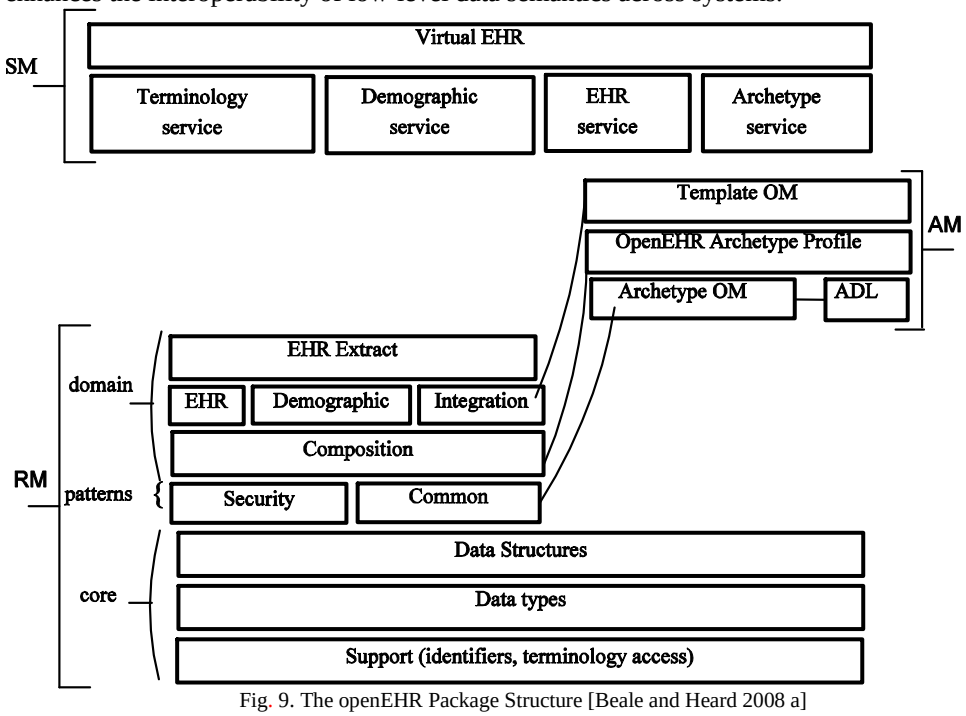


Fig. 9. The openEHR Package Structure [Beale and Heard 2008 a]

Pattern: Components include: Security and Common. The Security Information Model

defines the semantics of access control and privacy setting for information in the EHR. The Common Information Model (IM) contains classes (such as LOCATABLE and ARCHETYPED) that provide the link between information and archetype models.
Domain: The EHR IM defines the containment and context semantics of the concepts EHR, COMPOSITION, SECTION and ENTRY. Components within this 'domain' are described in section 4.4. The EHR Extract IM defines how an EHR extract is built from COMPOSITIONs, demographic, and access control information from the EHR. The Integration model defines the class GENERIC_ENTRY, a subtype of ENTRY used to represent free form legacy or external data as a tree. The demographic model defines generic concepts of PARTY, ROLE and related details such as contact addresses.

Archetype Model (AM):
The openEHR AM package contains the models necessary to describe the semantics of archetypes and templates, and their usage within openEHR. The openehr_profile package defines a profile of the generic archetype model defined in the archetype package, for use in openEHR (and other health computing endeavors). Further details of this level are discussed in section 5.

Service Model (SM):
The openEHR service model includes definitions of basic services centered around the EHR. These service sets will be evolving with time. Some details of the service model are discussed in Section 7. Its components are described immediately below (refer also to Figure 9).
(i)The virtual EHR Application Programming Interface (API) defines the interface to EHR data, at the level of Compositions and below. It allows an application to create new EHR information, and to request parts of an existing EHR and modify them. This API enables fine-grained archetype-mediated data manipulation. Changes to the EHR are committed via the EHR service.
(ii)The EHR service model defines the coarse-grained interface to electronic health record service. It also defines the semantics of server-side querying, i.e., queries which cause large amounts of data to be processed, generally returning small aggregated answers, such as averages, or sets of identifications of patients matching a particular criterion.
(iii)The archetype service model defines the interface to online repositories of archetypes.
(iv)The terminology interface service provides the means for all other services to access any terminology available in the health information environment. The terminology service is the gateway to all ontology- and terminology-based knowledge services in the environment.
(v)The demographic service provides all services regarding the demographic information so as to provide privacy and security.

4.4 Detailed EHR Reference Model Architecture

4.4.1 EHR Extract Information Model

Each content item potentially contains a whole hierarchy of information items, only some of which is generally of interest to the Requestor. The typical database idea of a "query result" is usually expected to return only such fine-grained pieces. Clinician or software may need to obtain some or all of a single patient's EHR. A hospital or clinic may require from a laboratory results of testing done for multiple patients. The openEHR Extract supports detailed access to the versioned view of data (Figure 9 and Figure 10(b)). Information transferred in an EHR Extract needs to be self-standing in the clinical sense, i.e. it can be understood by the requestor without assuming any other means of access to the responding system.

As the topmost layer in RM, the EHR extract information model defines architecture for communication of EHR extracts, or documents (Figure 10(a) and 10(b)). It is prescribed under 'domain' in RM as 'EHR extract' (Figure 9).

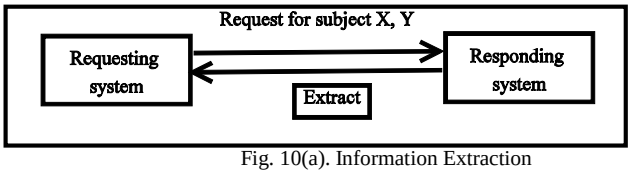


Fig. 10(a). Information Extraction
(Requesting System and Responding System)

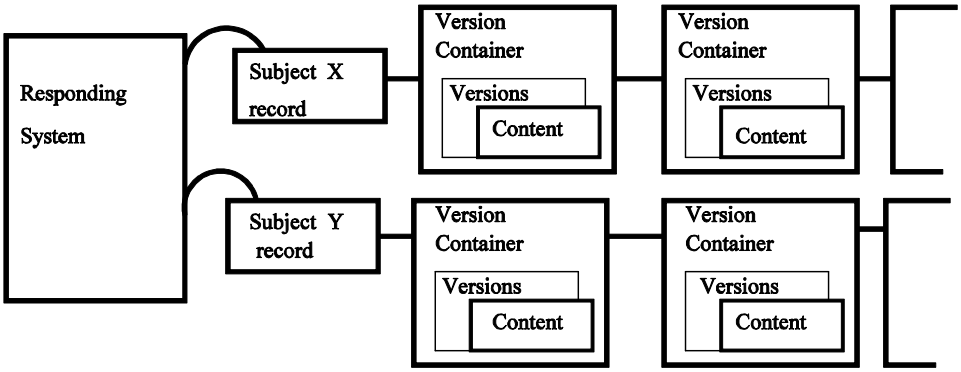


Fig. 10(b). Operational openEHR Environment for Extracts based on [Beale and Frankel 2007]

The Responding system contains one or more subject records. Each subject record consists of one or more version containers, each of which contains the version history of a particular piece of content. Each version corresponds to the state of a particular content item at some point in time, when it was committed by a user. Contribution corresponds to the set of Versions (each from a different version container) committed at one time, by a particular user to a particular system. For example, a patient may have EHR both at clinic and home PC. Whenever changes are made at either place, it is possible to copy just the required changes (copying Contributions) to the device since the last synchronization, thus enhancing the quality of the information system.

4.4.2 The openEHR EHR Information Model

This model is a part of RM (domain) which defines a logical EHR information architecture rather than just architecture for communication of EHR extracts or documents between EHR systems. The Package structure of openEHR EHR contains the elements ehr, compositon, and content (Figure 11).

ehr

The EHR consists of distinct, coarse-grained items known as compositions added over time and organized by Folders. Each composition consists of Entries, organised by Sections within the composition (Figure 11). The audit information for each context is recorded at the corresponding level of the EHR.

The root EHR object records three pieces of information that are immutable after creation: the identifier of the system in which the EHR was created (system_id), the identifier of the EHR (distinct from any identifier for the subject of care) (ehr_id), and the time of creation of the EHR (time_created). It acts as an access point for the component parts of the EHR. It contains the versioned objects by references. This package contains the top level structure, the EHR (the root object, identified by a globally unique EHR identifier), which consists of an EHR_ACCESS object (containing access control settings

for the record), an EHR_STATUS object (containing various status and control information, optionally including the identifier of the subject (i.e. patient) currently associated with the record), versioned data containers in the form of VERSIONED_COMPOSITIONs (containers of all clinical and administrative content of the record), optionally indexed by a hierarchical directory of FOLDERs (which contain compositions by reference). A collection of CONTRIBUTIONs is also included, which document the changes to the EHR over time.

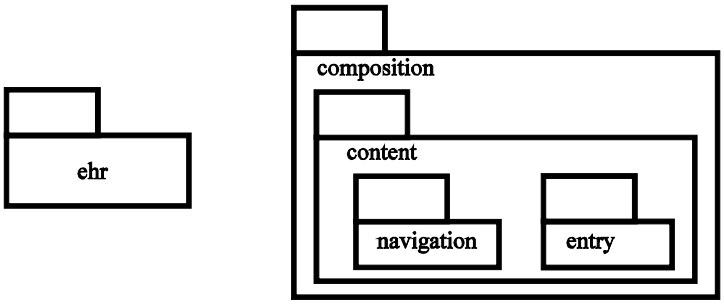


Fig. 11. Package Structure of openEHR EHR Information Model [Beale et al. 2008]

composition

The composition is the EHR's top level "data container". It is described by the COMPOSITION class. The main data of the EHR are found in its compositions (Figure 11). There are two general categories of information at the coarse level which are found in an EHR: event items, and persistent items. Events record what happens during healthcare system events (with or for the patient), such as patient contacts. These also record sessions in which the patient is not a participant or not present (e.g., pathology testing). There are various kinds of persistent (longitudinal) items. Persistent Compositions record items of long-term interest in the record. They can be thought of as proxies for the state or situation of the patient. These provide a picture of the patient at a point in time. These are maintained by clinicians. A persistent composition is known as a continuant, whereas event compositions record occurrences, (i.e., things that were true or did happen but have no longevity). In any clinical session, an event composition will be created, and in many cases, persistent compositions will be modified (Figure 12).

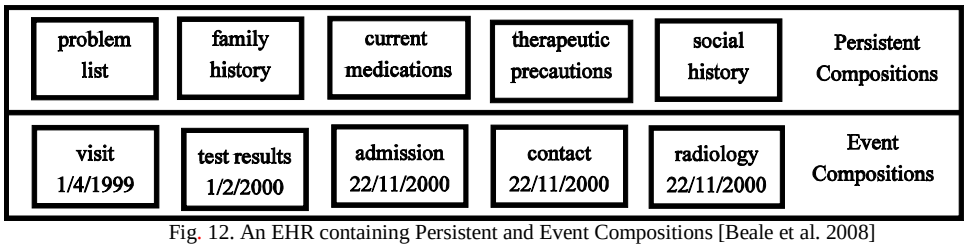


Fig. 12. An EHR containing Persistent and Event Compositions [Beale et al. 2008]

The composition concept in the openEHR's EHR originated from the Transaction concept of the GEHR project. It was based on the concept of a unit of information corresponding to the interaction of a healthcare agent with the EHR. It was designed to satisfy the ACID characteristics [Silberschatz et al. 2005] along with indelibility, modification and traceability.

The key information in a COMPOSITION is found in its content, context, and composer attributes.

content

The content package contains the CONTENT_ITEM class, ancestor class of all content types, and the navigation and entry packages, which contain SECTION, ENTRY and related types. The classes in the package describe the structure and semantics of the

contents of compositions in the health record.

a) navigation: The SECTION class provides a navigational structure to the record, similar to "headings" in the paper record. ENTRYs and other SECTIONs can appear under SECTIONs. Sections provide both a logical structure for the author to arrange entries, and a navigational structure for readers of the record. The main benefit of Sections is that they may provide significant performance benefits to querying by automated systems.

b) entry: This package contains the generic structures for recording clinical statements. All information which is created in the openEHR health record is expressed as an instance of a class in the entry package, containing the ENTRY class and a number of descendants. An ENTRY instance is logically a single 'clinical statement'. It may contain a significant amount of data, e.g., a microbiology result, a psychiatric examination, a complex prescription. In terms of clinical content, the entry classes are the most important in the openEHR EHR Information Model. These define the semantics of all the 'hard' information in the record. These are intended to be archetyped.

The design of Entry package is based on the Clinical Investigator Recording process as shown in Figure 13. The observation, evaluation, instruction and action cycle for building the elements of EHR is analogous to continuous data quality improvement by following the cycles of define, measure, analyze and improve [Madnick et al. 2009].

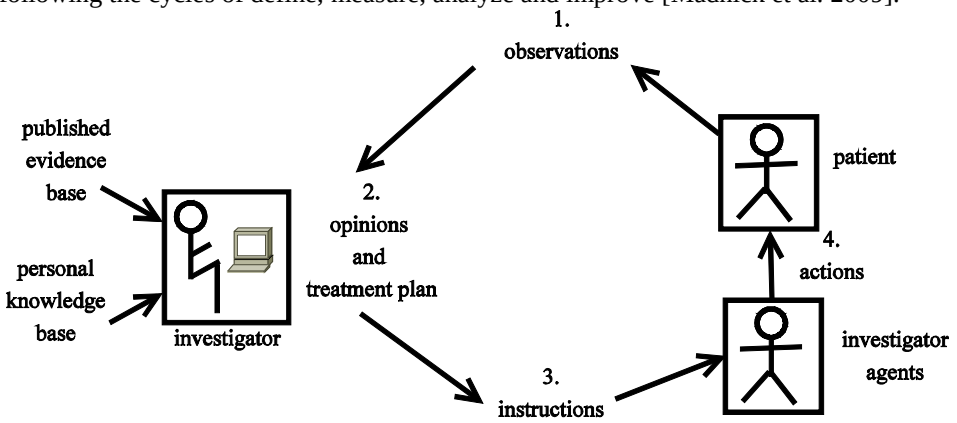


Fig. 13. Clinical Investigator Recording Process [Beale and Heard 2008 a]

The details of generic structures within the ENTRY package are shown in Figure 14. Entry types include CARE_ENTRY. This contains information that relates to care process. OBSERVATION includes all observed phenomena, including mechanically or manually measured, and responses in interviews. EVALUATION includes assessments, diagnoses, and plans. INSTRUCTION contains actionable statements such as medication orders, recalls, monitoring, and reviews. ACTION contains information recorded as a result of performing Instructions. Consider a few examples.

Example of contents in ENTRY package: Under entry, in the EHR information model, (Figure 14), the CARE_ENTRY class is an abstract precursor of classes that express information of any clinical activity in the care process around the patient. The CARE_ENTRY type includes two attributes particular to all clinical entries, namely protocol and guideline_id, which allow the "how" and "why" aspects of any clinical recording to be expressed. Also the ADMIN_ENTRY is used to capture administrative information. Administrative information is created by staff. It expresses details to do with coordinating the clinical process. These include admission information, appointments, discharge/dismissal, billing and insurance information.

Example of contents in CARE_ENTRY: The clinical information about Problem list is recorded inside persistent composition. It is maintained as one or more 'Evaluations'

(which are generated by clinicians), as a result of 'observations' made. The clinical information about referral is recorded inside event composition and is recorded as 'instructions'. The concepts in these examples are defined in terms of archetypes of entry and other reference model types in openEHR.

Further 'Actions' are interventions whereas 'Observations' record only information relating to the situation of the patient (not what is done to him/her). Observation is expressed in terms of "data", "state" and "protocol" (Figure 3) as shown in Table I.

Table I. Observation expressed as data, state and protocol.

Data	Expressed in the form of a History of Events. It can be List, Table, Single (value), or Tree.
State	Information about the state of the Entry (necessary to correctly interpret the data).
Protocol	Details of how the observation was carried out.

The 'time' in the Observation category has a linear historical structure, whereas in Instruction it has a branching, potentially cyclic structure. Time is used for all kinds of statements which evaluate other information, such as interpretations of observations, diagnoses, differential diagnoses, hypotheses, risk assessments, goals and plans. It has attribute data in the form of a spatial data structure.

'Instruction' is used to specify actions in the future. It enables simple and complex specifications to be expressed, including in a fully-computable workflow form. 'Activity' defines a single activity within an Instruction, such as a medication administration. 'Action' is used to record a clinical action that has been performed, which may have been adhoc, or due to the execution of an Activity in an Instruction workflow (recorded as attribute 'instruction_details').

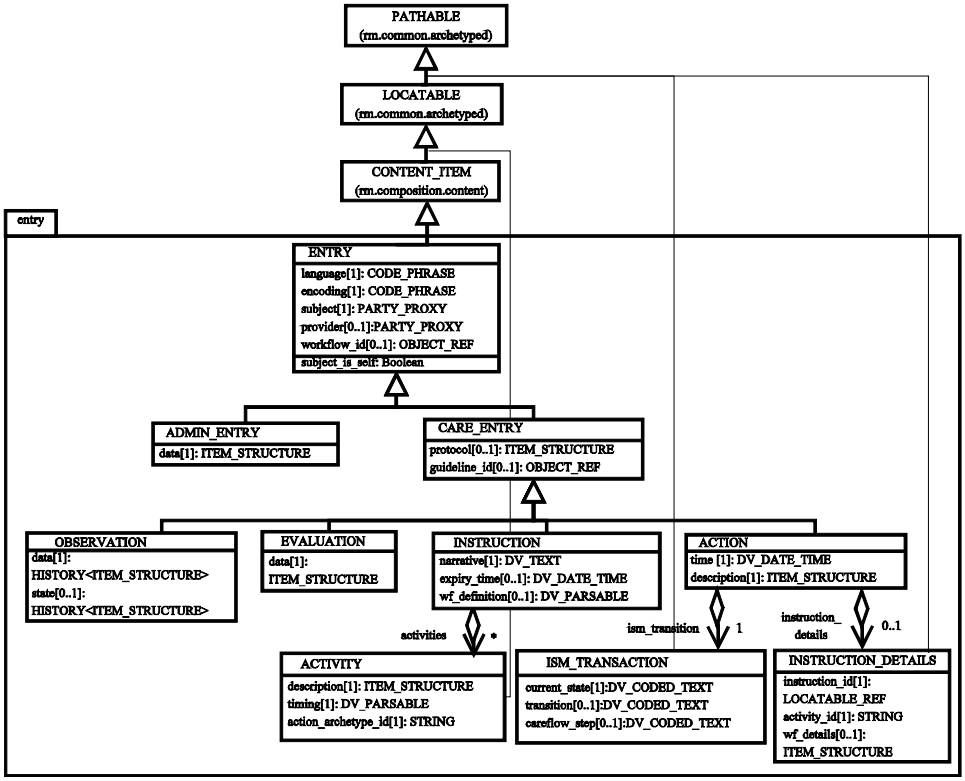


Fig. 14. The rm.composition.content.entry Package (in UML) [Beale et al. 2008]

4.4.3 Demographic

The Demographic IM defines demographic information. The general approach of openEHR is to enable the complete separation of demographic (particularly patient-identification information) from health records. This is in the interests of privacy (in some cases required by national legislation) and separated data management. Thus, the demographic information regarding a patient is kept separately from the medical record of a patient. It is shared if the patient agrees to share it. This helps in maintaining quality.

5. ARCHETYPES IN A KNOWLEDGE-BASED SYSTEM

5.1 Introduction to Archetypes

Archetypes specify the design of the clinical data. As per the concise Oxford Dictionary, an archetype is "an original model, prototype or typical specimen". Knowledge representation of clinical concepts is through archetypes which enable semantic interoperability of heterogeneous systems. These define data quality constraints to be placed on the organisation and the content of record entries. An archetype defines a data structure, including optionality and multiplicity, data value constraints, and relevant bindings to natural language and terminology systems. An archetype might define or constrain relationships between data values within a data structure. These are expressed as algorithms, formulae or rules. Its metadata defines its core concept, purpose and use, evidence, authorship and versioning. An archetype ensures maximal dataset. It contains all the relevant information regarding a clinical concept. For example, the archetype for blood pressure contains all the relevant information (the data, state and protocol parts) (Figure 3).

Consequently, there are checks on quality of information for every clinical concept. For example, an archetype on blood pressure measurement would comprehensively and formally describe what a clinician needs to know about a measurement, including its clinically safe interpretation [Beale and Heard 2005]. High quality archetypes with high-quality clinical content are the key to semantic interoperability of clinical systems [Bisbal and Berry 2009]. Archetypes incorporate an ontology section. These domain-specific ontologies provide a common semantics for all shared data.

Different archetypes are instances of an archetype model. The Archetype Object Model is an object-oriented data model. An object can be specialized as well as composed (aggregated). An archetype may logically include other archetypes, and may be a specialisation of another archetype. Thus, these are flexible and vary in form. In terms of scope, these are general-purpose, re-usable and composable. Archetypes are separate from the data, and are stored in archetype repositories. The archetype repository at any particular location will include archetypes from well-known online archetype libraries.

The function of archetypes and templates at runtime is to facilitate data validation (at data capture or import time). These guarantee that elements of data conform to the reference model (and to the archetypes themselves) [Beale and Heard 2005]. Data validation with archetypes is mediated by the use of openEHR templates. Figure 15 illustrates the relationships between EHR and archetypes. Thus, the clinical user can share data with other Health Record Systems (HRS's) through archetypes.

5.1.1 Archetypes and Data Validation

By design, archetypes incorporate rules. Data entered into an archetype-enabled system will only be captured if it fits those rules. This feature improves the data quality of a system considerably, and archetype-powered searches or queries on the EHR are able to find specific data. Similarly, archetyped data is able to be viewed consistently and reproducibly, no matter where they appear within a single EHR or within any number of EHR systems. Some of the rules that clinicians can set within archetypes to make

information captured or viewed fit their requirements include:

- The maximum and minimum value of a measurement, e.g., not allow a pulse rate that is less than zero;
- The allowed units of measurement, e.g. weight in gm or kg;
- The appropriate set of terms from a terminology, e.g. a set of blood groups including the associated rhesus typing;
- An internal value set that is allowed for an element, e.g. in a subjective assessment of blood loss there may be the options of 'none', 'light', 'normal', and 'heavy'; and
- Establish whether a piece of clinical data is required (or optional).

These rules constrain data entry, thus improving data quality considerably. Some of the key data quality problems such as missing values, syntax violation, domain violation, existence of synonyms, heterogeneity of syntaxes and heterogeneity of measure_units are taken care of by archetypes.

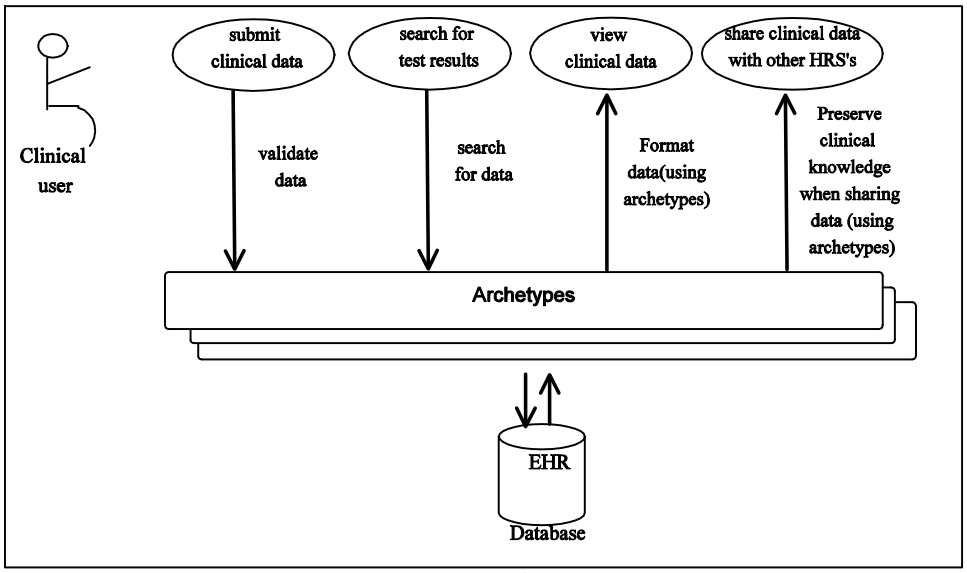


Fig. 15. All Functions use Archetypes to refer to the Clinical Data in the EHR Systems

5.2 Major Categories of openEHR Archetypes

With reference to openEHR specifications, an archetypable data instance in an EHR is a composition, a section, an entry or an item structure. According to the openEHR reference model [Beale et al. 2008], there are five sorts of entries and four types of item structures which form categories within archetypes in openEHR EHRs (Figure 16) [Thurston 2006]. Every type of clinical knowledge (information) can be mapped to one of these categories.

1. **Composition** (or document) - this contains information committed to the EHR by a clinician. Compositions contain Sections - or organizing classes, which themselves contain Entry. Examples of Compositions are documents created by a clinician and stored in the EHR, laboratory results and report, ECG results and report, a problem list which is separate from any particular document, and a family history list which is separate from any particular document.
2. **Section** - this allows information within a composition to be organized. Sections provide both a logical structure and a navigational structure. Sections are archetyped in trees with each tree containing a root section, one or more sub-sections, and any number of Entries at each node.

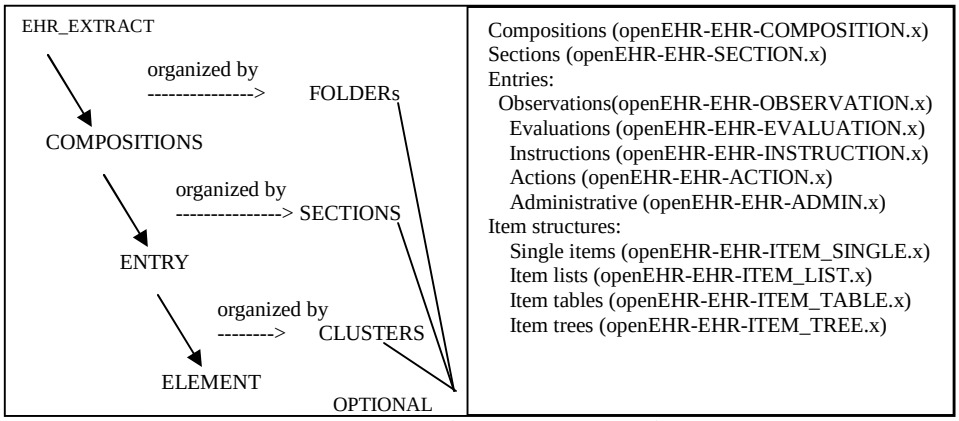


Fig. 16. Hierarchical Structure of EHR and Categories of openEHR Archetypes

3. **Entry** – An entry is like the leaf node of the document and contains information, such as blood pressure, assessment, diagnosis, and medication order. The 'Entries' can be interpreted independent of the composition (or the section) within which they are located. This is a key principle of the openEHR methodology. It is very important for automatic processing of EHR data. An ENTRY instance is logically a single 'clinical statement'. It may be a single short narrative phrase, and may also contain a significant amount of data, e.g., a microbiology result, a psychiatric examination, a complex prescription. In terms of clinical content, these define the semantics of all the 'hard' information in the record. Archetypes for Entries make up the vast majority of important clinical archetypes defined.

For example, Entry can be a 'care entry' or an 'admin entry' (relating to similar entities in RM). Components of 'care entry' are described in the following paragraphs.

- Evaluation: Evaluations are 'meta-observations' - ideas, labels or views which arise within the clinician's mind. E.g, diagnosis, assessment, problem, issue, adverse reaction.
- Observation: Observations are the 'uninterrupted' or raw information – based on the clinical observations. An observation entry contains data (core information), state and protocol as shown in blood pressure archetype (Figure 3). In addition to the above mentioned parts, it also contains a history. E.g. blood pressure, ECG, weight and height.
- Instruction: Instructions are statements about what should happen in the future. These consist of two parts: an action specification, which is a composition or an observation archetype, and information about the timing and status of the action to be completed. For example, a medication order is recorded as an instruction, whereas a medication administration is recorded as an observation.

- Actions: These are clinical activities concerning procedure or medication administration. Actions can record the ensuing state of the instruction, such as 'completed' or 'cancelled'.

4. **Item Structures**- An EHR requires structured data in the form of single values (e.g., weight, height), or, as lists of values (e.g., blood test results), or, as tables (e.g., visual acuity results) or trees (e.g., biochemistry results) of values.

5.3 Archetypes and Data Quality

Archetypes are standalone entities and can be created, shared, reused, specialized, revised and versioned. These are both language and terminology independent. These provide knowledge level interoperability. Thus, systems reliably communicate with each other at

the level of knowledge concepts. Archetypes contain validation rules for all the data entered into systems, and thus help in preventing errors in health records. These can help in epidemiological and other public health research functions. There will be a transformation in clinical care and research by virtue of the ability to work from shared knowledge framework. Thus, the archetypes are capable of providing intelligent generic decision support programs. These provide future-proof (designed not to be obsolete in the future) EHR systems to enhance patient care and safety over the course of a lifetime.

Archetypes aid in quality of data as they are defined by clinical domain experts and not by the software experts for their use, and thus provide maximal information regarding a clinical concept [Leslie 2006]. These are measured and analyzed by the clinical review board before being published, .i.e., these are developed through a domain knowledge governance tool (Clinical Knowledge Manager) [CKM 2009]. These can be improved, as there is a process of versioning of archetypes. With expansion of knowledge, one can have a new version of the archetype or a new specialization as per the specialization property of the archetype (see section 5.5.2).

5.4 Terminology and Archetypes

To share information (for computers rather than humans) we need to share two things - a schema or rules for how the information is stored, and a domain vocabulary for each point in the schema.

The openEHR approach is to have a single logical schema for all systems. It is attuned to the European approach (CEN 13606) - which allows storage and retrieval of potentially infinitely complex information. This means that everyone can receive data (as atomic information - not text) from everyone else. To add meaning to the data, it must conform to a second 'schema' or set of rules - held in files which can be shared - called 'archetypes'. For computers to understand the data it must be compliant with at least one archetype (this means, as a specialization of one of the archetypes).

This simplifies the requirements for terminology, as we now have shared data points that require a specific vocabulary. It is appropriate to describe these within the archetype itself - as there is no useful classification of small domain vocabularies used for one data point. This has a number of advantages:

- The domain vocabularies can be safely translated as it is quite clear what the context of the term is within the archetype.
- These vocabularies can be 'bound' to the different terminologies used within systems (there are many such cases).
- All of these things can be done after the fact - that is to say, there is no need to have access to a terminology to create an archetype.

Terminologies, coding and classification systems are for structured data collection. By providing a basis for semantic level interoperability and a common language, these facilitate the exchange of information among different applications. They remove ambiguity from information and language dependence and also enable proper automatic processing such as medical decision support. Some data points within an archetype (via a constraint definition and binding) point to an external terminology. OpenEHR and International Health Terminology Standards Development Organization (IHTSDO) collaborate to explore how clinical terminologies and archetype-based record structures can best be aligned to support electronic health records [IHTSDO collaboration 2009].

5.5 Detailed Description-Archetype Definition Language (ADL)

Archetype Definition Language (ADL) is a formal language for expressing archetypes,

which are constraint-based models of domain entities, or 'structured business rules'[2]. It has evolved on notions similar to those of KML by Google Maps API, or the use of XML for web documents or databases. Available support such as from XML alone is, however, not enough for expressing healthcare objects [Sokolowski 1999]. Table II shows the comparison between ADL and XML.

Table II. Comparison between ADL and XML

Properties	ADL	XML
Machine Processable	Yes	Yes
Human Readable	Yes	Sometimes unreadable (e.g., XML-schema instance, OWL-RDF ontologies)
Leaf data Types	More comprehensive set, including interval of numerics and date/time types	String data; with XML Schema option- more comprehensive set
Structure	Universal schema for temporal database (EHRs) (history database)	Semi-structured data (rooted acyclic graph with unique path from root to leaf)
Adhering to object-oriented semantics	Yes, particularly for container types	XML schema languages do not follow object-oriented semantics
Ontological reference	Domain entities/ archetypes	Global terms/ concepts
Representation of object properties	Uses attributes	Uses attributes and Sub-elements
Space (for storage)	Uses nearly half of space for tags	May have data redundancy in contents
Efficiency	Is a domain specific language (sufficiently rich to capture and model medical domain)	Good for web document modeling with limited ability to represent database contents

Every ADL archetype is written with respect to a reference model. Archetypes are applied to data via the use of templates, which are defined at a local level. The openEHR Archetype Object Model (AOM) [Beale 2008] describes the definitive semantic model of archetypes, in the form of an object model. The AOM defines relationships which must hold true between the parts of an archetype for it to be valid as a whole. The ADL syntax is one possible serialization of an archetype. ADL and AOM have been adopted by CEN TC/251, the European standards agency Health Telematics Committee for use in its revised EN 13606 Electronic Health Record standard.

Previously, archetypes have been expressed as XML instance documents conforming to W3C XML schemas, for example, in the Good Electronic Health Record [GEHR] and openEHR projects. XML archetypes are equivalent to serialised instances of the parse tree, i.e., particular ADL archetypes serialized from objects into XML instances. Archetypes connect information structures to formal terminologies. They are completely

[2] Business rule is a widely-used documentation concept for conceptual schemas. Business rules are used to describe the properties of an application, e.g., the fact that an employee cannot earn more than his or her manager. A business rule can be:

_ The description of a concept relevant to the application (also known as a business object),

_ an integrity constraint on the data of the application,

_ a derivation rule, whereby information can be derived from other information within a schema.

path-addressable in a manner similar to XML data, using path expressions that are directly convertible to Xpath expressions. With ADL parsing tools, it is possible to convert ADL to any number of forms, including various XML formats. XML instances can be generated from the object form of an archetype in memory. An XML-schema corresponding to the ADL Object Model has been published at openEHR.org [openEHR 2009].

Example of XML/ADL use in openEHR: In order to accept a report from pathology laboratory for inclusion in EHR repository of a patient (in the ADL form), an XML form is generated, using the archetype. This form is shared with the laboratory for on-site validation of data input. Thus, XML is used as an input and transport medium.

5.5.1 Organization of ADL

In serialized form, archetypes are represented in Archetype Definition Language (ADL) [Beale and Heard 2008 b], and in XML based serializations [Beale and Heard 2008 a]. ADL is an abstract language based on Frame Logic queries (also known as F-logic) with the addition of terminology. F-logic is a knowledge representation and ontology language. It accounts in a declarative fashion for structural aspects of object-oriented and frame-based language. An ADL archetype is a guaranteed 100% lossless rendering of the semantics of any archetype, and is designed to be a syntactic analogue of the AOM. Thus, ADL is a textual language for specifying constraints on data instances of an RM in a formal way [Beale and Heard 2008 b]. An archetype expressed in ADL is composed of four main parts: header, definition, ontology and revision history (Figure 17). The header section contains the archetype metadata. In the definition section, the modeled clinical concept is represented in terms of a particular RM class. This description is built by constraining several properties of classes and attributes, such as existence, occurrences or cardinality, or by constraining the domain of atomic attributes. In this section, only those entities appear that need to be constrained. In the ontology section, the entities defined in the definition section are described and bound to terminologies. Finally, the revision history section contains the audit of changes to the archetype.

5.5.2 Structure of Archetype in ADL

ADL uses three other syntaxes, cADL (constraint form of ADL), dADL (data definition form of ADL), and a version of first-order predicate logic (FOPL), to describe constraints on data which are instances of some information model (e.g., expressed in UML) [Beale and Heard 2008 b]. Thus, ADL can be used to write archetypes for any domain where formal object model(s) exist, which describe data instances. Further, when archetypes are used at runtime in particular contexts, these are composed into larger constraint structures, with local or specialist constraints added, via the use of templates. The formalism of templates is presented by using dADL. The cADL syntax is used to express the archetype definition, while the dADL syntax is used to express data which appears in the language, description, ontology, and revision_history sections of an ADL archetype. The various keywords in ADL are - archetype, specialise/specialize, concept, language, description, definition, invariant, ontology. The top-level structure of an ADL archetype is shown in Figure 17. Its components are described in the following paragraphs. Please see sample example (Example of ADL Archetype Structure).

A) Header Section

The Header Section consists of archetype, ‘specialize’, concept, language and language translations, and description sections. Archetype section introduces the archetype and must include an identifier. The ‘specialize’ section indicates that the archetype is a specialization of some other archetype, whose identity must be given (only one specialization parent is allowed, i.e. an archetype cannot ‘multiply inherit’ from other archetypes).

All archetypes represent some real world concept, such as “patient”, “blood pressure”, or “antenatal examination”. The concept is always coded, ensuring that it can be displayed in any language the archetype has been translated to. There can be only one original_language. The language section includes data describing the original language in which the archetype was authored (essential for evaluating natural language quality), and the total list of languages available in the archetype. The translations list must be updated every time a translation of the archetype is undertaken. The description section of an archetype contains descriptive information (“meta-data” i.e., items that can be used in repository indexes and for searching). The dADL syntax is used for the description. dADL is a formal means of expressing instance data based on an underlying information model.

B) Definition Section

The definition section contains the formal definition of the archetype. It is written in the Constraint Definition Language (cADL). For example, for ‘blood pressure’, the definition expresses constraints on instances of the types ENTRY, HISTORY, EVENT, ITEM_LIST, ELEMENT, QUANTITY, and CODED_TEXT (RM classes) so as to allow them to represent a blood pressure measurement.

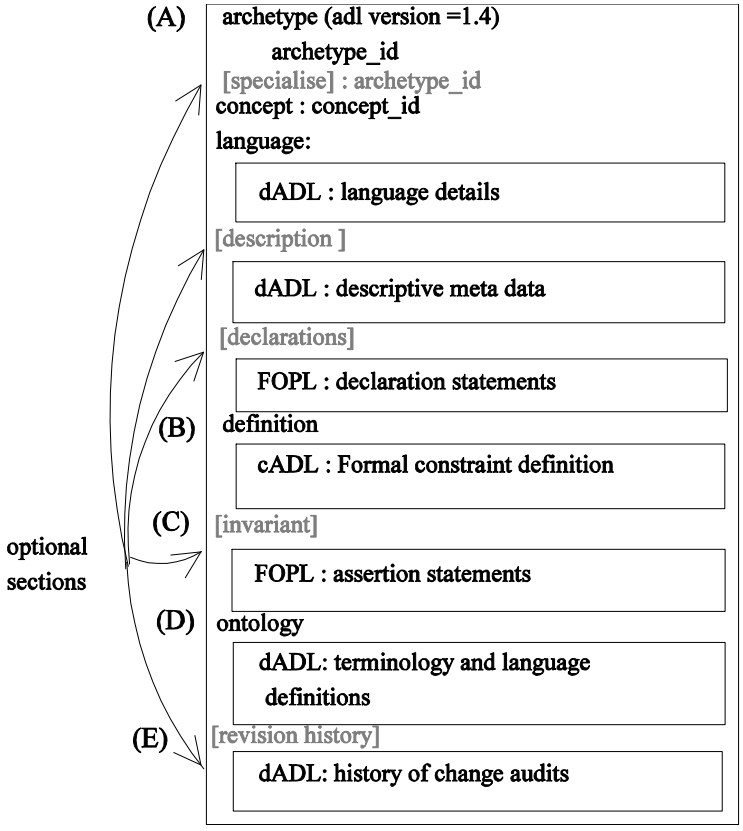


Fig. 17. ADL Archetype Structure [Beale and Heard 2008 b]

C) Invariant Section

The invariant (or rules section) introduces assertions. These relate to the entire archetype. These are used to make statements which are not possible within the block structure of the definition section. Any constraint which relates more than one property to another is

in this category. Most constraints containing mathematical or logical formulae fall in this category. These are expressed in cADL. It may be a first-order predicate logic statement which can be evaluated to a boolean result at runtime. Objects and properties are referred to by using paths. The following example states that the 'speed in kilometers' of some node is related to the 'speed in-miles' by a factor of 1.6:

```
invariant = (aadl) <#
        validity: /speed[at0002]/kilometres/magnitude =
                /speed[at0004]/miles/magnitude * 1.6
        #>
```

D) Ontology Section

The ontology section is expressed in dADL. Exploiting health ontology can enhance the quality of the system. This section defines codes representing node IDs, constraints on terms, and bindings to terminologies. Linguistic language translations are added in the form of extra blocks keyed by the relevant language. It contains both a term definition section and a term binding section.

a) Term definition section of ADL

This section is where all archetype local terms (that is, terms of the form [atNNNN]) are defined. Example of ADL Archetype Structure (under the ontology part) shows an extract from the English term definitions for the archetype local terms. Each term is defined using a structure of name/value pairs, and must at least include the names "text" and "description", which are akin to the full definition found in terminologies like SNOMED-CT.

b) Term-binding section of ADL

This section is used to describe the equivalences between archetype local terms and terms found in external terminologies, such as SNOMED–CT or Unified Medical Language System (UMLS). The purpose is to allow the query engine to determine an equivalent from within the archetype. This part has been omitted in Example of ADL Archetype Structure. However, in the following example for archetype (apgar score), an external term is bound directly to an archetype local term, and the binding holds globally throughout the archetype [Beale and Heard 2008 b]. Thus, cardiac score has local term "at0004" bound to UMLS term "umls::C234305".

```
term_bindings = <
["umls"] = <
items =<
["at0000"] = <[umls::C124305]> -- apgar result
["at0002"] = <[umls::0000000]> -- 1-minute event
["at0004"] = <[umls::C234305]> -- cardiac score
["at0005"] = <[umls::C232405]> -- respiratory score
["at0006"] = <[umls::C254305]> -- muscle tone score
["at0007"] = <[umls::C987305]> -- reflex response score
["at0008"] = <[umls::C189305]> -- color score
["at0009"] = <[umls::C187305]> -- apgar score
["at0010"] = <[umls::C325305]> -- 2-minute apgar
["at0011"] = <[umls::C725354]> -- 5-minute apgar
>
> >
```

E) Revision History

The revision history section of an archetype shows the audit history of changes to the archetype, and is expressed in dADL syntax.

5.5.2.1 Example of ADL Archetype Structure

In the following example, notion of 'patient' is defined in terms of constraints on a

generic model of the concept PARTY (Figure 18).

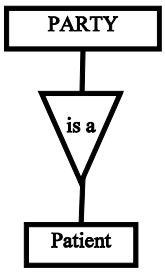


Fig. 18: Specialization- Patient is a PARTY (sub-part of Figure 6)

```
archetype (adl_version=1.4)                                   Archetype section
        adl-test-party.patient.draft
concept
        [at0000]                                              -- patient
language
        original_language = <[iso_639-1::en]>
definition
        PARTY[at0000] matches                 {
          details matches {                                   -- details
          address matches {[a-zA-Z0-9_]+)*}                   -- alphanumeric ok
          identity cardinality matches {1..*} matches {
            PARTY_IDENTITY[at0001] matches {                  -- demographic details
              name matches      {[local::at0002] }            -- patient's name
               contact matches     {[local::at0003]}          – patient's contact
                                                    } }
          relationships cardinality matches {0..*; ordered} matches {
            PARTY_RELATIONSHIP [at0004] matches { } -- patient relationships
                          }                 }                       }
ontology
        term_definitions = <
          ["en"] = <
            items = <
              ["at0000"] = <
                          text = <"patient">;
                          description = <"patient's data">
                        >
              ["at0001"] = <
                          text = <" Demographic details ">;
                          description = <" A patient's demographic details ">
                        >
              ["at0002"] = <
                          text = <"name">;
                          description = <" A patient's name ">
                        >
              ["at0003"] = <
                          text = <"contacts">;
                          description = <" A patient's contact">
                        >
              ["at0004"] = <
                          text = <"Relationships">
```

```
                description = <"A patient's relationships, especially family ties.">
                    >
>   >   >
```

5.5.3 Comparison with structure of XML

The structure of ADL is somewhat similar to XML.

i) ADL Paths

The ADL path syntax is semantically a subset of the Xpath query language for XML data. The ADL path includes a few syntactic shortcuts to reduce the verbosity of the most common cases. Xpath differentiates between "children" and "attributes" sub-items of an object. In ADL, there is no such distinction, and all subparts of any object are referenced in the manner of Xpath children.

ADL paths are absolute or relative with respect to the document in which they are mentioned. Absolute paths commence with an initial slash (/) character. Movable path patterns (that can be used to find a section anywhere in a hierarchy) are indicated with a leading double slash (//) as in Xpath.

The details of an archetype in ADL with dADL, cADL and assertion language are given in appendix A. The example illustrates that the ADL formal model facilitates conversion to the XML form.

6. EHR DATA ENTRY AND VALIDATION - A DQ PERSPECTIVE

This section provides as an example solution how the two-level model provides data quality. Data entry and validation ensures accurate and consistent data. The data can be entered into the EHR repository, if and only if, it can satisfy the constraints defined in templates and archetypes. For example, the unit of temperature which can be entered into the EHR system can be either degree Celsius or degree Fahrenheit and cannot be any other unit. Also, the archetypes and templates refer to the terminology which is standardized (e.g. the SNOMED-CT). This has an impact on the data quality (Figure 19).

The mechanism of data entry and validation in EHR systems does not resemble other existing common information systems. Before capturing the data, it must be validated by archetypes. For example, in the data entry regarding BP parameter, a unit of measurement must conform to 'mm/Hg'. For the purpose of data entry and validation, the archetypes are used at runtime by templates. The template is a directly usable definition which composes archetypes into larger structures often corresponding to a screen form, document, report or message [Beale and Heard 2008 a].

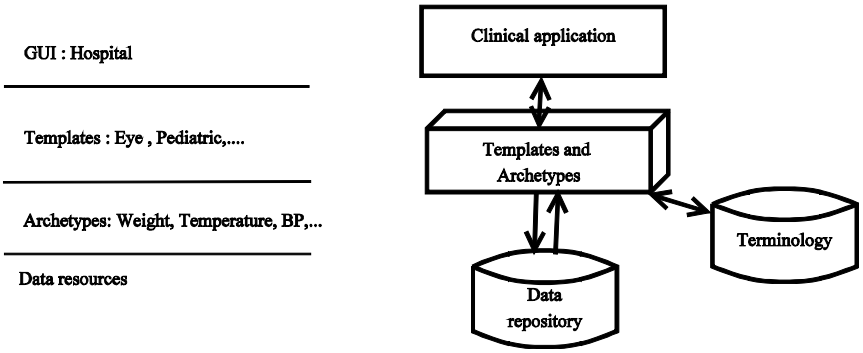


Fig. 19. EHR Data Quality Controls based on Archetypes

Thus, there are templates on top of archetypes. All the data created due to the use of templates are guaranteed to conform to the referenced archetypes. The generating archetype allocates an archetype node identifier on every node of data. This forms a

semantic imprint. As an example, the XML form of EHR data refers to a template to generate a suitable structure for data.

6.1 Templates

Templates are artifact that enables the content defined in archetypes to be used for a particular business purpose [Beale and Heard 2005]. These are created department-specific or disease-specific. These are in medical record format on a departmental basis for cardiology, eye, or liver, etc. These support bindings to terminology subsets specific to their intended use, and can be used to generate or partly generate a number of other artefact types including screen forms and message schemas as shown in Figure 20. In general, these comprise the complete application-level lumps of information to be captured or sent. They are generally developed and used locally, while archetypes are usually widely used.

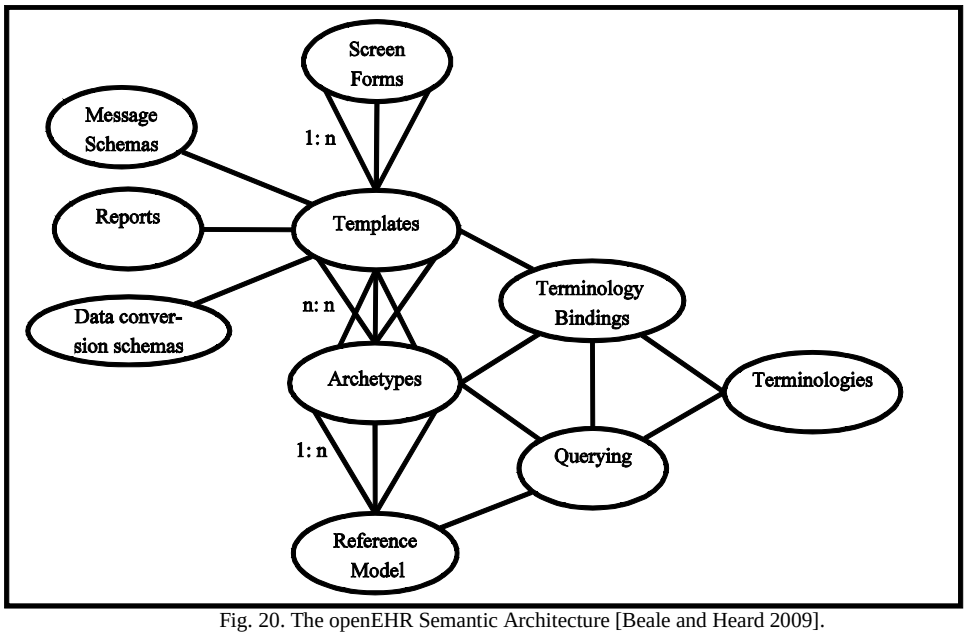


Fig. 20. The openEHR Semantic Architecture [Beale and Heard 2009].

An openEHR template is a specification that defines a tree of one or more archetypes, each constraining instances of various reference model types, such as Composition, Section, Entry subtypes and so on. Thus, while there are likely to be archetypes for such things as “biochemistry results” (an Observation archetype) and “SOAP headings” (a Section archetype), templates are used to put archetypes together to form whole Compositions in the EHR, e.g. for “discharge summary” and “antenatal exam”.

The following specifications are related to templates [Beale and Heard 2009].

(i)Template Definition Language (TDL) - an abstract language for expressing template definitions in a syntactic fashion;

(ii)The Template Object Model (TOM) - an object model that expresses the same semantics as TDL in a structural fashion;

(iii)The Operational Template Model (OTM) - an object model describing the standalone, operational template which is generated from template definitions and referenced archetypes and terminologies.

6.2 Three-Layered Building Block Structure

Figure 21 illustrates the three-layered building block structure of components in Figure

20. The first layer consists of archetypes. These are medical parameters (concepts) which define content on the basis of topic or theme (such as height, weight, BP), independent of particular business event. These can be created using the Archetype editor tool (which is a free open-source tool provided by Ocean Informatics) [Archetype Editor 2009].

The second layer consists of templates. It refers to one or more archetypes and usually imposes further constraints. These provide a way of using a particular set of archetypes, choosing a particular set of nodes from each and then limiting values and/or terminology in a way specific to a particular kind of event, such as 'diabetic patient admission' and 'discharge'. The template is often a direct precursor to a form in the presentation layer of the application software. These are the principal means of using archetypes in runtime systems. Thus the functions of templates are archetype slot filling, tightening constraints, providing default values, and meta-data (including node-level annotations).

The third layer consists of the patient record. The EHR of a specific patient can be created by assembling appropriate template(s). The record of patient 1 refers to data from multiple departments. The record of patient 2 refers to data from a single department. A new page of EHR will be created for a new date as shown in Figure 21.

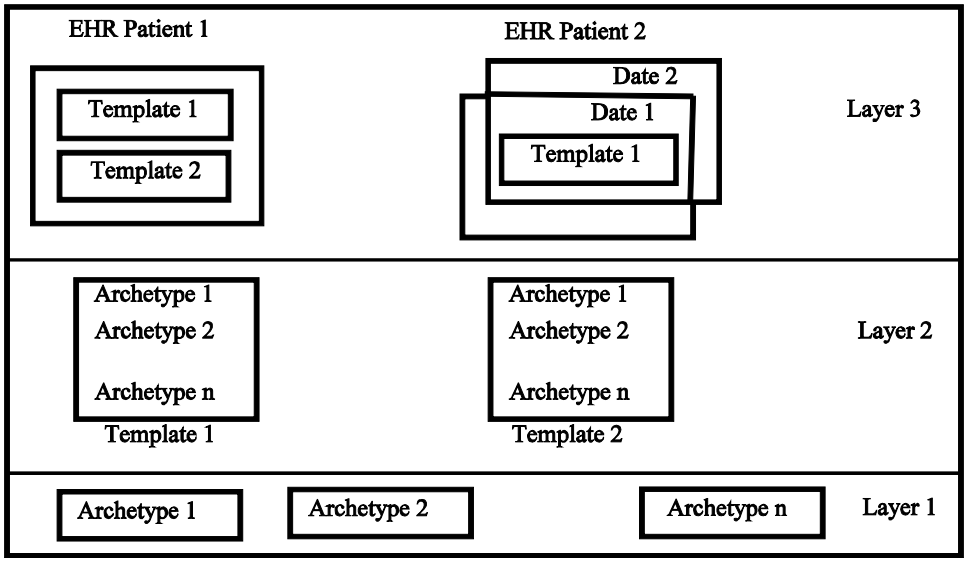


Fig. 21. Three-Layered Building Block Structure [MOSS8 2009]

7. SERVICE MODEL

The service model consists of service definitions for the major services in the EHR computing environment. These interfaces help application developers to safely assume a 'standard' API, regardless of which implementation they use. The openEHR provides different implementation technology, i.e., java / .Net / other [openEHR Java Project and openEHR .Net Project]. The service model helps back-end system implementers know what interfaces they need to expose in order to enable middleware and application developers. Also, through these interfaces, healthcare enterprises engaged in system procurement can rely on a standardized middleware 'bus' definition. This ensures that the environment that is built is always open when purchasing services.

7.1 Screen Forms

For the graphical interface of application, screen forms play an important role. These are provided by archetypes through templates.

7.2 Querying of EHR Data using AQL

Archetype paths form the basis of reusable semantic queries on archetyped data. Archetype paths can be used to construct queries that specify data items at a domain level. Hence, these are not limited to the directly connected classes and attributes of the reference model. This is in contrast to a query in standard database theory. For example, paths from a "blood pressure measurement" archetype may identify the systolic blood pressure (baseline), systolic pressures for other time offsets, the patient position, and numerous other data items [Beale and Heard 2005].

Hence, all the above components of data elements are captured by using archetypes. Thus, openEHR data elements are guaranteed to conform to the "semantic paths". These are created by the composition of archetypes within a template. The paths are incorporated within a familiar SQL-style syntax, to form queries that can be evaluated to retrieve items on a semantic basis. Queries are expressed in a language which is a synthesis of SQL (SELECT/FROM/WHERE) and W3C XPaths, extracted from the archetypes. The language is named as Archetype Query Language (AQL).

At the present stage, the technical and design aspects of openEHR have largely been outlined. EHRs should be designed with clinicians in mind. In the next phase, clinicians will be involved in archetype development. The function of archetype is to act as a design basis for queries. However, this concerns complex access plans. Querying has been identified as a major area of review by NEHTA because of lack of clear standards of EHR query services [NEHTA 2006].

7.3 Research Challenges in Querying

EHRs allow multiple representations [Chunlan et al. 2007]. In principle, EHRs can be represented as relational structures (governed by an object/relational mapping layer), and in various XML storage representations. There are many properties and classes in the reference model, but the archetypes will constrain only those parts of a model which are meaningful to constrain. These constraints cannot be stronger than those in the reference model. For example, if an attribute is mandatory in RM, it is not valid to express a constraint allowing the attribute to be optional in the archetype (ADL). So, the ADL file (alone) is not sufficient for querying. The user may want to query some properties or attributes from RM, along with the querying from properties in archetypes. 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.

At the time of query, a user or an application faces the problem while querying archetype systems. For example, the different categories have different structures. To create a data instance for blood pressure, we need two different archetypes, namely Encounter and Blood Pressure. These archetypes belong to different categories. Thus, the following archetypes must be included in querying - Encounter archetype (belonging to COMPOSITION category of RM) and Blood Pressure archetype (belonging to OBSERVATION category of RM). This problem can be addressed by the use of templates. Archetypes are encapsulated by templates for the purpose of intelligent querying [Beale and Heard 2009]. The templates are used for archetype composition or chaining. Archetypes provide the pattern for data rather than an exact template. As a result, the structure of data in any top-level object conforms to the constraints defined in a composition of archetypes chosen by a template.

Querying the system with the dual-model architecture is not the same as querying a relational database system or an XML database system. At the user level, querying data regarding 'Blood Pressure' (BP) must be made very simple. The user only knows BP as a parameter and will query that parameter only. There is a need for a query support that is neutral to system implementation, application environment and programming language. The domain professionals and software developers, both, should be able to use the query

language. For example, a patient may need to query details of medicines actually taken by him.

7.4 Archetype Query Language

A query language, AQL is being supported to query data described by archetypes in AM and RM. This is the query support that is provided by openEHR and is going to be proposed as a standardized language for querying dual-model based (standardized) EHRs. AQL [AQL 2009] is a declarative language, neutral to EHR systems, programming languages and system environments. It depends on an openEHR archetype model and semantics. It was developed on the basis of many observations, namely, a set of clinical query scenarios, study of the current available query language syntaxes (including XQuery, SQL and Object Query Language), and study of the archetypes technology, openEHR RM, and openEHR path mechanisms. It was first named as EQL (EHR Query Language) [Chunlan et al. 2007]. It has evolved and subsequently has the following two innovations:

i) utilizing the openEHR path mechanism to represent the query criteria and returned results (Figure 22); and

ii) using a 'containment' mechanism to indicate the data hierarchy and constrain the source data to which the query is applied (Figure 22).

OpenEHR path mechanism enables any node within a top-level structure to be specified from the top of the structure using a "semantic" (i.e. Archetype-based) X-path compatible path. The use of a common RM, archetypes, and a companion query language, such as AQL, facilitates semantic interoperability of EHR information. The syntax of AQL is illustrated by the help of an example. The syntax makes use of the path expression, naming retrieved results, the class expression and archetype predicate, as shown in the example in Figure 22.

Query: Find all blood pressure values, where systolic value is greater than or equal to 140, and diastolic value is greater than or equal to 90, within a specified EHR.

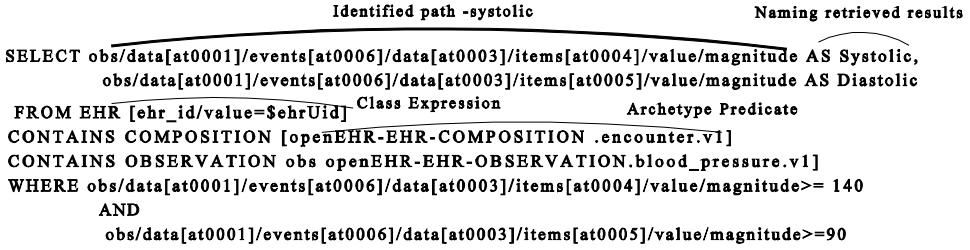


Fig. 22. Syntax of AQL [Chunlan et al. 2007]

7.5 Archetype Query Language versus other Query Languages

With existing query languages (such as XQuery, SQL, OQL), users must know the persistent data structure of an EHR in order to write an appropriate query for querying EHR data. Thus, none of these can be directly used as query language required by integrated care EHRs. It is possible to convert specification (in ADL), and patient data into its equivalent form, presented through XML. There is a large variety of software querying on XML. Table III shows the comparison between AQL and XQuery through sample queries.

Some of the features of AQL are still under development whereas XQuery is standard language incorporating all the important features of a database query language. The structure of AQL query results are still not standardized because the results representation has to be neutral to system environment and structure should be flexible (results may be structured using relational tables or represented using hierarchical structure). However,

AQL Query builder uses a generic Resultset, which has structure similar to a table [AQB 2009].

Table III. Comparison of AQL versus XQuery

S. No	Features	AQL	XQuery
1	Expression syntax	Neutral	Dependent on system implementation and environment.
2	Path expression	Yes	Yes
3	Existential Quantification	Yes	Yes
4	Projection	Yes	Yes
5	Selection Predicates	Yes	Yes
6	Relational operators/ Boolean Operators	Yes	Limited to simple cases
7	Renaming	Yes	Yes
8	Parameterization Support	Yes	No
9	Construction of new elements	No	Yes
10	Negation	Yes	Yes
11	Nesting	Yes	Yes
12	Portable	Yes	No
13	Value range as Leaf data	Yes	No
14	TOP operator	Yes	No
15	Querying Multiple	Allows multiple archetypes	Allows multiple documents
16	Universal quantification	Not Yet	Yes
17	Cartesian Product	Not Yet	Yes
18	Arithmetic functions	Yes (still to be finalized)	Yes
19	Timewindow clause	Yes	No

All products provided by Ocean Informatics (including Query Builder) are based on release 1.0.1 of the openEHR specifications. They are designed for deployment within traditional and service-oriented architectures, and support major published and emerging standards, including CEN EN13606, HL7 CDA, and HL7/OMG HSSP [Ocean Informatics].

8. DISCUSSION

Traditionally, clinicians and system users were not considered as users in the development and design phase of the systems. Also, few people are trained to work at the intersection of biomedicine and IT. Ultimately, implementing and enforcing the standards can help in improving quality [Øvretveit 2003]. At the present stage, it is important to develop a query capability which allows healthcare professionals to examine the data from a variety of perspectives.

Similarly, patients generally are not highly advanced in computer skills. They cannot access their EHR and Patient Health Information (PHI) without the help of an easy query

interface. Thus, there is a need to bridge the gap between these consumers and EHR systems.

Database query languages that assist database programmers have been around for over a decade. The languages are very good and versatile (the specifications have evolved very well over the years). But these are too demanding for the hospital-based users. AQL is a language which is at the developer's level. SQL and XQuery are at the application level. Thus, AQL is even one level lower than the SQL, XQuery. SQL is very suitable for querying relational databases. XQuery is well-suited for semi-structured data. Object-Oriented query languages are meant for object-oriented databases, but they are complex. None of the above can support medical personnel. For skilled users, query builders and 'input form and search' techniques [Jayapandian and Jagdish 2009] are available for querying systems. At the system developer's level, ADL requires highly skilled programmers for developmental stages.

One of the possible approaches for querying EHRs is to use ADL to generate a storable XML output of corresponding XML database and then to use XQuery. Also, we can use XQBE on the top of the generated XML file [Sachdeva and Bhalla 2009]. The corresponding XQuery and XQBE for the example in Figure 22 are given in Appendix B and Appendix C. XQuery uses extensible Mark-up Language (XML) as its underlying data model. It is limited to purely XML data environments. Direct use of XQuery for archetype-based EHR would require that all data be generated in XML format [Sachdeva and Bhalla 2009; Sachdeva and Bhalla 2010]. The approach suffers from many difficulties.

Firstly, openEHR is designed as an object-oriented framework. It allows for a multitude of data representations. These include programming language persistent objects (e.g., in the form of Java objects in a product such as db4o2); as language neutral objects (as in a database like Matisse3); as relational structures (governed by an object/relational mapping layer), and in various XML storage representations (e.g., XML blob or XML databases) [Chunlan et al. 2007]. These systems may lose form or content detail with changes in data representations. Usage of XQuery is therefore problematic, because the query syntax is directly tied to the representational format of the data. Considerable efforts would be required to convert openEHR data in each deployment context to XML just for the purpose of querying; such transformation may well be custom made in each case [Chunlan et al. 2007; Sachdeva and Bhalla 2009] (Example of ADL/ XML use in openEHR).

9. HIGH-LEVEL QUERY INTERFACES

At the end-user level in a hospital environment, the openEHR proposal supports forms through templates. There is a strong need for studying the available query languages and high-level query language interfaces for a match with healthcare professionals' needs. The existing query languages such as XQuery and SQL are not suitable for users. Many new approaches are being studied in different domains to provide high-level query interfaces. Several high-level query languages such as QBE, QBO, XQBE and XML-GL exist [Zloof 1975; Rahman et al. 2006; Braga et al. 2005; Ceri et al. 1999].

At the system level, the AQL language is supported for development of initial support infrastructure. The work of Sachdeva and Bhalla [2010] describes how the application level can benefit from XML conversions and support of query language at application level, in the form of XQuery. Higher-level support is an active area of research. Many research efforts aim to improve user interaction facilities [Jayapandian and Jagdish 2009; Braga et al. 2005]. This will improve the quality of care.

One possible proposal is to provide an interface at the user's level. Query-by-Object (QBO) is a high-level query interface which is user-friendly and simple to query [Bhalla and Hasegawa 2006; Rahman et al. 2006]. It follows a step-by-step procedure which

helps in removing ambiguities in user's intentions. It is based on the Information Requirement Elicitation (IRE) approach [Sun 2003]. IRE is an interactive communication activity in which an information system helps users specify their requirements with adaptive choice prompts. This is a calculator-oriented approach using an object-by-object query. The users need not possess programming skills prior to accessing the web-based information system. A similar approach, such as QBE [Zloof 1975] or QBO, may be used by the end users (such as clinicians, decision makers and patients) to simplify the process of querying EHR. An aim of the new approaches is that the user is not required to know details of persistent data. In the near future, it is recommended to support such a high-level query interface that will help to query and collect the required knowledge.

10. DATA QUALITY CONSIDERATIONS

10.1 Data Quality in EHRs

Wang and Strong [1996] describe various categories and dimensions of Information Quality as shown in Figure 23.

IQ Categories	**IQ dimensions**
Intrinsic IQ	Accuracy, Objectivity, Believability, Reputation
Accessibility IQ	Accessibility, Security
Contextual IQ	Relevancy, Value-Added, Timeliness, Completeness, Amount of Info
Representational IQ	Interpretability, Ease of Understanding, Concise Representation, Consistent Representation

Fig. 23. IQ Categories and Dimensions [Wang and Strong 1996]

Some of the data quality requirements are as follows [Orfanidis et al. 2004]:

- Accuracy and Validity. This can be achieved by structured data entry, which forces users to enter all the necessary data and thus helps to maintain structured data storage with effective data retrieval. Also, transfer of data between systems is made easier if the systems have the same data structures. The status of the EHR data should be described by metadata, for example, to indicate if data are pending, and times of entry and retention. Patients should be allowed to check the validity of their EHR data.
- Believability. The EHR should be credible and trustworthy.
- Reliability. The data should yield the same results on repeated collection, processing, storing and display of information.
- Accessibility. Current EHR data should be available to authorized users, i.e. patients, including those with special needs, care providers, mobile users, emergency services, and members of integrated care teams. Access should be fast, via easy-to-use interfaces for both care professionals and patients. Accessibility and availability of EHR data are dependent on the availability to users of the means to access EHR data, and the means to retrieve data across different, often heterogeneous, EHR systems. Giving patients access to their records is a key part of EHR strategies.
- Security. EHRs must be secure and confidential. Patients should be allowed to check who has access to their data and in what circumstances. Privacy from unauthorized users should be strictly maintained, and overriding of authorization constraints should be recorded with documented reasons.
- Timeliness. The content of an EHR should be as near real-time as possible. Thus, data should be timely, in that it relates to the present.
- Completeness. EHRs should contain all pertinent information which is complete and appropriate.

- Interpretability. EHRs should be comprehensible or understandable by all.
- Ease of Understanding. EHRs should be accessible in different data formats and from different kinds of hardware and networks, to ensure interoperability between different systems. Data held within an EHR should be organized (including chronologically) and presented for ease of retrieval.
- Consistency. There should be consistency between items of multiple data from multiple sources. EHRs should comply with the existing relevant standards, such as security, data protection, and communication standards (HL7, CEN13606). Data accreditation standards should be established for new data, and inconsistency and duplication should be removed.

Table IV gives a narrative view of how the DQ aspect is enhanced in standardized EHRs.

Table IV. DQ aspect in standardized EHRs

DQ Dimensions	**DQ Enhancements in EHRs**
Accuracy and Validity	• Business rules defined in archetypes • 'null_flavor' in ELEMENT class of RM
Believability	• RM based on clinical investigator recording process; • Archetypes developed by domain experts
Reliability	• Two-level modeling approach (stable RM) • Richer data structures and data types • Data Independence among RM,AM and SM
Accessibility	• Sharable archetypes
Security	• Security Information Model of RM
Timeliness	• Version Control in RM • New and modified archetypes are developed as clinical knowledge expands
Completeness	• Standardized data definitions, content and structure
Interpretability	• Fine granularity of data in archetypes • Linkage to terminology standards
Ease of Understanding	• User-level query ability and usability
Concise Representation	• Rich health data definition (archetypes)
Consistency	• Interfaces for legacy systems (non-standardized) • Interfaces to other systems (HL7, CEN 13606) • Ontology-based archetype transformation process (e.g. openEHR archetypes to HL7 CDA archetypes or CEN 13606 archetypes)

Example of Accuracy: In the openEHR RM, the class 'ELEMENT' has attribute 'null_flavor'. It is used to mark a 'lack of data'. Using this attribute was inspired by a) the need to do something about marking missing data in health information and b) the use of 'data quality markers' in SCADA control systems which show on the screen when a measured value from the field is out of date or wrong due to technical failure to obtain the current value. In the development of openEHR, a data quality marker has been made available for a similar reason: to indicate technical incapacity to obtain data. Thus, the problems of incompatible basic data types and overlapping and incompatible definitions of clinical content have been addressed and solved by openEHR

10.2 Data Users Impacting DQ in EHR systems

The DQ is analyzed using three data users, viz., collectors, custodians and consumers [Wang et al. 2001]. In healthcare, the data producers are medical or administrative staff. The data custodians are the DBA or computer scientists, and the data consumers are

physicians, researchers and managers. The various users and their associated tasks are shown in Figure 24. Quality in data collection is achieved by adherence to guidelines and data definitions. Sufficient data checks at the point of data entry are enforced by the use of interfaces generated on the top of archetypes (i.e. templates). The medical staff requires training to avoid typing errors, transcription errors and incomplete transcription. The protocol of data collection is also significant regarding the DQ aspect. Special attention is given in openEHR standard as it contains a protocol section in all the ENTRY classes of RM. The data custodians can easily maintain the data quality aspect of the system because in two-level modeling, the software development is separated by domain knowledge. Healthcare professionals require aggregated and integrated patient information that may be distributed across multiple sites.

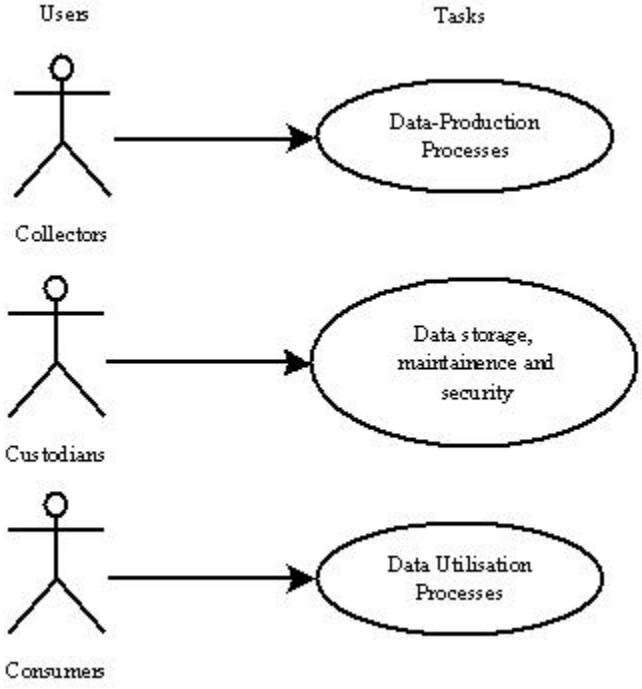
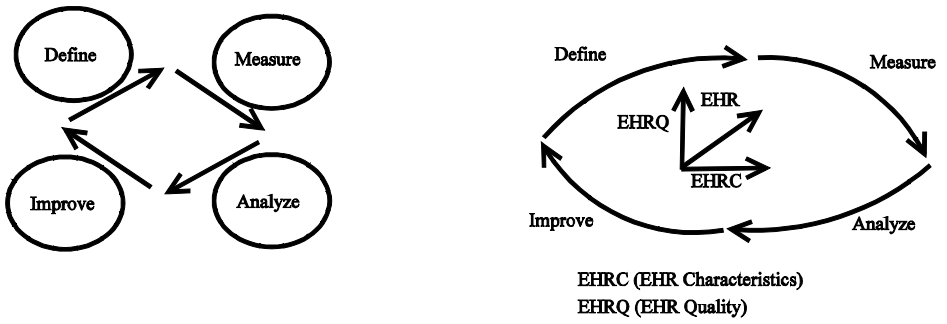


Fig. 24. Data Users impacting Data Quality

10.3 Data Quality Improvement Methodology for EHRs

A number of methodologies (AIMQ [Lee et al. 2002], TDQM [Wang 1998], and CDQM [Batini and Scannapieco 2006]) have been developed to help information quality practitioners to discover which forms of information quality are of relevance to their stakeholders, and to help convert them into specific quality scores.

A data quality framework is a tool for the assessment of data quality within an organization [Wang et al., 1996]. In a Total Data Quality Management (TDQM) framework [11], the 'Define' component identifies the important DQ dimensions and the corresponding DQ requirements. The 'Measure' component produces the DQ metrics. The 'Analyze' component identifies the root causes for DQ problems and calculates the impact of poor quality information. The 'Improve' component provides techniques for improving DQ. Analogous to the TDQM cycle, a data quality improvement methodology for EHRs in an EHR system has been proposed as shown in Figure 25. In applying the framework, one must define the characteristics of EHR, assess the EHR data quality requirements, and identify the EHR system for the EHR. In the figure, EHRC stands for EHR characteristics and EHRQ stands for EHR quality. Figure 25 also depicts the DQ components defined below.



(a) A Schematic of the TDQM Cycle [Wang 1998]

(b) Data Quality Improvement Methodology for EHRs in EHR System over a period of time

Fig. 25. Similarity between TDQM Cycle and Data Quality Improvements for EHRs

(i) **Define:** The EHRQ definition component defines DQ requirements of EHRs. It describes how the EHR must be produced, along with the interaction or communication among EHR suppliers (vendors), manufacturers, consumers and managers. This is achieved by means of standardization of EHR. Rich health data definitions (archetypes) are defined and agreed upon by the clinicians themselves to ensure that each piece of health information is unambiguously understood, and can be dynamically used and reused to support wise and safe health choices.

Many of the DQ problems associated with the legacy (non-standardized) systems can be corrected with this standardized EHR system. Thus, this forms the basis for efforts now undergoing to bring data from legacy systems to openEHR-compliant data.

(ii)**Measure:** The key to measurement resides in the development of data quality metrics. The DQ metrics can be basic DQ measures such as data accuracy, timeliness, completeness, and consistency [Wang et al. 2001]. In the EHR, the DQ metrics are designed to track, for instance:

- The percentage of incorrect patient address zip codes found in randomly selected patient accounts (inaccuracy)
- A mechanism for checking the last updating of EHR (timeliness)
- Which contents of the EHR to be marked as mandatory / optional (completeness)
- The number of records that violate referential integrity (consistency), with reference to the same piece of data in another part of storage.

DQ metrics also measures the EHR's link to specific terminology standards (term-binding section of archetypes). The security concerns of patient are equally important. First, who can access the full EHR? Secondly, the researchers may require only the medical information of the patient. Also, the interoperability of the EHR needs to be examined. Is the EHR interoperable at the syntactic, structural or semantic level? With the DQ metrics, DQ measures can be obtained along various DQ dimensions for analysis.

Mandl and Porter [1999] identify two important quality indicators, i.e. completeness of medical record data and good communication between physicians and patients. However, procedures and metrics for measuring these and other indicators have yet to be established. However, the Canadian Institute for Health Information (CIHI) has developed an integrated framework in order to measure the data quality of health data [Long et al. 2002].

Measuring data quality is a complex process and current solutions cover only limited aspects of data quality. Data quality metrics and design methods which address this problem throughout the system lifecycle are required, and perhaps best integrated within object-oriented analysis and design notations, such as UML, so that data quality can be

addressed from both structural and behavioral perspectives.

(iii) **Analyze:** From the measurement results, the root cause for potential DQ problems can be investigated. The ISO and other standard bodies examine the EHR architecture, evaluate how representative or comprehensive the DQ metrics are, and whether these metrics are the right set of metrics. This is an ongoing process.

(iv) **Improve:** Based on the analysis phase, EHR improvement is carried out. Kerr et al., found that for DQ improvement strategy, it is important to derive and impose standards that facilitate data and information transfer whilst preserving quality [Kerr et al. 2007]. The EHR team needs to identify key areas for improvement such as,
(1) Making the EHRs semantically more interoperable, and
(2) Exchange of EHRs among different standards-compliant EHR systems.

The new and modified archetypes are developed as the clinical knowledge is enhanced. Quality improvement is an iterative cycle [Kerr et al. 2008]. The requirements may continue to change over time.
A recent study highlights how the information quality can be improved [Donoghue et al. 2009]. It integrates remote patient monitoring solutions i.e. a Body Area Network (BAN) datasets within patient EHR.

The openEHR standard-based data aims to be accurate, complete, relevant, timely, sufficiently detailed, approximately represented (e.g., consistently coded using a coding system), and retain sufficient contextual information to support decision making. Standardized data definitions, content, structure, and the establishment of quality checkpoints throughout the data capture process enhance the interoperability of healthcare systems. Based on that, certain ways of expressing value constraints for validation will enhance DQ.

11. SUMMARY AND CONCLUSIONS

Healthcare activity needs to be automated to bring uniformity and to improve quality [Øvretveit et al. 2007]. EHRs are life-long health records that can transform medical practice, making it more efficient and saving money and time [Wang et al. 2003]. Enhancements in EHR architecture have become available. These focus on two main issues, standardization and interoperability. Standardization is being accomplished through a dual-level modeling approach. In this approach, the software development can proceed separately from domain modeling, and if new concept models are introduced or altered, the software need not have to be redesigned, coded, tested and redeployed. The dual-level model thus enhances the quality of information systems. Knowledge level interoperability will be achieved through the establishment of archetypes. Archetypes are developed through domain knowledge governance, which resolves the human problem involving agreement on what is contained within the domain and why is it important.

Tayi and Ballou [1998] consider four dimensions of data quality: accuracy, completeness, consistency and timeliness. Archetypes are accurate insofar as they provide correctness and precision with which the real-world data are represented. They are complete insofar as fine granularity of data is provided (i.e. all relevant data are recorded). They are consistent insofar as they are the basis for data that satisfy specified constraints and business rules. They provide timeliness insofar as versioning is possible, so the recorded data are up-to-date.

EHR semantic interoperability ensures the necessary data quality and consistency. It will enable meaningful and reliable use of longitudinal and heterogeneous data for public health, research, and health service management. The growing size and quality of the openEHR repository of archetypes means that individual organizations using the technology have to do less work to be interoperable, while gaining access to the content

models created and used by some of the largest health organizations in the world, including the Australian government and the NHS in UK.

Pioneering new archetypes will allow the clinical concepts to be expanded. These also provide the basis for querying EHR repositories. Querying over EHR data has to be neutral to EHR systems, programming languages, and system environments. The query syntax has to be neutral with respect to the reference model, i.e., the common data model of the information being queried. These objectives are met by the new architectural design. EHRs can help in delivering the right information to the right person at the right time. Patients can have complete control over access and distribution of their health records. The openEHR uses low-cost software (because it is open source). Its maintenance due to software robustness (suitable for developing countries' economy) is easy. For these reasons, it has been chosen for study and adoption by various organizations. Microsoft, Queensland Health (Australia), Bert Verhees (Netherlands), NexJ systems of Canada, National e-health programs (going on in Singapore, Sweden, Denmark, Great Britain) are working on an archetype-based openEHR approach [Microsoft 2009] [MOSS8 2009].

The openEHR approach can, moreover, provide the common basis for ubiquitous presence of meaningful and computer-processable knowledge and information and thus contribute to the usability of clinical systems, improve data quality, and improve semantic interoperability. To utilize the full potential of interoperable EHR systems they have to be accepted by their users, the healthcare providers. Graphical user interfaces that support customization and data validation play a decisive role for user acceptance and data quality. Further research and study to provide information that focuses on improving existing tools/ algorithms are required.

REFERENCES

ALEXANDROU, D. AND MENTZAS, G. 2009. Research Challenges for achieving healthcare business process interoperability. In IEEE International conference on eHealth, telemedicine and social medicine.

AQB (Archetype Query Builder). [Online]. Available: http://www.oceaninformatics.com/Solutions/ocean-products/Clinical-Modelling/Ocean-Query-Builder.html (retrieved Dec, 2009).

AQL (Archetype Query Language). 2009. [Online] Available: http://www.openehr.org/wiki/display/spec/Archetype+Query+Language+Description (retrieved 10 Dec, 2009).

ARCHETYPE EDITOR TOOL. [Online]. Available: http://wiki.oceaninformatics.com/confluence/display/TTL/Archetype+Editor+Releases.

ATALAG, K., KINGSFORD, D., PATON, C. AND WARREN, J. 2010. Putting Health Record Interoperability Standards to Work. The electronic Journal of Health Informatics (eJHI), 5 (1).

BATINI, C. AND SCANNAPIECO, M. 2006. Data Quality: Concepts, Methodologies, and Techniques, Springer, Berlin.

BEALE, T. AND FRANKEL, H. 2007. The openEHR Reference Model: Extract Information Model. The openEHR release 1, openEHR Foundation.

BEALE, T. AND HEARD, S. 2005. Archetype Definitions and Principles. In the openEHR release 1.0.2, the openEHR foundation.

BEALE ,T. AND HEARD, S. 2008 a. The openEHR Architecture: Architecture Overview. In the openEHR release 1.0.2, openEHR Foundation.

BEALE, T., AND HEARD, S. 2008 b. The openEHR Archetype Model-Archetype Definition Language ADL 1.4. In openEHR release 1.0.2, issue date 12 Dec 2008.

BEALE, T., HEARD, S., KALRA, D. AND LLYOD, D. 2008. The openEHR Reference Model: EHR Information Model. In the openEHR release 1.0.2. , openEHR Foundation.

BEALE, T. AND HEARD S. 2009. The openEHR Archetype Model- openEHR Templates. In openEHR release 1.0.2, issue date 20 April 2009.

BEALE, T., 2008. The openEHR Archetype Model: Archetype Object Model. In the openEHR release 1.0.2, openEHR Foundation.

BEALE, T. 2010. OpenEHR to ISO 13606-1, ISO 21090 mapping. [Online]. Available: http://www.openehr.org/wiki/display/stds/openEHR+to+ISO+13606-1%2C+ISO+21090+mapping

BHALLA, S. AND HASEGAWA, M. 2006. Query Interface for Ubiquitous Access to Database Resources. In 13th International Conference on Management of Data (COMAD), New Delhi, India, Dec 14, 2006.

BISBAL, J. AND BERRY, D. 2009. Archetype Alignment - A Two-level Driven Semantic Matching Approach to Interoperability in the Clinical Domain. HEALTHINFO 2009, 216-221

BLOBEL, B.G.M.E. AND PHAROW P. 2008. Analysis and Evaluation of EHR approaches. In eHealth Beyond the Horizon- Get It There, MIE 08.

BOTT, O. J. 2004. The Electronic Health Record: Standardization and Implementation. In 2nd OpenECG Workshop, Berlin, Germany.

BRAGA, D., CAMPI, A. AND CERI, S. 2005. XQBE (XQueryBy Example): A Visual Interface to the Standard XML Query Language. ACM Transactions on Database Systems, Vol. 30, No. 2, 398–443.

CEN TC/251 (European Standardization of Health Informatics) ENV 13606 Electronic Health Record Communication. [Online]. Available: http://www.centc251.org/

CERI, S., COMAI, S., DAMIANI, E., FRATERNALI, P., PARABOSCHI, S., AND TANCA L. 1999. XML-GL: a graphical language of querying and restructuring XML documents. In Proc. WWW8, Toronto, Canada, May 1999.

CHUNLAN, M., FRANKEL, H., BEALE, T. AND HEARD S. 2007. EHR Query Language (EQL)-A Query Language for Archetype-Based Health Records. In MEDINFO.

CKM (Clinical Knowledge Manager). [Online] Available: http://www.openehr.org/knowledge/ (retrieved Sep, 2009).

DONOGHUE J. O., HERBERT J., REILLY, P.O., AND SAMMON D. 2009. Towards Improved Information Quality: The Integration of Body Area Network Data within Electronic Health Records, M. Mokhtari et al. (Eds.): ICOST 2009, LNCS 5597, 299–302.

EHR STANDARDS (Electronic Health Records Standards). [Online]. Available: http://en.wikipedia.org/wiki/Electronic_health_record#Standards (retrieved 12 Dec, 2009).

EICHELBERG, M., ADEN, T., RIESMEIER, J., DOGAC, A., LALECI, G.B. 2005. A survey and analysis of Electronic Healthcare Record standards. ACM Comput. Surv., 37(4), 277-315.

GEHR(Good Electronic Health Record Project). [Online]. Available: http://www.gehr.org/

GENDRON, M. S., AND D'ONOFRIO, M. J. 2001. Data Quality in the Healthcare Industry. Data Quality Journal, vol 7, no.1.

GÖK, M. 2008. Introducing an openEHR-Based Electronic Health Record System in a Hospital. Master thesis, University of Goettingen, May 2008.

HRISTIDIS, V. 2009. Chapter 4 on Data Quality and Integration Issues in EHRs , In book on Information Discovery on Electronic Health Records , series of Chapman & Hall/CRC Data Mining and Knowledge Discovery Series.

IHTSDO AND OPENEHR COLLABORATION. 2009, 15 Sep. [Online] Available: http://www.openehr.org/292-OE.html?branch=1&language=1

ISO 13606-1. 2008. Health informatics - Electronic health record communication - Part 1: RM, 1st edition.

ISO 13606-2. 2008. Health informatics - Electronic health record communication - Part 2: Archetype Interchange Specification, 1st edition.

ISO/TC 215 TECHNICAL REPORT. 2003. Electronic Health Record Definition, Scope, and Context. 2nd Draft, Aug 2003.

JAYAPANDIAN, M. AND JAGADISH, H.V. 2009. Automating the Design and Construction of Query Forms. IEEE Transactions on Knowledge and Data Engineering, Vol 21, Issue 10, 1389 – 1402.

KENNELLY, R.J. 1998. IEEE 1073, Standard for Medical Device Communications. In AUTOTESTCON '98. IEEE Systems Readiness Technology Conference, 335-336.

KERR, K., NORRIS, T., AND STOCKDALE R. 2007. Data Quality Information and Decision Making: A Healthcare Case Study, In 18th Australasian Conference on Information Systems, Toowoomba.

KERR, K. A., NORRIS, T., AND STOCKDALE, R. 2008. The strategic management of data quality in healthcare, Health Informatics Journal, 14: 259.

LEE, Y., STRONG, D., KAHN, B., AND WANG, R. 2002. AIMQ: A methodology for information quality assessment. Inform. Manag. 40, 2, 133–146.

LESLIE, H. AND HEARD S. 2006. Archetypes 101. In HIC 2006 and HINZ 2006: Proceedings. Brunswick East, Vic.: Health Informatics Society of Australia,18-23.

LEWIS, G. A., MORRIS, E., SIMANTA, S. AND WRAGE L. 2008. Why standards are not enough to guarantee end-to-end interoperability. In IEEE 7th International conference on Composition-based software systems.

LONG, J., RICHARDS, J., AND SEKO, C. 2002. The Canadian Institute for Health Information (CIHI) Data Quality Framework, Version 1. A Meta-Evaluation and Future Directions.

MADNICK, S.E., WANG, R.Y., LEE, Y.W., ZHU, H. 2009. Overview and Framework for Data and Information Quality Research. ACM Journal of Data and Information Quality, Vol 1, No. 1, Article 2.

MALDONADOA, .J.A., MONERA, D., TOMÁSA, D., ÁNGULOA, C., ROBLESA, M. AND FERNÁNDEZB J.T. 2007. Framework for Clinical Data Standardization Based on Archetypes. In Medinfo.

MANDL, K AND PORTER, S. 1999. Data quality and the electronic medical record: a role for direct parental data entry. In American Medical Informatics Association Three Year Cumulative Symposium Proceedings 1999, 1998, 1997.AMIA, 1999.

MICROSOFT CONNECTED HEALTH FRAMEWORK. [Online]. Available: http://www.microsoft.com/industry/healthcare/technology/HealthFramework.mspx. (retrieved Nov, 2009).

MIETTINEN, M. AND KORHONEN, M. 2008. Information Quality in Healthcare: Coherence of Data Compared between organization's Electronic Patient Records. In 21st IEEE International Symposium on Computer-Based Medical Systems.

MIKKELSEN, G. AND AASLY, J. 2005. Consequences of impaired data quality on information retrieval in electronic patient records. *Int. J. Med. Inf. 74*, 5, 387–394.

MOSS8 (8th Medical Open Source Software seminar). 2009. [Online] Available: http://www.openehr.org/293-OE.html?branch=1&language=1

NEHTA (National E-Health Transition Authority). 2006. Review of Shared Electronic Health Records Standards. In Publications, in: eds. National E-Health Transition Authority.

NHIN. 2005. NHIN:Interoperability for the National Health Information Network.IEEE, USA e-books,Nov 2005.

OCEAN INFORMATICS. [Online]. Available: http://www.oceaninformatics.com/ (retrieved Jan,2010).

OPENEHR COMMUNITY. [Online]. Available: http://www.openehr.org/ (retrieved May, 2009).

ORFANIDIS, L., BAMIDIS, P. D., AND EAGLESTONE, B. 2004. "Data Quality Issues in Electronic Health Records: An Adaptation Framework for the Greek Health System, Health Informatics Journal, vol 10(1), 23.

ØVRETVEIT, J. 2003. What are the best strategies for ensuring quality in hospitals?. In WHO Regional Office for Europe's Health Evidence Network (HEN), Nov 2003.

ØVRETVEIT, J., SCOTT, T., RUNDALL, G. T., SHORTELL, S. M. AND BROMMELS, M. 2007. Improving quality through effective implementation of information technology in healthcare. In International Journal for Quality in Health Care, Vol 19, No. 5,259–266.

PATRICK, J., LY, R. AND TRURAN, D. 2006. Evaluation of a Persistent Store for openEHR. In Proceedings of HIC 2006 and HINZ 2006, Brunswick East, Vic.: Health Informatics Society of Australia,83-89.

PISHEV, O. 2006. The openEHR advantage. White Paper. [Online] Available: http://www.oceaninformatics.com/ocean-informatics-resources/ocean-documentation/Published-Articles/The-iopeniEHR-advantage2.html.

POISSANT, L., PEREIRA, J., TAMBLYN, R. AND KAWASUMI, Y. 2005. The impact of electronic health records on time efficiency of physicians and nurses: A systematic review. In Journal of the American Medical Informatics Association, Elsevier, vol 12, 505–516.

RAHMAN, S. A., BHALLA, S., AND HASHIMOTO, T. 2006. Query-by-object Interface for Information Requirement Elicitation in M-commerce. International Journal of Human Computer Interaction (HCI), Mar 2006.

SACHDEVA S. AND BHALLA S. 2010. Semantic Interoperability in Healthcare information for EHR databases. In 6th International Workshop, Databases in Networked Information Systems (DNIS) Mar 2010, Aizu-Wakamatsu, Japan, 157-173.

SACHDEVA, S. AND BHALLA, S. 2009. Implementing High-Level Query Language Interfaces for Archetype-Based Electronic Health Records Database. In International Conference on Management of Data (COMAD) Dec 2009, 235-238.

SILBERSCHATZ, A., KORTH, H. F. AND SUDARSHAN, S. 2005. Database Systems Concepts. 5th edition, Mc Graw Hill, international edition.

SIMONOV, M., SAMMARTINO, L., ANCONA, M., PINI, S., CAZZOLA, W. AND FRASCIO M. 2005. Information, knowledge and interoperability for healthcare domain. In IEEE, AXMEDIS' 05.

SNOMED (Systematized Nomenclature of Medicine) Clinical Terms. [Online]. Available: http://www.snomed.org/documents/snomed_overview.pdf

SOKOLOWSKI, R. 1999. Expressing Health Care Objects in XML. In Proceedings of the 8th Workshop on Enabling Technologies on Infrastructure for Collaborative Enterprises.

SUN, J. 2003. Information Requirement Elicitation in M-Commerce - An Interactive Approach to Facilitate Information Search for Mobile Users. Communications of ACM, Vol. 46, No. 12, 45-47.

TAYI,G. K. AND BALLOU, D. P. 1998. Examining Data Quality. Communications of ACM, 41(2), 54-57.

THE OPENEHR .NET KNOWLEDGE TOOLS PROJECT. [Online]. Available: http://www.openehr.org/projects/dotnet.html

THE OPENEHR JAVA REFERENCE IMPLEMENTATION PROJECT. [Online]. Available: http://www.openehr.org/projects/java.html

THURSTON, L. M. 2006. 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,28-36.

WALKER, J., PAN, E., JOHNSTON, D., ADLER-MILSTEIN, J., BATES, D.W., AND MIDDLETON, B. 2005. The Value of Health Care Information Exchange and Interoperability. Health affairs, 19 Jan 2005. http://content.healthaffairs.org/cgi/content/abstract/hlthaff.w5.10 .

WANG, S. J., MIDDLETON, B., PROSSER. L. A., BARDON, C. G., SPURR, C.D., CARCHIDI, P.J., KITTLER, A. F., GOLDSZER, R.C., FAIRCHILD, D. G., SUSSMAN, A.J., KUPERMAN, G.. J. AND BATES, D. W. 2003. A cost-benefit analysis of electronic medical records in primary care. In Am J Med 2003, 114 (5), 397-403.

WANG, R. Y. 1998. A product perspective on total data quality management. Comm. ACM 41, 2, 58–65.

WANG, R. Y. AND STRONG, D. M. 1996. Beyond accuracy: what data quality means to data consumers. Journal of Management Information Systems, vol 12, no. 4, 5-33.

WANG, R. Y., ZAID, M. AND LEE, Y. W. 2001. Book on "Data Quality". Kluwer Academic publishesrs.

ZLOOF, M. M. 1975. Query-By-Example. In Proceedings of AFIPS '75, national computer conference and exposition.

Appendix A

i) dADL [Beale and Heard 2008 b]

An ADL archetype is a dADL document with the structure, as shown in Figure 17. In this dADL document, all top-level attribute names are an exact match in name and type for corresponding attributes in the "openEHR" Archetype Object Model (AOM) [Beale 2008]. A dADL document may contain one or more objects from the same object model. Comments are indiacted by characters "–".Two types of identifiers – viz, type names and attribute names from information models, are used in dADL. Type names are in all uppercase, e.g., PERSON, except for 'built-in' types, such as primitive types (Integer, String, Boolean, Real, Double). The assumed container types (List<T>, Set<T>, Hash<T, U>), are in mixed case, in order to provide easy differentiation of built-in types from constructed types defined in the reference model. Attribute names are all in lowercase. Semi-colons can be used to separate dADL blocks, for example when it is preferable to include multiple attribute/value pairs on one line.

Intervals of any ordered primitive type, i.e., Integer, Real, Date, Time, Date_time and Duration, can be stated using the following uniform syntax, where N, M are instances of any one of the ordered types:

|N..M| the two-sided range N >= x <= M;
|N..<M| the two-sided range N >= x <M;
|N>..<M| the two-sided range N > x <M;
|<N| the one-sided range x < N;
|>N| the one-sided range x > N;
|>=N| the one-sided range x >= N;
|<=N| the one-sided range x <= N;
|N +/-M| interval of N ± M.

URI can be expressed as dADL data in the usual way found on the web. Examples of URIs in dADL: http://archetypes.are.us/home.html; ftp://get.this.file.com#section_5; http://www.mozilla.org/products/firefox/upgrade/?application=thunderbird.

XML does not have systematic object-oriented semantics. The dADL syntax maps quite easily to XML instance. As a result of mapping, the Xpath expressions are the same for dADL and XML. These correspond to - what one would expect, based on an underlying object model.

Example of similarity with XML (a): Container attribute nodes in dADL map to a series of tagged nodes of the same name, each with the XML attribute 'id' set to the dADL key. For example [Beale and Heard 2008 b], see Figure 26. This guarantees that the path subjects[@id="philosophy:plato"]/name navigates to the same element in both dADL and the XML.

dADL	XML
subjects = < ["philosophy:plato"] = < name = <"philosophy"> > ["philosophy:kant"] = < name = <"philosophy"> > >	<subjects id="philosophy:plato"> <name> philosophy </name> </subjects> <subjects id="philosophy:kant"> <name> philosophy </name> </subjects>

Fig. 26. Fragment of container attribute nodes in dADL mapped to XML

Example of similarity with XML (b): Nested container attribute nodes in dADL map to a series of tagged nodes of the same name, each with the XML attribute 'id' set to the

dADL key. For example, consider an object structure defined by the signature countries: Hash<Hash<Hotel,String>,String>. An instance of this in dADL can be mapped to the XML (Figure 27) in which the synthesised element tag "_items" and the attribute "key" are used. In this example, the dADL path countries["spain"]/["hotels"] will be transformed to the Xpath countries[@key="spain"]/_items[@key="hotels"] in order to navigate to the same element.

dADL	XML
countries = < ["Spain"] = < ["hotels"] = <...> ["attractions"] = <...> > ["Egypt"] = < ["hotels"] = <...> ["attractions"] = <...> > >	<countries key="Spain"> <_items key="hotels"> ... </_items> <_items key="attractions"> ... </_items> </countries> <countries key="Egypt"> <_items id="hotels"> ... </_items> <_items key="attractions"> ... </_items> </countries>

Fig. 27. Fragment of nested container attribute nodes in dADL mapped to XML

dADL data is hierarchical, every node is identified by unique path. Paths are directly convertible to XPath expressions for use in XML-encoded data. It allows for attributes of container types such as lists, sets and hash tables. dADL opts for the array-style syntax, known in dADL as container member keys. The use of string values as keys for the contained items is also allowed. Container structures can appear anywhere in an overall instance structure, allowing complex data to be expressed in a readable way. Nested container objects such as List<List<String>> are also allowed in dADL syntax. Typing information is added to instance data using a syntactical addition inspired by the (type) casting operator of the C language. Type identifiers can also include namespace information, which is necessary when same-named types appear in different packages of a model. All dADL data eventually devolve to instances of the primitive types String, Integer, Real, Double, String, Character, various date/time types, lists or intervals of these types, and a few special types.

Example of similarity with XML (c): Type names in dADL map to XML 'type' attributes (Figure 28).

ii) Constraint ADL (cADL)

cADL is a syntax which enables constraints on data defined by object-oriented information models. It is used in archetypes or other knowledge definition formalisms. cADL is used both at "design time", by authors and tools, and at runtime, by computational systems. These validate data by comparing it to the appropriate sections of cADL in an archetype.

For example, in a demographic information model (part of reference model in Figure 9) which has only the types PARTY and PERSON, one can write cADL. It can define the concepts of entities such as PATIENT, DOCTOR and NURSE, in terms of constraints on the types available in the information model.

dADL	XML
destinations = < ["seville"] = (**TOURIST_DESTINATION**) < profile = (**DESTINATION_PROFILE**) <> hotels = < ["gran sevilla"] = (**HISTORIC_HOTEL**) <> > > >	<destinations id="seville" adl:type=**"TOURIST_DESTINATION">** <profile adl:type="**DESTINATION_PROFILE"**> ... </profile> <hotels id="gran sevilla" adl:type="**HISTORIC_HOTEL"**> ... </hotels> >

Fig. 28. Fragment of type name in dADL mapped to XML

iii) Assertion Language

Assertions are used in archetype "slot" clauses in the cADL definition section, and in the invariant section. The archetype assertion language is a small language of its own. Formally it is a first-order predicate logic with equality and comparison operators (=, >, etc). It is very nearly a subset of the OMG's emerging OCL (Object Constraint Language) syntax. It has reserved keywords (Table V). A constraint can be defined which allows other archetypes to be used. This is known as an archetype "slot", or "chaining point", written in the ADL assertion language. Two kinds of reference may be used in a slot assertion. The first is a reference to an object-oriented property of the filler archetype itself, where the property names are defined by the ARCHETYPE class in the Archetype Object Model. The second kind of reference is to absolute paths in the definition section of the filler archetype. Both kinds of reference take the form of an Xpath style path, with the distinction that paths referring to ARCHETYPE attributes not in the definition section do not start with a slash.

Table V. Keywords in Assertion Language [Beale and Heard 2008 b].

Textual Rendering	**Symbolic Rendering**	**Meaning**
matches, is_in	∈	Set membership, "p is in P"
exists	∃	Existential quantifier, "there exists…"
for_all	∀	Universal quantifier, "for all x…"
implies	⊃, →	Material implication, "p implies q", or "if p then q"
and	∧	Logical conjunction, "p and q"
or	∨	Logical disjunction, "p or q"
xor	⩒	Exclusive or, "only one of p or q"
Not, ~	~, ¬	Negation, "not p"

Three types of structure representing constraints on complex objects have been presented so far. These are,

• **complex object structures:** any node introduced by a type name and followed by {} containing constraints on attributes;

• **internal references:** any node introduced by the keyword use_node, followed by a type name. Such nodes indicate re-use of a complex object constraint that has already been expressed elsewhere in the archetype;

• **archetype slots:** any node introduced by the keyword allow_archetype, followed by a type name. Such nodes indicate a complex object constraint which is expressed in some

other archetype.
Operators can be arithmetic, equality, Boolean, relational and quantifiers. Operands in an assertion expression can be any of the following:

- manifest constant: any constant of any primitive type, expressed according to the dADL syntax for values;
- variable reference: any name starting with '$', e.g. $body_weight;
- object reference: a path referring to an object node, i.e., any path ending in a node identifier; and
- property reference: a path referring to a property, i.e., any path ending in ".property_name"

If an assertion is used in an archetype slot definition, its paths refer to the archetype filling the slot, not the one containing the slot. There are some predefined variables that can be referenced in ADL assertion expressions such as $ current_date, $current_time. Variables can also be defined inside an archetype, as part of the assertion statements in an invariant. The assertion grammar is a part of cADL grammar. The assertions do not constrain data in the way that other archetype statements do, instead they constrain archetypes.

Appendix B. XQuery for the 'BP' query example.

```
for     $data in doc("ehr.xml")/version/data,
        $content in $data/content,
        $data_2 in $content//data
where(  some $items in $data_2/items satisfies
        some $items_2 in $data_2/items satisfies
        $data/@archetype_id = "Composition.enconter.v1" and
        $content/@archetype_id = "Observation.BP.v1" and
        $items/@archetype_node_id = "at0004 or Diastolic" and
        $items/value/magnitude/text() >90 and
        $items_2/@archetype_node_id = "at0005 or systolic" and
        $items_2/value/magnitude/text() >140 and
        $data_2/@archetype_node_id = "at0003 or BP" )
return  <BP> {
                $data_2/@*,
                $data_2/text(),
                $data_2/*
        } </BP>
```

Appendix C. XQBE interface for the 'BP' query example [Sachdeva and Bhalla 2009].