The *open*EHR Archetype System

Editors:{T Beale, S Heard}[1]

Revision: 0.3.1

Pages: 23

Keywords: EHR, health records, modelling, constraints, internet

1. Ocean Informatics Australia

Copyright Notice

Amendment Record

Issue	Details	Who	Completed
R E L E A S E 0.95			
R E L E A S E 0.9			
0.3.1	Updated diagram; added notes on dissemination network.	S Heard	20 Dec 2003
0.3	Initial Writing	T Beale	29 Nov 2003

Table of Contents

1 Introduction

1.1 Purpose

This document provides a description of the *open*EHR Archetype System, a network of archetype authoring, quality assurance and propagation to archetype-enabled information systems.

1.2 Status

This document is under development, and is published as a proposal for input to standards processes and implementation works.

The latest version of this document can be found in PDF format at http://svn.openehr.org/specification/TRUNK/publishing/architecture/am/archetype_system.pdf. New versions are announced on openehr-announce@openehr.org.

1.3 Overview

The Archetype "System" is a proposed global system consisting of two parts: the first is collaborative archetype development, review and publishing; and the second is online propagation of published, approved archetypes to runtime systems. In the first "network", the key features are collaborative development, testing and review by domain experts, enabled by a managed archetype document lifecycle; this network functions like an open source network of developers using version control repositories to share and maintain their deliverables. The second network functions more like the DNS in that it is a high-speed, caching, runtime access mechanism for systems to obtain published archetypes. Archetypes migrate from the development network to the dissemination network via a quality assurance process which is undertaken by recognised expert bodies (e.g. national institute of classification, international college of oncologists) and overseen by recognised standards organisations.

2 Archetype Design Principles

Defining the information complexes required for interoperability is a complex task. There is a clear opportunity to solve many of the issues of healthcare computing, but if this is not taken up generally then little will be achieved. The approach taken in the *open*EHR architecture is described in this document and is a straight forward attempt to make these as generic and widely useful as possible. There are a number of developments that make this possible and worthwhile.

First, it is possible to express the archetypes - which must always be a constraint on an underlying information or data model - in terms of a 'harmonised' information model. The harmonised information model is a somewhat abstract expression of the common features of a number of key EHR information models in development. The has been possible through the efforts of those working in EHR standards development and, in particular, to the substantial harmonisation of the *open*EHR reference model, the HL7 Clinical Document Architecture and CEN ENV13606.

> **Design principle 1:** Archetypes for the EHR are expressed as constraints on a 'generic' EHR information model which shares the features of the major models being developed in this domain.

Second, there is a growing interest in making the many decision support tools that have been shown to deliver real benefits for patients and providers in specific implementations available in a generic form. This requires a view of the EHR which is generic and generalisable. The *open*EHR approach, and archetypes in particular, offer this possibility. In fact, archetypes provide the means of expressing any information requirements in terms of the harmonised information model. This is of interest to national data collections as well as applications which need to run queries on EHRs.

> **Design principle 2:** The archetypes allow a generic querying interface to the data to be developed, with specific queries being based on the 'archetypes' used to validate data during data entry or aquisition.

Finally, the ability to share information across language barriers is critical in a number of countries - Belgium and Canada being two examples. The archetype methodology has removed all language primacy to achieve international utility and acceptance.

> **Design principle 3:** There is no language primacy in archetypes: that is to say, archetypes should be able to be developed in any language - with the ability to add translations at any time. If this is taken up internationally then language translations must be able to be added *a posteriori*.

The methodology has also removed primacy of any particular terminology, allowing users (e.g. a rural doctor in a small country) without access to any terminological resources to build archetypes. When a terminology is deemed to be applicable in the archetype, bindings should be able to be added at any time.

> **Design principle 4:** Archetypes are neutral with respect to terminologies, i.e, archetypes are able to be developed without access to terminology - with the ability to add bindings to terminologies at any later time.

This rather liberal approach which underpins the design of new archetypes has to be balanced with a means of ensuring interoperability is achieved through the sharing of these concepts. As people travel and software is available globally, this sharing of knowledge in a form that can be automatically processed should be achieved, to the extent required, internationally. Such a requirement demands a thoroughly designed system to ensure that this is the case.

> **Design principle 5:** The system for developing archetypes should enable international sharing of personal health information (i.e. electronic health records), and the sharing of software which offers significant improvements in patient care, safety or population health.

It is clear that the laudible aim of international agreement on the concepts we need to share to enable interoperable health care systems will not literally be achieved - it would require too much time spent agreeing such matters. A further barrier is the pace of change in health care, and increasing diversity of care paradigms. Regardless, a *system* which supports collaborative archetype development can be developed. It would have features such as the following:

- it would allow local adaption to meet local needs while preserving the more general concepts that are required for generally available software and automatic processing;
- it would allow evolution of health care concepts over time, with increasing (or even decreasing) agreement about what should be shared locally, nationally and internationally; and
- it would enable the tools and processes required for standardisation and propagation of standardised concepts.

> **Design principle 6:** The archetype system should support local adaptation, general evolution of concepts and the process of increasing standardisation.

3 Template design principles

Templates describe what is possible at the time of data collection. Templates are expressed in terms of archetypes - they do not add meaning to the data.

> **Design principle 7:** Templates do not add to or in any way alter the semantics expressed in the underlying archetypes.

Templates, like archetypes, are documents and can be shared. Their use is to express the data collection requirements for specific clinical situations - many will be situation specific and some will express the requirements of individual users. From this we can deduce that just a few archetypes may lead to a plethora of templates.

As templates only express information in terms of archetypes re-authoring (or versioning) of an archetype is the only manner that templates may develop errors.

> **Design principle 8:** Templates must relate to specific versions of archetypes.

Templates express constraints through aggregation and through further constraining specific archetypes. They may describe:

- Which archetypes must be used (i.e. are mandatory)
- Which archetypes may be used (i.e. are optional)
- Which optional nodes of the archetypes are not utilised
- Which optional nodes of the archetypes are mandatory
- Which ‘fillers’ for a slot are optional
- Which ‘fillers’ for a slot are mandatory
- Which language (or languages) are available to the user
- Which terminology (or terminologies) are available to the user at each node
- Which optional values of any element are available
- Which optional value of any element is the default value

> **Design principle 9:** There is a defined set of constraints that can be expressed in a template - all relate to archetypes.

Because templates do not describe the semantics of the data, their use does not have to be recorded in the EHR - but it may be if required. The recording of a template is not part of the reference model information and should itself be archetyped so that the recording meets clinical needs.

> **Design principle 10:** The use of a template does not have to be recorded in the EHR.

4 The Archetype System

The Archetype System consists of two "networks" of communication: one 'authoring' or development network which operates among human authors of archetypes, and one 'Dissemination' network which provides inter-system communication of archetypes for runtime use. Archetypes are created or edited within the first environment, which includes limited local authoring of specialisations, and migrate into the second via a quality assurance process, as shown in FIGURE 1.

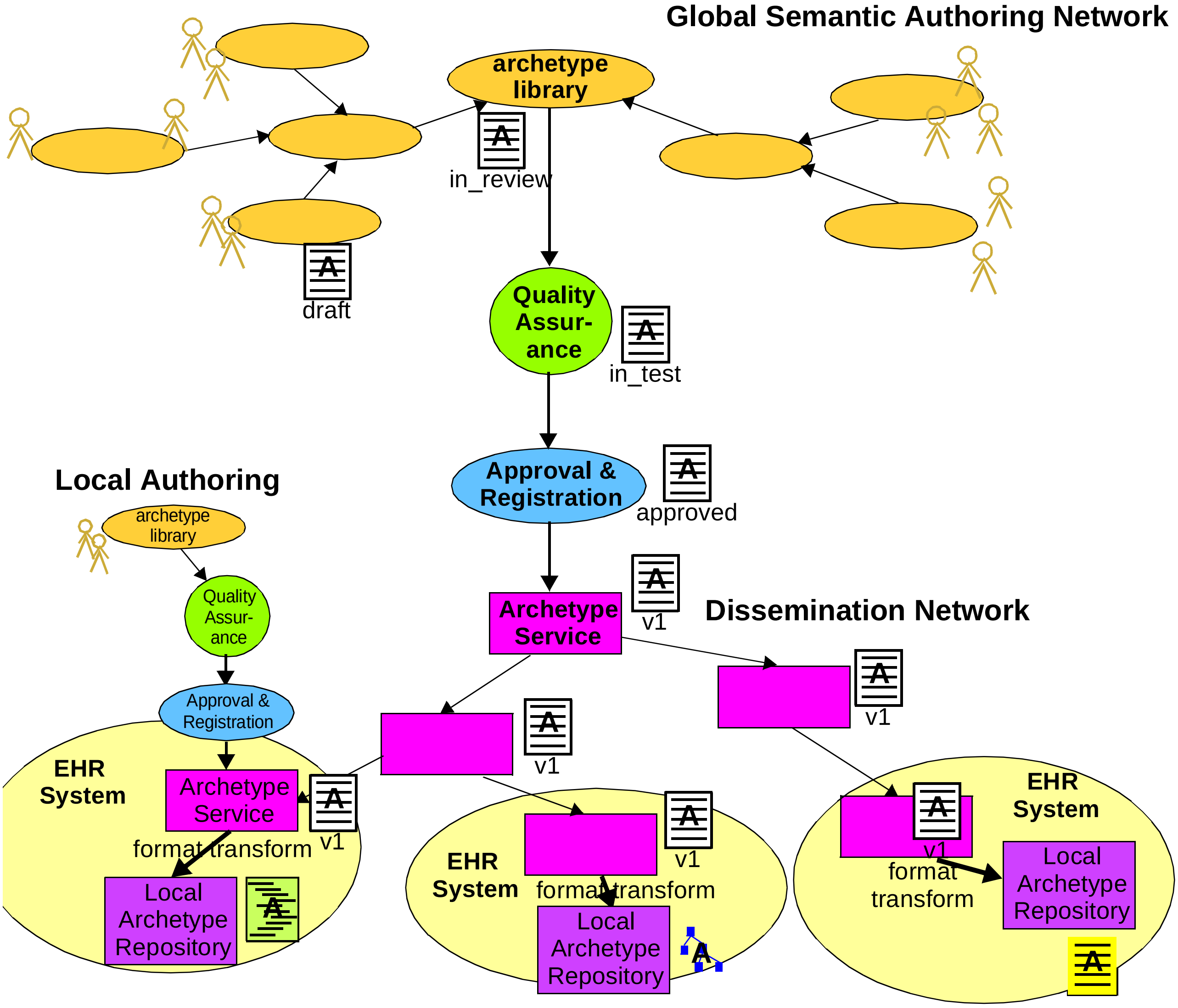

FIGURE 1 The Archetype System

This quality assurance and change control process is described fully in the *open*EHR Change Control Process documentation (FIXME - Tom - what is this called?).

Templates may be propagated using the Archetype system as required by users but there is no control over authoring of templates as this is not deemed to be appropriate or of any benefit. Naming of templates does not need to be controlled and hence an OID must be used for specific templates which are propogated through the dissemination network.

4.1 The Authoring Network

The authoring network takes place within the *open*EHR document and change control environment - the archetypes exist as stand alone documents and each alteration leads to a defined change set. The library provides a knowledge framework within which the archetypes are identified and classified, based on the semantic web, potentially using OWL. This knowledge framework provides firm links to the underlying reference model classes that have been archetyped, the relation between archetypes in terms of revisions, specialisations and versions and documents the discussion about each archetype change requests etc.

Each archetype has a 'custodian' who monitors the discussion and pulls together the views on what revisions are required. This person also moderates the discussion list relating to that archetype. The custodian maintains a document lifecycle in human readable form in addition to the document management system.

The library maintains a set of metadata regarding each archetype, some of which is duplicated in the 'description' part of the archetype itself.

4.1.1 Archetype Library Indexing

Although archetypes can be authored completely independently of any other resource, it is more usual to create them within the shared knowledge building environment, consisting of the *archetype library*, as well as terminological and ontological resources. As with software development, the first step to creating an archetype should be to determine whether any archetypes already exist corresponding to the requirements; a basic service is therefore to *find archetypes*. Two types of "indexing" are required, which we might term "literal" and "semantic". Literal indexing means that archetypes are indexed by concrete properties of individual archetypes, such as:

- identifier
- meta-data, such as author, version, lifecycle state, date of submission etc
- contained terms
- other textual contents, such as term definitions

This kind of indexing can easily be achieved by standard database indexing and content-based retrieval (if indexing is required on the contained text). Querying for archetypes based on combinations of properties can then be supported, such as "find all archetypes created by UK organisations within the last six months, which have 'pain' in the title".

In contrast, semantic indexing allows for much more sophisticated queries, but relies on making inferences on archetypes in the presence of other archetypes, terminologies and ontologies. For example, a user might wish to find all archetypes describing (clinical) examinations or investigations implicating the "liver". An archetype for XXXXXX blood test for hepatitis should clearly be returned in the result, but might not contain the word "liver", so literal indexing won't work in this case. However, assuming it contains the term or text for "hepatitis", a query into an medical ontology can establish its relevance, via the inference that for one of its terms - "hepatitis" - the assertion "has-infection-site liver" can be made. With a second query to determine that "blood test" "is-a-kind-of investigation", the archetype will be correctly retrieved.

4.1.2 Design-time Ontology Access

One it is determined whether a new archetype needs to be authored or an existing one adpated (usually specailised), the next step is to make the changes. Ontological and terminological resources within the environment can be used to enable the archetype author to understand the meaning of the archetype under construction, in a number of ways.

Finding internal archetype terms: internal terms defining node meanings may be derived from terms defined in published terminologies. For example, the author might wish to define an archetype node whose local code means “myocardial infaction”, a term which will easily be found in a number of terminologies and ontologies. If searching the available terminologies reveals the exact definition required by the author, she can create the local term, copy the definition, and include a “provenance” meta-data item in the term definition indicating the source.

Verifying value sets. In the ‘value’ position of an archetype leaf node, constraints are often defined which evaluate to a set of terms, raning from the simple such as “blood groups” to the more complex, such as “any disease which is-a auto-immune disease with-site liver”. If these constraints can be tested at authoring time against an appropriate terminology or ontology, the archetype designer will be able to ensure that the intended meaning is achieved.

4.2 The Dissemination Network

publish/subscribe relationship; flooding algorithm of servers

A publishing server has three calls:

Add_new_archetype

- Id
- Notes

Add_new_specialised_archetype

- Id
- Notes
- Predecessor

Add_new_archetype_version

- Id
- parent Id
- Notes

Obsolete_archetype

- Id
- Notes

Update_archetype -- revisions only - updated ontology etc

- Id
- Notes

4.3 Local Archetype Repositories

5 The Development Lifecycle

5.1 The Authoring Lifecycle

- initial creation
- information testing & debugging

6 Archetype Identification

Archetypes can be identified with various kinds of identifiers. We propose only two here: the ISO Oid and a multi-axial meaningful identifier. The syntax described in this paper is not dependent on the particular form of the identifier. Identifiers are declared in the heading section of the archetype, e.g.

```
archetype openehr-ehr-observation.haematology.v1
```

6.1 Multi-axial Archetype Identifier

A meaningful multi-axial identifier has a different purpose from the OID-based id: it encodes the partitioning of the archetype concept space in the identifier. Each identifier instance denotes a single archetype within a versioned 2-dimensional space, with the dimensions being:

- reference model entity, i.e. target of archetype
- domain concept

As with any multi-axial identifier, the underlying principle of an archetype id is that all parts of the id must be able to be considered immutable. This means that no variable characteristic of an archetype (e.g. accrediting authority, which might change due to later accreditation by another authority, or may be multiple) can be included in its identifier.

The inclusion of versioning and lifecycle state to the identifier have the effect of defining a 4-dimensional space. The syntax of an Archetype id is as follows:

archetype_id: qualified_model_entity '.' domain_concept '.' version_id ['.' lifecycle_state]

qualified_model_entity: model_originator '-' model_name '-' model_entity_name

domain_concept: concept_name { '-' specialisation }*

version_id: 'v' NUMBER

model_originator: NAME

The name of the authority issuing the underlying reference model to which this archetype applies e.g. openEHR, HL7, CEN etc.

model_name: NAME

The name of the model as labelled by the model_originator e.g. ehr, demographics, CDA etc.

model_entity_name: NAME

The name of the class in the model (model_name) to which this archetype applies.

domain_concept: NAME

A unique string that describes the concept expressed in the archetype e.g. weight

specialisation: NAME

A string which describes this specialisation e.g. 'birth' as a specialisation of 'weight' to describe birth-weight

NUMBER: [0-9]*

NAME: [a-z][a-z0-9()/%$#&]*

The field meanings are as follows:

model_originator: id of organisation originating the model on which this archetype is based;

model_name: name of the model on which this archetype is based;

model_entity_name: entity type in the model;

domain_concept: the domain concept name, including any specialisations;

version_id: numeric version identifier;

lifecycle_state: state of this archetype in its lifecycle

Examples of archetype identifiers include:

```
openehr-ehr-organiser.physical_examination.v2.draft
openehr-ehr-organiser.physical_examination-prenatal.v1
hl7-rim-act.progress_note.v1.in_test
```

A basic rule for the multi-axial archetype identifier is that it changes as soon as anything is done to the archetype which makes data created using the previous form invalid with respect to the changed form. For this reason, version is included in the identifier (see discussion below).

6.2 ISO Oid

ISO Oids can be used to unambiguously identify archetypes within storage systems, online repositories etc., regardless of where the archetype sits in the concept space. In order that archetypes can be authored at any place and time without access to OIDs, the OID is optional at all times until publication. The OID can have been issued at any point in the publication lifecycle, but must be unique in the authority's database.

Local specialisations do not require OIDs.

7 Quality Assurance

7.1 Technical Validation

7.2 Semantic Validation

8 Registration

9 Archetype Propagation

9.1 Authoritative Servers

9.2 Digital Signing

9.3 Archetype Service Interface

9.4 Archetype Repositories

Archetypes and Templates in the local environment:

- converted from shared format to local computable format
- only those archetypes used in a given EHR or other server

9.5 Archetype Indexes

Archetype index maps of data built from archetypes:

- hierarchical indexes containing only those branches of archetypes actually chosen a) by templating and b) due to further runtime choices

10 Creation of New Archetypes

10.1 Semantic Rules

To Be Determined: Note: see rules described in original archetype paper and in ADL manual.

10.1.1 Specialisation

Basic Rule:

- Specialised archetype must not create data that is not a valid instance of parent archetype

Consequence:

- narrowed constraints ok

10.1.2 Revision

Basic Rule:

- Data created by precursor of revised archetype must be compatible with revised version
- Data created by revised version must be valid w.r.t. precursor

Consequence:

- wider constraints ok

10.1.3 New Versions

Basic Rule:

- new archetype may be neither clean superset or subset; may be incompatible
- must supply a conversion argorithm for existing data (equivalent to a viewing algorithm for existing data)

10.2 Local Authoring

10.3 Managing Conflicts

K References

Publications

1 Beale T. *Archetypes: Constraint-based Domain Models for Future-proof Information Systems*. OOPSLA 2002 workshop on behavioural semantics.
Available at http://www.deepthought.com.au/it/archetypes.html.

2 Beale T. *Archetypes: Constraint-based Domain Models for Future-proof Information Systems*. 2000.
Available at http://www.deepthought.com.au/it/archetypes.html.

3 Beale T, Heard S. *A Shared Language for Archetypes and Templates - Part I*. 2003.
Available at http://www.deepthought.com.au/health/archetypes/archetype_language_2v0.6.2.doc.

4 Beale T, Heard S. *A Shared Language for Archetypes and Templates - Part II*. 2003.
Available at http://www.deepthought.com.au/health/archetypes/archetype_language.doc.

5 Dolin R, Elkin P, Mead C *et al. HL7 Templates Proposal*. 2002.
Available at http://www.hl7.org.

6 Gruber T R. *Toward Principles for the Design of Ontologies Used for Knowledge Sharing*. in Formal Ontology in Conceptual Analysis and Knowledge Representation. Eds Guarino N, Poli R. Kluwer Academic Publishers. 1993 (Aug revision).

Resources

7 OWL - Web Ontology Language.
See http://www.w3.org/TR/2003/CR-owl-ref-20030818/.

8 *open*EHR. Knowledge-enabled EHR and related specifications.
See http://www.openEHR.org.

9 *open*EHR. EHR reference model.
See http://www.openEHR.org.

END OF DOCUMENT