

The openEHR Health Computing Platform

Introduction

The architecture of *openEHR* is a response largely to the requirement for *semantic interoperability*. Semantic interoperability is the ability of computers and software to mutually understand the meaning of information. It requires information to be computable, not just shareable, and is required in the following places.

- **in the 'application stack'**: i.e. from the GUI down to the database (regardless of the actual method of deployment of these things). It is essential that what is presented at the GUI is understood the same way by the software as it is by the human user. The meaning of information must be preserved from this point down through layers of business logic to its representation in the persistence (data storage) layer. If this is not the case, information entered by a user will not safely reach the part of the system which allows it to be shared with other users or other systems, or even the same user at a later time.
Achieving semantic coherence in the application stack is not always as simple as it seems given that more than one vendor may be responsible for various parts of the software.
- **between systems**: the meaning of the information captured in any particular application or system needs to be preserved when it is transmitted to another system. This applies to systems within the same provider enterprise and across enterprises.

Addressing this need in a sustainable way has not proven easy. One of the biggest challenges has been how to deal with context- and site-dependent capture and use of data, while standardising its meaning in a context-independent way (a 'blood pressure measurement' doesn't change its meaning after all) and yet avoid the trap of hard-modelling things in software, creating an unmaintainable solution.

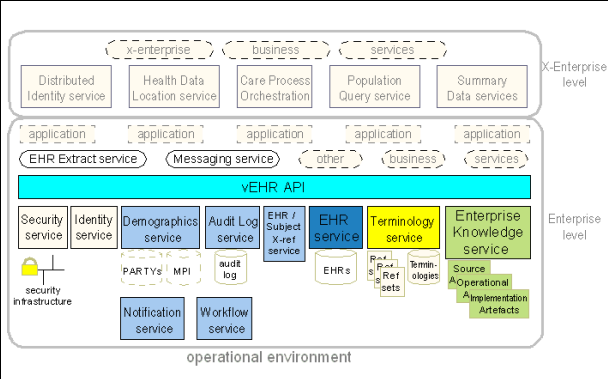
Some theorists in the field have thought that terminology was the sole answer to semantic interoperability, but this has turned out to be incorrect for a number of reasons, the main one being that terminology does not provide an appropriate place for modelling the compositional structures or semantics of structured information. Terminologies also do not represent constraints well, nor data types other than the textual. Terminology therefore has an important place in the infrastructure, but is not the totality of it.

Others claim that defining messages is sufficient to enable semantic interoperability. However, while messages of some form or other are literally needed between systems, using messages as the design basis for a semantically interoperable infrastructure does not work very well. Firstly, because they do not take account of the application stack - they treat applications as black boxes. Secondly, the basic notion of a message is a carrier of a *change of state* or an event, rather than (necessarily) a coherent model of structured information. Thirdly, message models usually contain a lot of details to do with the messaging activity, i.e. sending, receiving, acknowledgements, and so on. A proper solution to the problem of semantically interoperable content cannot avoid accounting for what happens inside systems as well as what happens between them. A more appropriate way to model messages is as conversations between web-services, where the message schemas are machine-generated from other more reusable models of information.

Service Architecture

An engineering view of the architectural framework of <i>openEHR</i> is that of a service-oriented system of systems, with distinct	
---	--

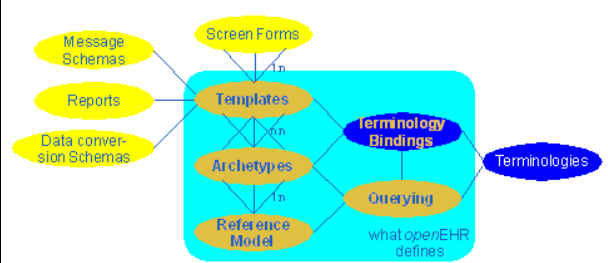
services for EHR, demographics, workflow, administration, knowledge artefacts (including terminology), identification, security, orchestration and discovery. One of the major features of *openEHR* is the a semantic architecture for the information-rich services, e.g. EHR, demographics and administration, and their corresponding user applications (e.g. doctors desktop, nurse systems).



Semantic Architecture

The *openEHR* information architecture takes the following form:

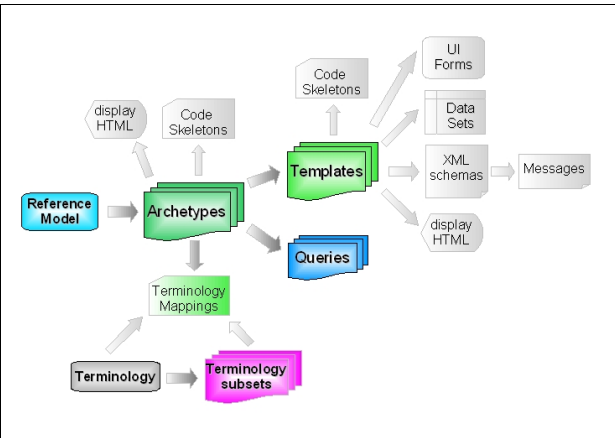
- Four levels of information organisation:
 - The cognitive **user interface** – a flexible approach to data capture and viewing
 - **Templates** - data capture sets for each *business process event*
 - **Archetypes** - standardised semantics of the data points in data capture sets, defined on the basis of *topic*
 - The **reference model** - standardised data representation, enabling interoperability
- Standardised **querying** capability based on archetype paths and terminology
- Standardised interface to **terminology** for inferencing



The *openEHR* semantic architecture is visualised on the right. The part shown in cyan is defined by *openEHR* specifications. A key feature of this architecture is the two middle model layers, i.e. archetypes and templates. Archetypes are models of clinical content, defined on the basis of topic, such as 'blood pressure measurement', 'physical examination' and 'diagnosis'. Templates are artefacts that reuse data points defined within archetypes to create refined data sets particular to use context, which almost always means that they correspond to a particular kind of business event (health service event) - e.g. a certain kind of patient visit. Templates are then the basis for creating screen forms, message schemas and other derived artefacts for information use.

The relationships and dependency between different kinds of artefacts in this architecture are

visualised on the right. Within the artefact ecosystem, archetypes are of key importance, since they constitute the basis for building templates, building queries and defining mappings to terminology. It is therefore important that they are as far as possible independent of particular concrete expressions or uses, such as GUI forms, messages, and executable code. These latter artefacts should all be machine generated from more abstract artefacts such as archetypes and templates.

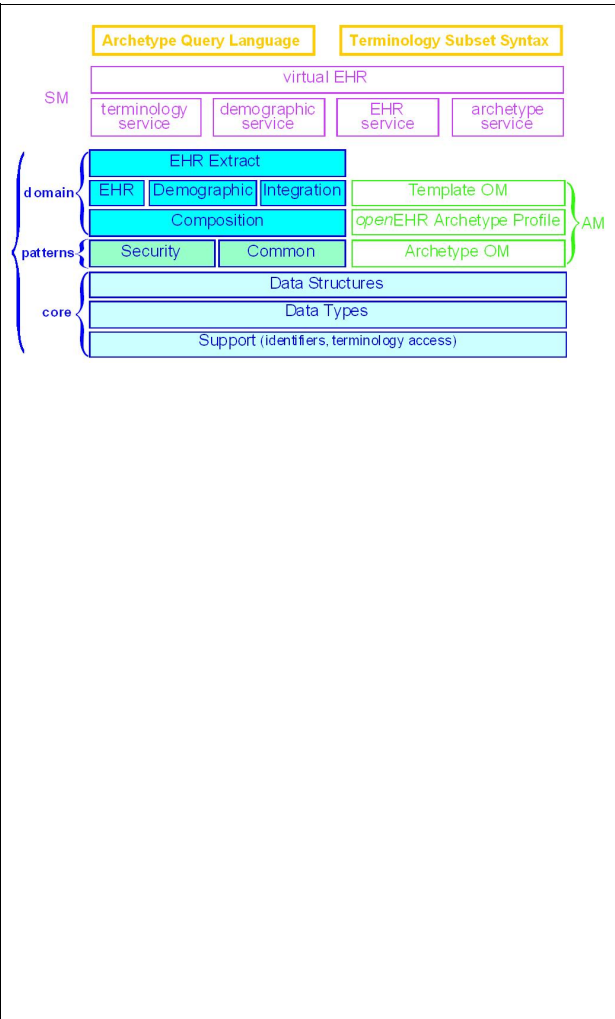


The openEHR Reference Model (RM)

The openEHR RM formalises the Electronic Health Record as follows:

1. Information model (RM)
 This is the model that describes the health record itself - not the clinical data that are contained within it. The reference model deals with containers such as Folders and Compositions. Compositions are a broader concept than documents - but include documents. Examples of Compositions are an ECG report, a progress note, a laboratory report and a referral. The Composition is the minimum unit of communication and committal to the EHR. The openEHR Reference Model specifications are available from [the specification page](#)¹; [online UML](#)².

2. Archetype Model (AM)
 Archetypes are descriptions of valid Entries, Sections and Compositions. These are expressed in a formal manner which enables them to be shared between systems. A blood pressure archetype represents a description of all the information a clinician might want to report about a blood pressure measurement, and may include some aspects which are mandatory. A 'SOAP' archetype describes the sections of a problem oriented health note and which entries are valid under each section - for example only diagnoses may be allowed under the 'A' section. Templates are logical models of user forms - and are described in terms of choices of archetypes whose data are captured on a particular form. See the [Archetypes and Templates FAQ](#)³. The openEHR Archetype Model specifications are available from the [openEHR Archetype Model specifications](#).



1. <http://www.openehr.org/Specification/FAQs/Release-1.0.1/publishing/roadmap.html>

2. http://www.openehr.org/Specification/FAQs/Release-1.0.1/publishing/architecture/computable/UML/uml_start_view.html

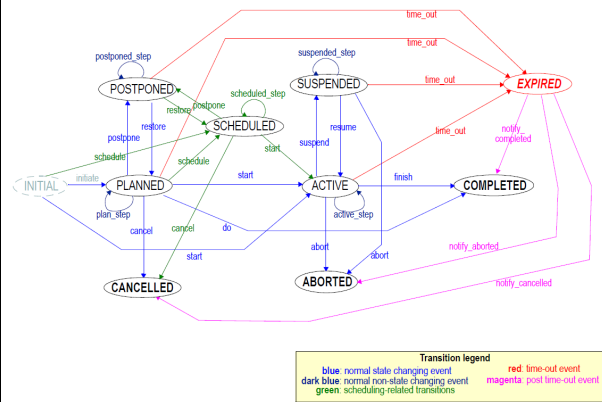
3. daisy:92-OE (Archetypes FAQs)

[specification page](#)⁴; [online UML](#)⁵.

3. Service Model (SM)

This is the computational viewpoint of the *openEHR* architecture. The service model consists of service definitions for the major services in the EHR computing environment. These are largely derived from existing work in OMG Corbamed, CEN HISA and implementation experience.

The dynamic view. In real world healthcare delivery, decisions lead to interventions, usually expressed as 'orders' and subsequent 'actions'. In *openEHR*, the INSTRUCTION and ACTION Entry types are used to represent these. When Actions are performed, the state of the order can be recorded, using the states of a standard state machine shown to the right. Concrete workflow steps can be associated with each abstract state. This enables concrete workflows (different for e.g. imaging order of surgery order) to be modelled and connected to a standard state machine, which then allows standardised querying of what interventions are 'active', 'suspended', 'complete', etc.



Advantages of the openEHR architecture

The advantages of the architecture include the following:

- *Semantic coherence* in the application stack (all layers of software know what the data mean)
- A *high level of re-use* of artefacts – define once, reuse many times
- A *single, stable reference model* for sharing clinical and related information
- A standardised query language for writing *portable queries*
- A standardised, re-usable way of connecting to *terminology*

The Architectural Overview ([PDF](#)⁶, [HTML](#)⁷) provides a good summary of the technical details of *openEHR*.

4. <http://www.openehr.org/svn/specification/TAGS/Release-1.0.1/publishing/roadmap.html>

5. http://www.openehr.org/svn/specification/TAGS/Release-1.0.1/publishing/architecture/computable/UML/uml_start_view.html

6. <http://www.openehr.org/releases/1.0.1/architecture/overview.pdf>

7. <http://www.openehr.org/releases/1.0.1/html/architecture/overview/Output/front.html>