The *open*EHR Technical Roadmap

Editors:{T Beale, S Heard}[1], *{D Kalra, D Lloyd}*[2]

Revision: 1.6

Pages: 21

1. Ocean Informatics Australia
2. Centre for Health Informatics and Multi-professional Education, University College London

The *open*EHR foundation

is an independent, non-profit community, facilitating the creation and sharing of health records by consumers and clinicians via open-source, standards-based implementations.

Founding Chairman	David Ingram, Professor of Health Informatics, CHIME, University College London
Founding Members	Dr P Schloeffel, Dr S Heard, Dr D Kalra, D Lloyd, T Beale

email: info@openEHR.org **web**: http://www.openEHR.org

Amendment Record

Issue	Details	Who	Completed
1.6	**CR-000147**. Make DIRECTORY Re-usable **CR-000167**. Move AOM description package to resource package in Common IM. **CR-000185**: Improved EVENT model.	T Beale	18 Dec 2005
1.5.1	Typo corrections.	G Hayworth	19 Apr 2005
1.5	Rewritten to correspond to most recent modelling structures.	T Beale	12 Feb 2005
1.4.2	Improved discussion, added content on archetypes. Updated package structure. Added AM package structure.	T Beale	28 Mar 2003
1.4.1	Corrected typos and grammatical errors.	D Lloyd	05 Mar 2003
1.4	Major update to bring into line with validated RM and AM docs.	T Beale	28 Feb 2003
1.3	Minor additions	T Beale	10 Dec 2002
1.2	Reorganised, new headings, ITS explanation.	T Beale	10 Nov 2002
1.1	Updated diagrams, text	T Beale	26 Oct 2002
1.0	Initial Writing	T Beale	11 Oct 2002

Acknowledgements

The work reported in this paper has been funded in by The University College, London and Ocean Informatics, Australia.

Table of Contents

1 Introduction

1.1 Purpose

This document provides an overview of the *open*EHR specifications for the Reference Model (RM), Service Model (SM) and Archetype Model (AM). The intended audience includes:

- Health data managers;
- Standards bodies producing health informatics standards;
- Software development organisations using *open*EHR;
- Academic groups using *open*EHR;
- The open source healthcare community;
- Medical informaticians and clinicians interested in health information.

The overall vision and in particular health system and clinical practice ideals of *open*EHR are described in other documents.

1.2 Related Documents

The following documents should be read prior to this document:

- Introducing *open*EHR

1.3 Status

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

1.4 Overview of Document

This document includes the following sections:

- An overview of the specification structure;
- An overview of the abstract models is given - these are the main technical specifications of *open*EHR;
- Rules for deriving implementation technology specifications (ITSs) are described.

2 Overview

2.1 The *open*EHR Specification Project

FIGURE 1 illustrates the *open*EHR Specification Project, a key project in the *open*EHR "technical space". The project consists of requirements, architectural specifications, implementation technology specifications (ITSs), and conformance specifications. The focus of this document is the architectural and implementation technology specifications (ITSs).

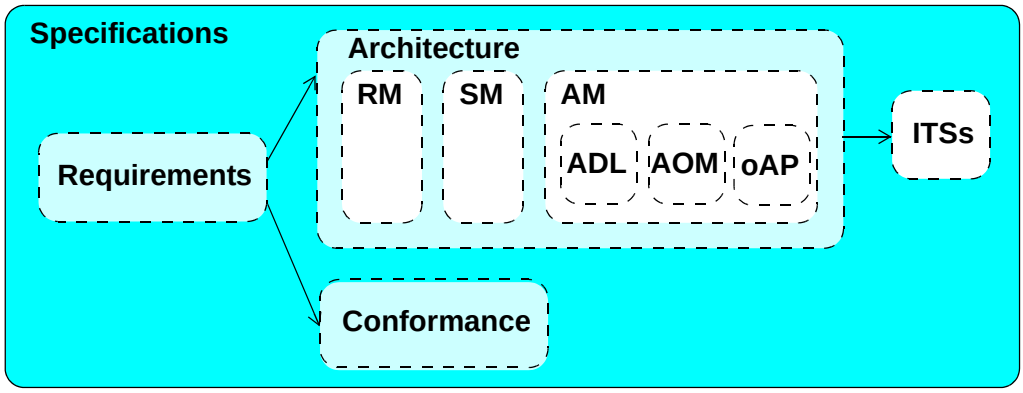

FIGURE 1 *open*EHR Specification project

The architecture specifications consist of the Reference Model (RM), the Service Model (SM) and Archetype Model (AM). The first two correspond to the ISO RM/ODP information and computational viewpoints respectively.

All of the architecture specifications published by *open*EHR are defined as a set of *abstract* models, using the UML notation and formal textual class specifications. These models remain the primary references for all semantics, regardless of what is done in any implementation domain. The *open*EHR Modelling Guide describes the semantics of the models. The presentation style of these abstract specifications is deliberately intended to be clear, and semantically close to the ideas being communicated. Conversely, the abstract specifications do not follow idioms or limitations of particular programming languages, schema languages or other formalisms. All such expressions are treated as ITSs, for which explicit mappings generally have to be developed and described (since almost no formalism natively implements complete UML semantics).

There are numerous implementation technologies, ranging from programming languages, serial formalisms such as XML, to database and distributed object interfaces. Each of these has its own limits and strengths. The approach to implementing any of the *open*EHR abstract models in a given implementation technology is to firstly define an "implementation technology specification" (ITS) for the particular technology, then to use it to formally map the abstract models into expressions in that technology.

2.2 Relationship to Standards

The *open*EHR specifications make use of available standards where relevant, and as far as possible in a compatible way. However, for the many standards have never been validated in their published form (i.e. the form published is not tested in implementations, and may contain errors), *open*EHR make adjustments so as to ensure quality and coherence of the *open*EHR models. There are many

good standards in the health computing space which *open*EHR uses and will use in the future, as they become available. In general, "using" a standard in openEHR may mean defining a set of classes which map it into the *open*EHR type system, or wrap it or express it in some other compatible way, allowing developers to build completely coherent *open*EHR systems, while retaining compliance or compatibility with standards. The standards relevant to *open*EHR fall into a number of categories as follows.

Standards by which *open*EHR can be evaluated

These standards define high-level requirements or compliance criteria which can be used to provide a means of normative comparison of *open*EHR with other related specifications or systems. The following ones have been used for this purpose so far:

- ISO TC 251 TS 18308 - Technical Specification for Requirements for an EHR Architecture.

Standards which have influenced the design of *open*EHR specifications

The following standards have influenced the design of the *open*EHR specifications:

- **OMG HDTF** Standards - general design
- **CEN EN 13606:2005**: Electronic Health Record Communication
- **CEN HISA 12967-3**: Health Informatics Service Architecture - Computational viewpoint

Standards which have influenced the design of *open*EHR archetypes

The following standards are mainly domain-level models of clinical practice or concepts, and are being used to design *open*EHR archetypes and templates.

- **CEN HISA 12967-2**: Health Informatics Service Architecture - Information viewpoint
- **CEN ENV xxxx**:General Purpose Information Components (GPICs)
- **CEN ENV 13940**: Continuity of Care.

Standards which are used "inside" *open*EHR

The following standards are used or referenced at a fine-grained level in *open*EHR:

- **ISO 8601**: Syntax for expressing dates and times (used in *open*EHR Quantity package)
- **ISO 11404**: General Purpose Datatypes (mapped to in *open*EHR `assumed_types` package in Support Information Model)
- **HL7 UCUM**: Unified Coding for Units of Measure (used by *open*EHR Data types)
- **HL7v3 GTS**: General Timing Specification syntax (used by *open*EHR Data types).
- some HL7v3 domain vocabularies are mapped to from the *open*EHR terminology.

Standards which require a conversion gateway

The following standards are in use and require data conversion for use with *open*EHR:

- **CEN EN 13606:2005**: Electronic Health Record Communication - near-direct conversion possible, as *open*EHR and CEN EN 13606 are actively maintained to be compatible.
- **HL7v3 CDA**: Clinical Document Architecture (CDA) release 2.0 - fairly close conversion may be possible.
- **HL7v3 messages**. Quality of conversion currently unknown due to flux in HL7v3 messaging specifications.

3 *open*EHR Specifications Design Approach

3.1 Goals

Like other published specifications, the *open*EHR specifications try to satisfy a wide range of requirements. There are many sources of requirements, including:

- *diverse implementors*: builders of small, large, generalist, specialist information systems
- *diverse cultures*: many cultural factors, including the practice of medicine, language, and many cultural specificities have an impact on health information, and hence on specifications such as those created by *open*EHR.
- *standards*: there are numerous standards which purchasers and users of health information systems are required to use. The main health information standards relevant to the openEHR specifications are CEN TC/251 specifications (ENV 13606 and others), HL7 version 3 messaging specifications, and ISO TC215 specifications. Numerous technology and 'lower' level standards are of course taken into account (e.g. ISO data type semantics).
- *legislation*: legislated requirements are often encoded in standards, but not always.
- *medico-legal & ethical requirements*: legal and ethical requirements have direct consequences for health information systems and models.

The approach taken by *open*EHR to developing specifications has been to observe three priorities above all others: *best-of-breed semantics*, *implementability*, and *wide applicability.* Clearly, these three are sometimes in conflict, and one of the challenges of developing the specifications has been to make good choices based on a consideration of the final result in the real world. The huge diversity of requirements implied by the list above is met within these tenets. As a result, the *open*EHR models are guaranteed (inasmuch as one can guarantee the outcome of any human enterprise) to be implementable, and in general, to have reference software implementations available. However, like any model meeting very diverse requirements, they are not guaranteed to meet any one user's requirements *exactly*; such differences have to be managed in implementation.

The specifications also try to respect standards as much as possible, while ensuring quality and implementability. However, where problems are found in existing standards, the approach has usually been to use better modelling semantics, and supply a transformation algorithm or function, as appropriate. Sometimes what may appear to be odd choices in models issued by standards bodies are actually replicated in the *open*EHR specifications, where doing so has been verified not to compromise the model In a few cases, it makes the model more "open" than it probably should be, and where this has occurred, it is clearly documented.

3.2 Underlying Paradigms

The Engineering Process

The *open*EHR specifications are founded upon a number of paradigms of computing. The first is the engineering process, which recognises that development of formal systems must be requirements-based, and that there are a number of formal expressions at various levels of granularity leading toward implementations. The engineering process inherently allows for negative (correcting) feedback, otherwise known as maintenance and enhancement. FIGURE 2 illustrates the process. Within this process, the *open*EHR specifications document a fair amount of the analysis and design thinking, as well as the actual models on which implementations are based. The *open*EHR specifications are change managed just like any other good engineering specification - taking into account ongoing

feedback from implementation and deployment experiences. In this way, the specifications are living, formal, documents undergoing continual testing and improvement. Disciplined version management ensures that changes are made in a coherent way, and that stable versions are always available to users.

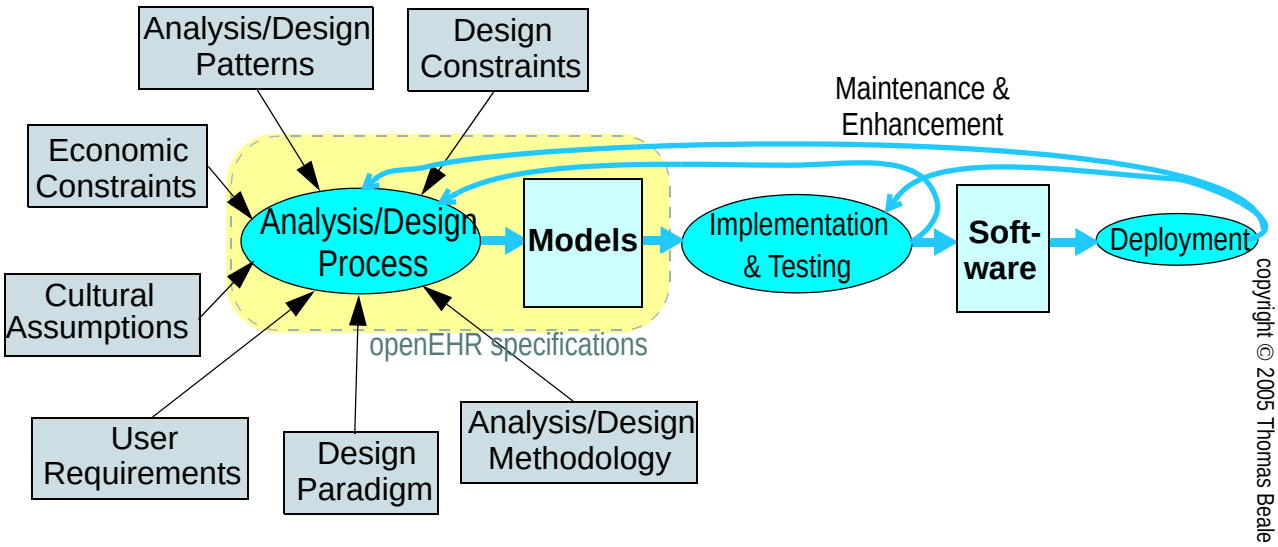

FIGURE 2 The Engineering Process

Two-level Modelling

The first paradigm on which *open*EHR models are based is known as "two-level" modelling, described in [2]. The information models constitute a first level of modelling, while formal structured constraint definitions of clinical concepts - *archetypes* - are the second level. Archetypes are themselves instances of an *archetype model*, which defines a language in which to write archetypes. Archetypes are general-purpose, re-usable, and composable. They are used at runtime by building *templates* from them. A template is a tree of archetypes each in turn constraining instances of the coarse-grained parts of the information model, e.g. in the EHR model, Compositions, Section hierarchies, Entries and so on. Thus, while there are likely to be archetypes for such things as "biochemistry results" (an Entry 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", "antenatal exam" and so on. Templates correspond closely to screen forms and printed report definitions.

When a template is ready for use, it is used to create default data structures and to validate data input, ensuring that all data in the EHR conform to the constraints defined in the archetypes comprising the template. In particular, it conforms to the *path* structure of the archetypes, as well as their terminological constraints. Which archetypes were used at data creation time is written into the data, in the form of both archetype identifiers at the relevant root nodes, and archetype node identifiers - normative node names, which are the basis for paths. When it comes time to modify or query data, these archetype data enable applications to retrieve and use the original archetypes, ensuring modifications respect the original constraints, and allowing queries to be intelligently constructed.

Separation of Responsibilities

A second key design paradigm used in *open*EHR is that of separation of responsibilities within the computing environment. Complex domains are only tractable if the functionality is first partitioned into broad areas of interest, i.e. into a "system of systems" [5]. This principle has been understood in computer science for a long time under the rubrics "low coupling", "encapsulation" and "componentisation", and has resulted in highly successful frameworks and standards, including the OMG's

CORBA specifications and the ISO Reference Model for Open Distibuted Processing (RM-ODP) [4]. Each area of functionality forms a focal point for a set of models formally describing that area, which, taken together usually describe a distinct information system or service.

FIGURE 3 illustrates a notional health information environment containing numerous services, each denoted by a bubble. Typical connections are indicated by lines, and bubbles closer to the centre correspond to services closer to the core needs of clinical care delivery, such as the EHR, terminology, demographics/identification, medical reference data and so on. The figure is not intended to be a software engineering diagram, nor is it definitive in any way; nor exhaustive. It simply illustrates a plausible set of distinct responsibilities in a distributed health computing environment, providing a basis for the models of *open*EHR. Of the services shown on the diagram, *open*EHR currently provides specifications only for the more central ones, including EHR, Demographics, and Workflow.

Since there are standards available for some aspects of many services, such as terminology, image formats, messages, EHR Extracts, service-based interoperation, and numerous standards for details such as date/time formats and string encoding, the *open*EHR specifications often act as a mechanism to create coherent structural definitions in the informational and computational viewpoints that integrate existing standards.

Separation of Viewpoints

The third computing paradigm used in *open*EHR is a natural consequence of the separation of responsibilities, namely the *separation of viewpoints*. When responsibilities are divided up among distinct components, it becomes necessary to define a) the information that each processes, and b) how they will communicate. These two aspects of models constitute the two central “viewpoints” of the ISO RM/ODP model [4], which are as follows:

Enterprise: concerned with the business activities, i.e. purpose, scope and policies of the specified system.

Information: concerned with the semantics of information that needs to be stored and processed in the system.

Computational: concerned with the description of the system as a set of objects that interact at interfaces - enabling system distribution.

Engineering: concerned with the mechanisms supporting system distribution.

Technological: concerned with the detail of the components from which the distributed system is constructed.

The *open*EHR specifications accordingly include an information viewpoint - the *open*EHR Reference Model - and a computational viewpoint - the *open*EHR service model. The Engineering and Technological viewpoints are the direct concerns of implementors, and indirectly related to *open*EHR view ITSs, and implementation guidelines. An important aspect of the division into viewpoints is that there is generally not a 1:1 relationship between model specifications in each viewpoint. For example, there might be a concept of “health mandate” (a CEN ENV13940 Continuity of Care concept) in the enterprise viewpoint. In the information viewpoint, this might have become a model containing many classes. In the computational viewpoint, the information structures defined in the information viewpoint are likely to recur in multiple services, and there may or may not be a “health mandate” service. The granularity of services defined in the computational viewpoint corresponds most strongly to divisions of function in an enterprise or region, while the granularity of components in the information view points corresponds to the granularity of mental concepts in the problem space, the latter usually being more fine-grained. The *open*EHR archetype model (AM) is in addition to the reference and service models, and provides the knowledge-enabling of the computational environment.

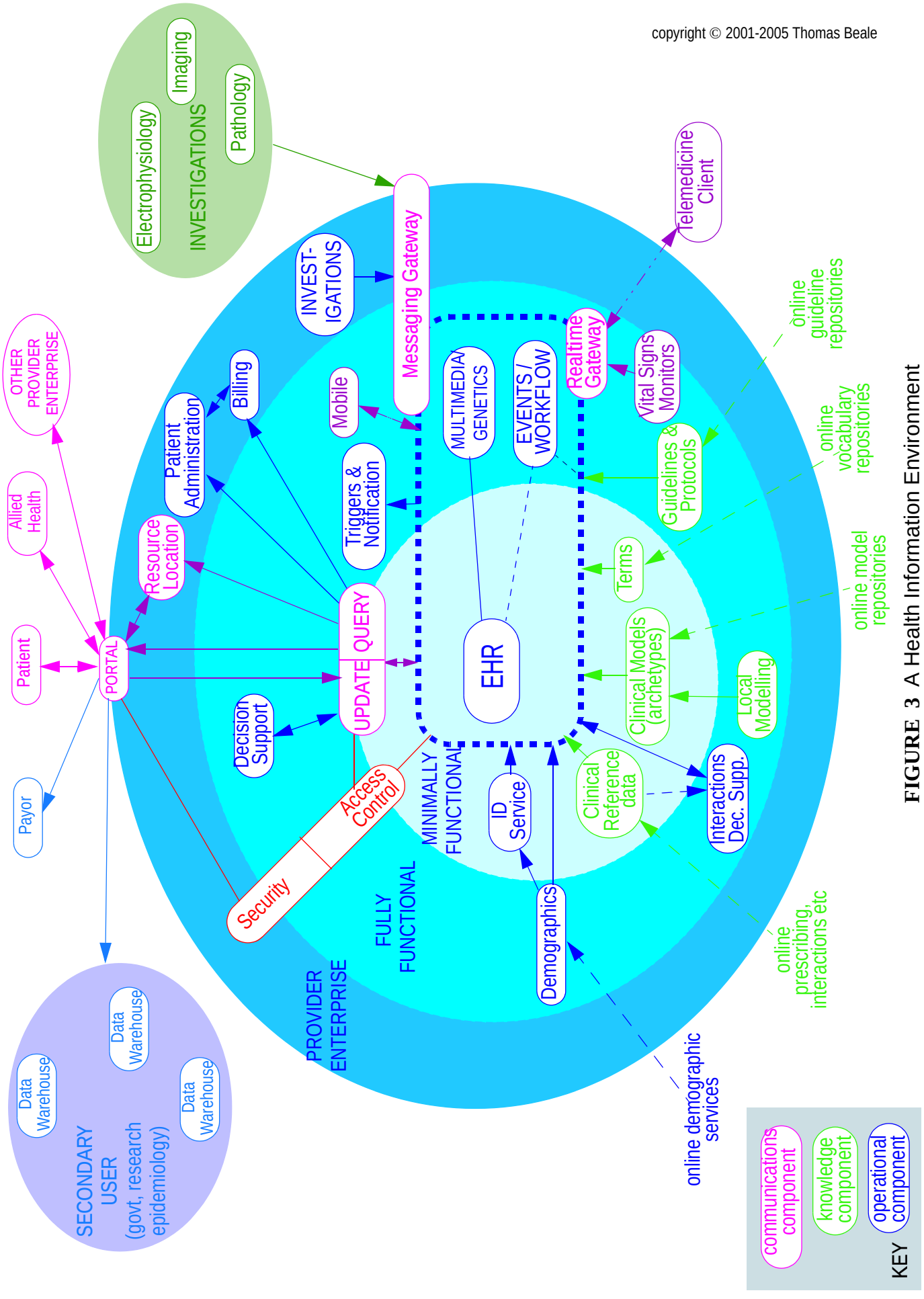

FIGURE 3 A Health Information Environment

4 *open*EHR Package Structure

FIGURE 5 illustrates the overall package structure of the *open*EHR formal specifications. Three major packages are defined: `rm`, `am` and `sm`. Packages describing detailed semantics appear in one of these outer packages, which may also be thought of as namespaces. They are conceptually defined within the `org.openehr` namespace, which is usually represented in UML as further packages. In some implementation technologies (e.g. java), the `org.openehr` namespace may actually be used within program texts.

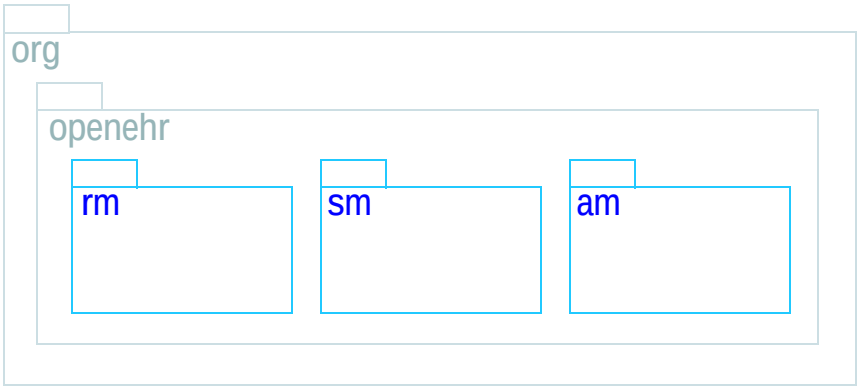

FIGURE 4 Global Package Structure of *open*EHR

4.1 Reference Model (RM)

Within the any given namespace, each package defines a local context for definition of classes. FIGURE 5 illustrates the package structure in the RM namespace. An informal division into “scientific computing” and “health information “ is shown. The packages in the latter group are generic, and are used by all *open*EHR models, in all the outer packages. Together, they provide identification, access to knowledge resources, data types and structures, versioning semantics, and support for archetyping. The packages in the former group define the semantics of enterprise level health information types, including the EHR and demographics.

Each outer package in FIGURE 5 corresponds to one document[1], documenting an “information model” (IM). Where packages are nested, the inner packages cannot exist outside of their parent package. The package structure will normally be replicated in all ITS expressions, e.g. XML schema, programming languages like Java, C# and Eiffel, and interoperability definitions like IDL and .Net.

The following sub-sections provide a brief overview of the RM packages.

Support Information Model

This package describes the most basic concepts, required by all other packages, and is comprised of the Definitions, Identification, Terminology and Measurement packages. The semantics defined in these packages allow all other models to use identifiers and to have access to knowledge services like terminology and other reference data. The support package includes the special package `assumed_types`, describing what basic types are assumed by *open*EHR in external type systems; this package is a guide for integrating *open*EHR models proper into the type systems of implementation technologies.

1. with the exception of the EHR and Composition packages, which are both described in the EHR Reference Model document; this may change in the future.

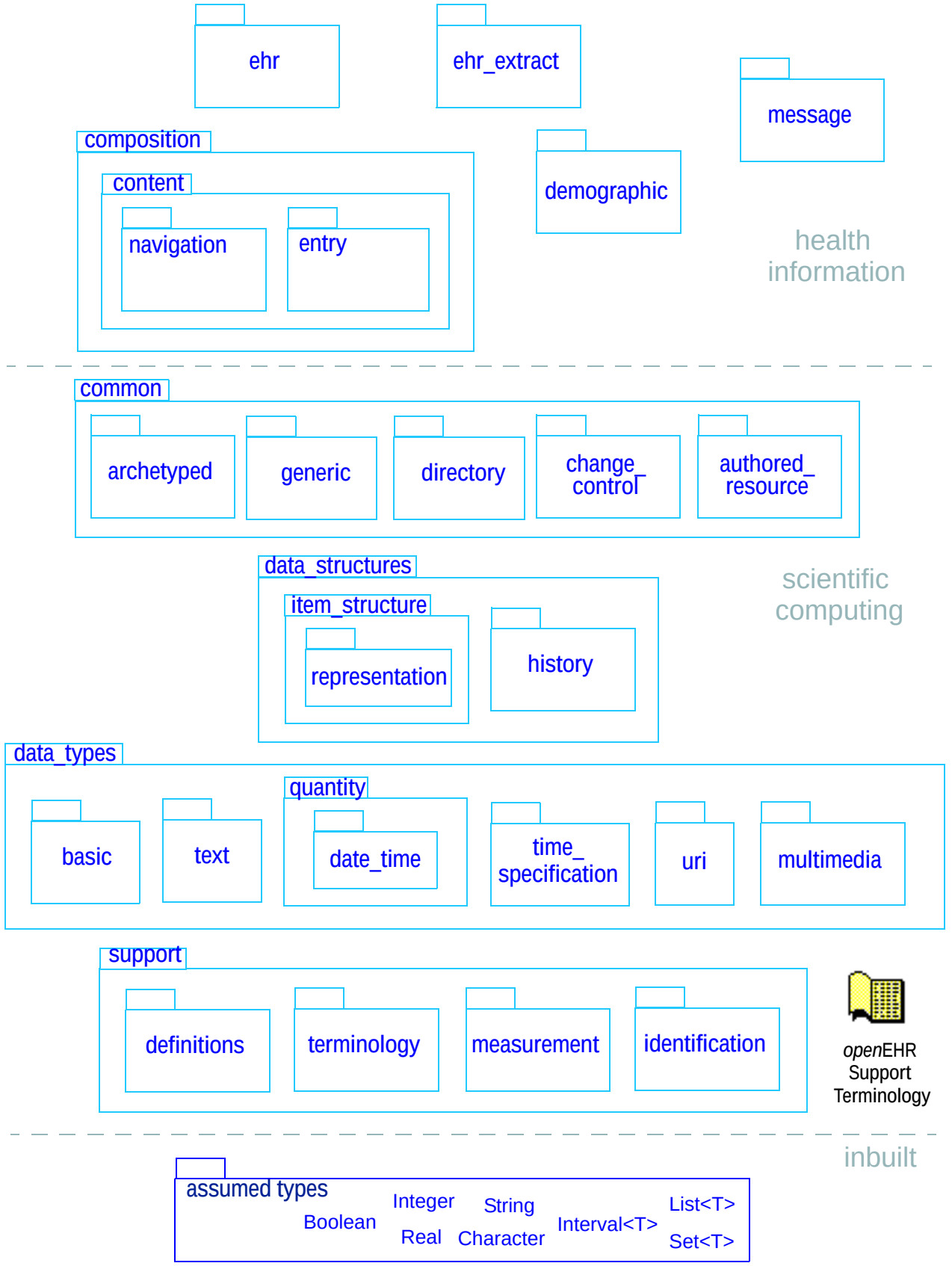

FIGURE 5 Structure of org.openehr.rm package

Data Types Information Model

A set of clearly defined data types underlies all other models, and provides a number of general and clinically specific types required for all kinds of health information. The following categories of datatypes are defined in the data types reference model.

Text: plain text, coded text, paragraphs.

Quantities: any ordered type including ordinal values (used for representing symbolic ordered values such as "+", "++", "+++"), measured quantities with values and units, and so on.

Date/times: date, time, date-time types, and partial date/time types.

Encapsulated data: multimedia.

Basic types: boolean, state variable.

Data Structures Information Model

In many reference models, generic data structures are available for expressing content whose particular structure will be defined by archetypes. The generic structures are as follows.

Single: single items, used to contain any single value, such as a height or weight.

List: linear lists of named items, such as many pathology test results.

Table: tabular data, including unlimited and limited length tables with named and ordered columns, and potentially named rows.

Tree: tree-shaped data, which may be conceptually a list of lists, or other deep structure.

History: time-series structures, where each time-point can be an entire data structure of any complexity, described by one of the above structure types. Point and interval samples are supported.

Common Information Model

Several semantic concepts are used in common by various models. The classes `LOCATABLE` and `ARCHETYPED` provide the link between information and archetype models. The classes `ATTESTATION` and `PARTICIPATION` are generic domain concepts that appear in various reference models. The last group of concepts consists of a formal model of change management which applies to any service that needs to be able to supply previous states of its information, in particular the demographic and EHR services.

EHR Information Model

The EHR IM defines the containment and context semantics of the concepts `EHR`, `COMPOSITION`, `SECTION`, and `ENTRY`. These classes are the major coarse-grained components of the EHR, and correspond directly to the classes of the same names in CEN EN13606:2005 and fairly closely to the "levels" of the same names in the HL7 Clinical Document Architecture (CDA) release 2.0.

EHR Extract

The EHR Extract IM defines how an EHR extract is built from `COMPOSITIONs`, demographic, and access control information from the EHR.

Demographics

The demographic model defines generic concepts of `PARTY`, `ROLE` and related details such as contact addresses. The archetype model defines the semantics of constraint on `PARTYs`, allowing archetypes for any type of person, organisation, role and role relationship to be described. This approach provides a flexible way of including the arbitrary demographic attributes allowed in the OMG HDTF PIDS standard.

Workflow

Workflow is the dynamic side of clinical care, and consists of models to describe the semantics of processes, such as recalls, as well as any care process resulting from execution of guidelines.

4.2 Archetype Model (AM)

The *open*EHR `am` package contains the models necessary to describe the semantics of archetypes and templates, and their use within *open*EHR. These include ADL, the Archetype Definition Language (expressed in the form of a syntax specification), the `archetype` and `template` packages, defining the object-oriented semantics of archetypes and templates, and the `openehr_profile` package, which defines a profile of the generic archetype model defined in the `archetype` package, for use in *open*EHR (and other health computing endeavours). The internal structure of the `am` package is shown in FIGURE 6.

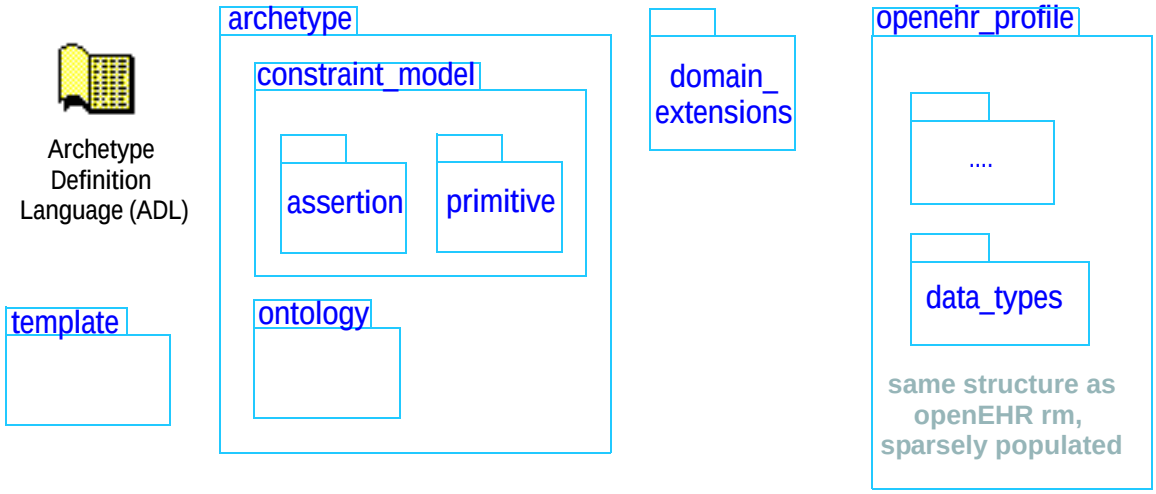

FIGURE 6 Structure of the org.openehr.am package

4.3 Service Model (SM)

The *open*EHR service model includes definitions of basic services in the health information environment, centred around the EHR. It is illustrated in FIGURE 7. The set of services actually included will undoubtedly evolve over time, so this diagram should not be seen as definitive.

EHR Client API

The EHR client API defines the fine-grained interface to EHR data, at the level of Compositions and below. It allows an application to create new information in an EHR, 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.

EHR Service Model

The EHR service model defines the coarse-grained interface to electronic health record service. The level of granularity is *open*EHR Contributions and Compositions, i.e. a version-control / change-set

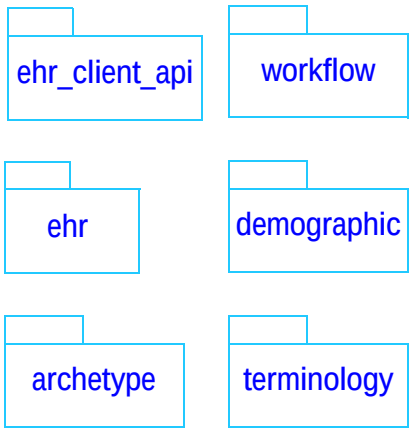

FIGURE 7 Structure of the org.openehr.sm package

interface. The finest object that can be requested or committed via the EHR service is a single Composition, or the EHR Directory structure.

Part of the model 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 ids of patients matching a particular criterion.

Archetype Service Model

The archetype service model defines the interface to online repositories of archetypes, and can be used both by GUI applications designed for human browsing as well as access by other software services such as the EHR.

Terminology Interface Model

The terminology interface service provides the means for all other services to access any terminology available in the health information environment, including basic classification vocabularies such as ICDx and ICPC, as well as more advanced ontology-based terminologies. Following the concept of division of responsibilites in a system-of-systems context, the terminology interface abstracts the different underlying architectures of each terminology, allowing other services in the environment to access terms in a standard way. The terminology service is thus the gateway to all ontology- and terminology-based knowledge services in the environment, which along with services for accessing guidelines, drug data and other “reference data” enables inferencing and decision support to be carried out in the environment.

5 Implementation Technology Specifications

5.1 Overview

ITSs are essentially created by the application of transformation rules from the "full-strength" semantics of the abstract models to equivalents in a particular technology. Transformation rules usually include mappings of:

- names of classes and attributes;
- property and function signature mapping;
- mapping of basic types e.g. strings, numerics;
- how to handle multiple inheritance;
- how to handle generic (template) types;
- how to handle covariant and contravariant redefinition semantics;
- the choice of mapping properties with signature *xxxx:T* (i.e. properties with no arguments) to stored attributes (*xxxx:T*) or functions (*xxxx():T*);
- how to express preconditions, postconditions and class invariants;
- mappings between assumed types such as `List<>`, `Set<>` and inbuilt types.

ITSs are being developed for a number of major implementation technologies, as summarised below. Implementors should always look for an ITS for the technology in question before proceeding. If none exists, it will need to be defined. A methodology to do this is being developed.

FIGURE 8 illustrates the implementation technology specification space. Each specification documents the mapping from the standard object-oriented semantics used in the *open*EHR abstract models, and also provides an expression of each of the abstract models in the ITS formalism.

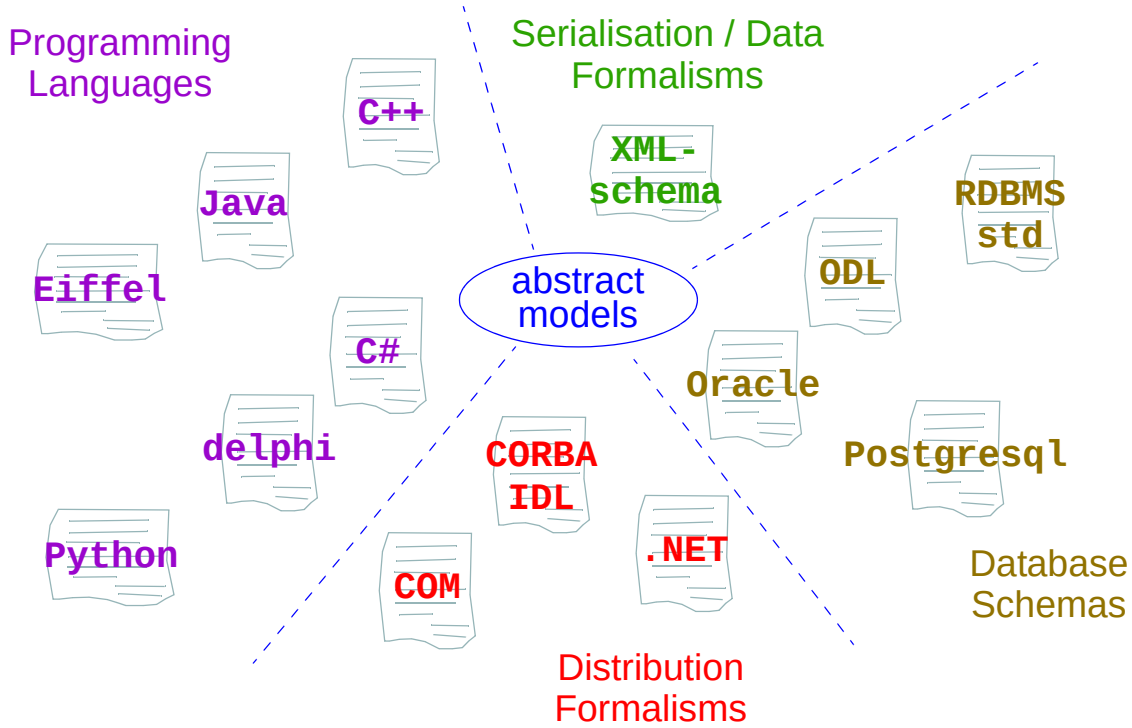

FIGURE 8 Implementation Technologies

A References

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

2 Beale T. *Archetypes: Constraint-based Domain Models for Future-proof Information Systems*. OOPSLA 2002 workshop on behavioural semantics. Available at http://www.deep-thought.com.au/XXX.

3 Beale T *et al*. *Design Principles for the EHR*. See http://www.openEHR.org/DP.html.

4 ISO:IEC: Information Technology. *Open Distributed Processing, Reference Model: Part 2:Foundations*.

5 Maier M. *Architecting Principles for Systems-of-Systems*. Technical Report, University of Alabama in Huntsville. 2000. Available at http://www.infoed.com/Open/PAPERS/systems.htm

6 Rector A L, Nowlan W A, Kay S. *Foundations for an Electronic Medical Record*. The IMIA Yearbook of Medical Informatics 1992 (Eds. van Bemmel J, McRay A). Stuttgart Schattauer 1994.

7 Schloeffel P. (Editor). Requirements for an Electronic Health Record Reference Architecture. International Standards Organisation, Australia; Feb 2002; ISO TC 215/SC N; ISO/WD 18308.

8 CORBAmed document: *Person Identification Service*. (March 1999). (Authors?)

9 CORBAmed document: *Lexicon Query Service*. (March 1999). (Authors?)

END OF DOCUMENT