



The *openEHR* Technical Roadmap

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

Revision: 1.4.2

Pages: 25

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

© 2003 The *openEHR* Foundation

The *openEHR* 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
Patrons	To Be Announced

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

Amendment Record

Issue	Details	Who	Completed
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	5
1.1	Purpose.....	5
1.2	Status.....	5
1.3	Overview.....	5
2	Guide to Readers	6
2.1	Background Documentation	6
2.2	Technical Documentation	6
3	Overview of Specifications	8
3.1	Abstract Models	8
3.2	Implementation Technology Specifications.....	8
4	Abstract Model Specifications	9
4.1	Origins	9
4.2	Model Universe Structure	9
4.2.1	Separation of Responsibilities	9
4.2.2	Viewpoints	9
4.3	Knowledge/Information Separation.....	11
4.3.1	Archetypes and Templates	11
4.4	openEHR Model Universe	11
4.4.1	Specification Structure.....	11
4.5	openEHR Package Structure.....	13
4.6	Reference Model Packages	13
4.6.1	Support Level.....	13
4.6.2	Information Level	15
4.6.3	Knowledge Level.....	16
4.7	Archetype Model Packages	17
4.7.1	Overview.....	17
4.7.2	Naming.....	17
4.7.3	Archetype Metadata	17
4.7.4	Data Types	17
4.7.5	Other Levels.....	18
4.8	Service Model Packages	18
4.9	Validation Tools	19
5	Implementation Technology Specifications	20
5.1	Overview.....	20
5.2	Programming Languages	21
5.3	Serialisation Formalisms.....	21
5.4	Distribution Formalisms	21
5.5	Database Schemas.....	21
6	Relationship to Standards	22
6.1	CEN ENV 13606 EHCRA.....	22
6.2	CEN ENV 13940 - Continuity of Care	22
6.3	CEN HISA 12xxx	22
6.4	HL7 version 3	22
6.4.1	Messaging	22

6.4.2	Clinical Document Architecture (CDA).....	22
6.5	OMG HDTF Standards.....	22
6.6	ISO TC 251 TS 18308.....	22
A	References	23

1 Introduction

1.1 Purpose

This document provides an overview of the *openEHR* specifications. The intended audience includes:

- Health data managers;
- Standards bodies producing health informatics standards;
- Software development organisations using *openEHR*;
- Academic groups using *openEHR*;
- 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 *openEHR* are described in other documents.

1.2 Status

The latest version of this document can be found in PDF and HTML formats at http://www.openEHR.org/Doc_html/Document/roadmap.htm. New versions are announced on openehr-announce@openehr.org.

1.3 Overview

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 *openEHR*;
- Rules for deriving implementation technology specifications (ITSs) are described.

2 Guide to Readers

2.1 Background Documentation

A number of key documents describe the thinking behind the *openEHR* approach. The most important of these are:

- the Requirements
- the [Design Principles](#)
- the Conformance Criteria

Anyone wishing to properly understand the motivations and abstractions behind *openEHR* should read these documents.

2.2 Technical Documentation

The *openEHR* technical specifications constitute a formal statement of semantics of EHRs and related services, and can be used for building software, schemas and other deliverables. The documentation has been created so as to enable efficient access to specifications for any particular service (e.g. EHR, demographics) independently, requiring the minimum number of other documents to be read in advance.

An order in which to approach the abstract specifications for the first time is as follows:

- **background documents:**
 - [openEHR modelling guide](#)
- **support documents:**
 - the Support Reference Model
 - the [Data Types Reference Model](#)
 - the [Data Structures Reference Model](#)
 - the [Common Reference Model](#)
- **Reference Models:**
 - the [EHR Reference Model](#)
 - the [Demographic Reference Model](#)
 - the [EHR Extract Reference Model](#)
- **Archetype Models**
 - the Support Archetype Model
 - the Data Types Archetype Model
 - the Data Structures Archetype Model
 - the Common Archetype Model
 - the EHR Archetype Model
 - the Demographic Archetype Model
- **Service Models**
 - the Data Factory Service
 - the Terminology interface
 - the EHR Service
 - the Demographic Service

Following this, implementation specifications can be used for each of the above, as relevant, e.g.:

- the XML data types specification
- the Java EHR API

3 Overview of Specifications

3.1 Abstract Models

All formal specifications published by *openEHR* 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 *openEHR* modelling guide describes the semantics of the models.

3.2 Implementation Technology Specifications

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 *openEHR* 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.

4 Abstract Model Specifications

4.1 Origins

The *openEHR* specifications and systems are founded upon a number of paradigms of computing. The first is that of distributed systems, whereby a number of services are defined in a distributed computing environment. The second is that of “two-level modelling”, in which the information and knowledge layers of systems are fully separated out, and systems are built from information models only, and process knowledge models at runtime. The knowledge models are known as *archetypes* and *templates*. The combination of distribution and two-level modelling gives rise to a model universe in which there is a family of models for each service.

The following sections describe the *openEHR* specifications in terms of these paradigms.

4.2 Model Universe Structure

4.2.1 Separation of Responsibilities

FIGURE 1 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 is it exhaustive. It simply illustrates a plausible set of distinct responsibilities in a distributed health computing environment, providing a basis for the models of *openEHR*. Of the services shown on the diagram, *openEHR* provides specifications only for the more central ones. 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 *openEHR* specifications often act as a mechanism to create coherent structural definitions in the informational and computational viewpoints that integrate existing standards.

4.2.2 Viewpoints

At the coarsest level, the *openEHR* specifications define semantics for services in an EHR-centred health computing environment. Each service provides information via an interface. This is the central principle of distributed computing, described by the ISO RM/ODP specifications [4]. One underlying concept of ISO RM/ODP is the notion of *viewpoints*, 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.

Other principles include those of *low coupling* and *separation of responsibility*, which are applied in order to make the specification and building of system tractable. Rather than building a single mono-

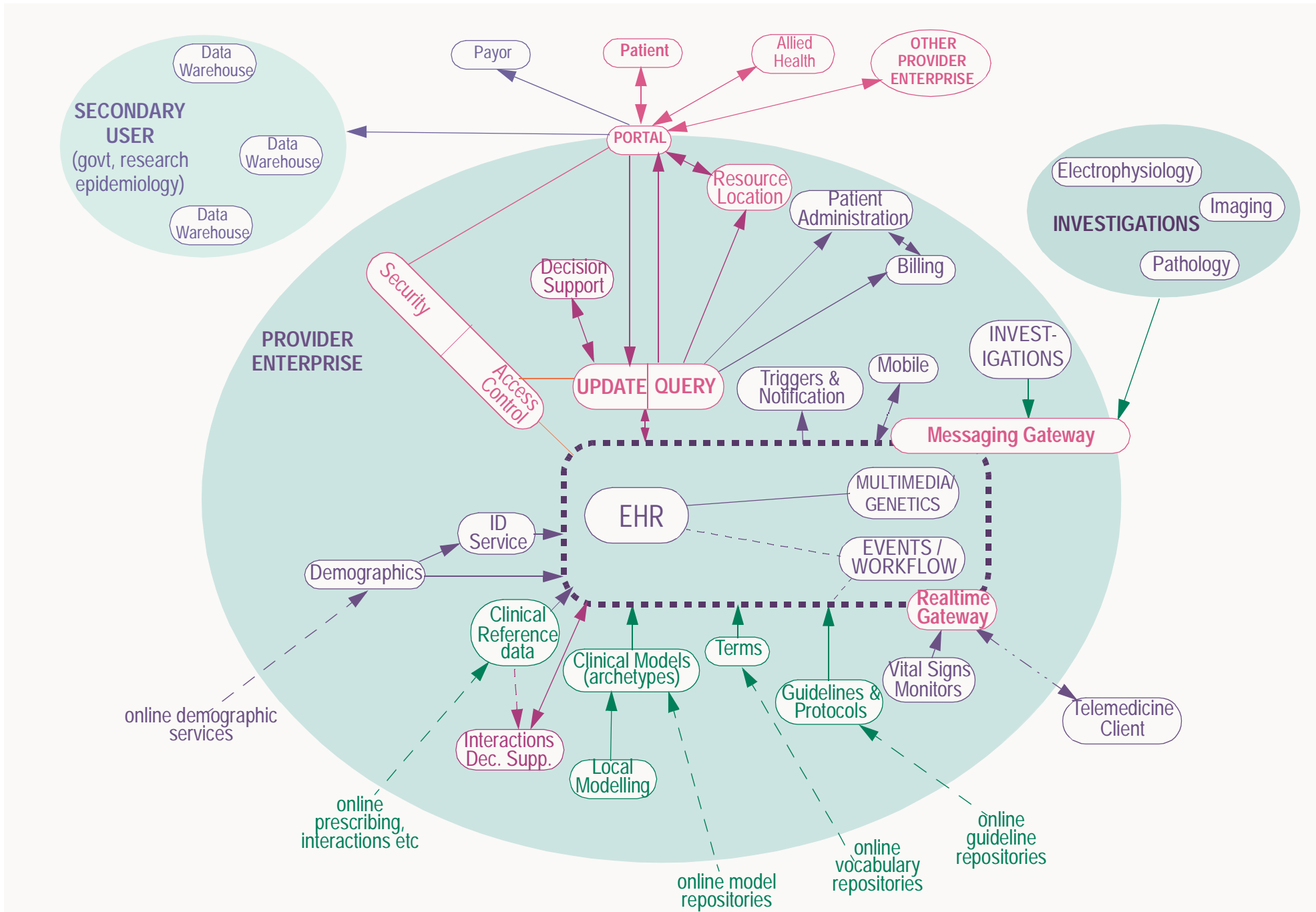


FIGURE 1 A Health Information Environment

lithic, hypercomplex system, the distributed approach defines multiple services which are relatively small, and can be specified, built and deployed independently. The use of standards ensures that such services can communicate and therefore cooperate.

4.3 Knowledge/Information Separation

4.3.1 Archetypes and Templates

The *openEHR* reference models are designed within the framework of “two-level” modelling, described in [2]. The reference 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 *archetype models*, which are formally related to the reference models. There is an archetype model for any reference model for which it is desired that archetypes be created.

Archetypes are used at runtime by building templates from them. A template is a tree of archetypes forming a composition of the coarse-grained parts of the record, e.g. Transactions, Organiser trees, Entries and so on. Thus, while there are likely to be archetypes for such things as “biochemistry results” (an Entry archetype) and “SOAP headings” (an Organiser archetype), templates are used to put archetypes together to form whole Transactions in the record, e.g. for “discharge summary”, “antenatal exam” and so on.

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 “meanings” - 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.

4.4 *openEHR* Model Universe

4.4.1 Specification Structure

As a consequence of the issues discussed above, the *openEHR* “model universe” is divided up according to areas of responsibility, and then in each area, potentially distinct models, as follows:

- Reference Model (RM): information semantics;
- Service Model (SM): service interface / API definition;
- Archetype Model (AM): a model of the archetype semantics for a given reference model.

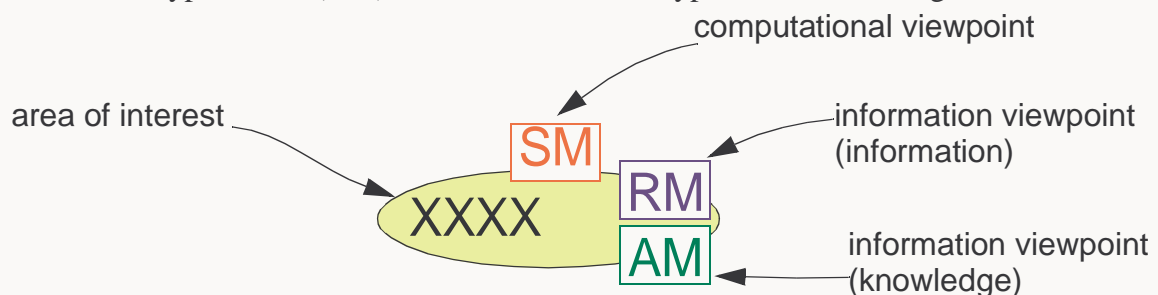


FIGURE 2 Structure of *openEHR* Specifications

The RM belongs in the information viewpoint, and its purpose is to describe the informational semantics of the entities in the area of interest - i.e. of real data. The AM also belongs in the information viewpoint, but describes the information semantics of knowledge, in this case archetypes. A service model is also sometimes defined, which describes the interface semantics of a service in the area of interest. FIGURE 2 shows these relationships.

FIGURE 3 illustrates the growing *openEHR* universe of models, divided into three groups, namely information, knowledge and support. Models in the information group relate to services providing operational information, such as the EHR and demographics. Models in the knowledge group relate to services which provide reference information, (i.e. “knowledge”) such as drug definitions, terminologies, archetypes and quantitative knowledge.

In the diagram, each area of responsibility has a reference model (denoted “RM”) and an archetype model (“AM”). Those areas which can be conceived of as online services also have service models (denoted by “SM”). Each “RM”, “AM” and “SM” box on the diagram corresponds to a distinct *openEHR* specification, having its own document and potentially numerous expressions in various formalisms.

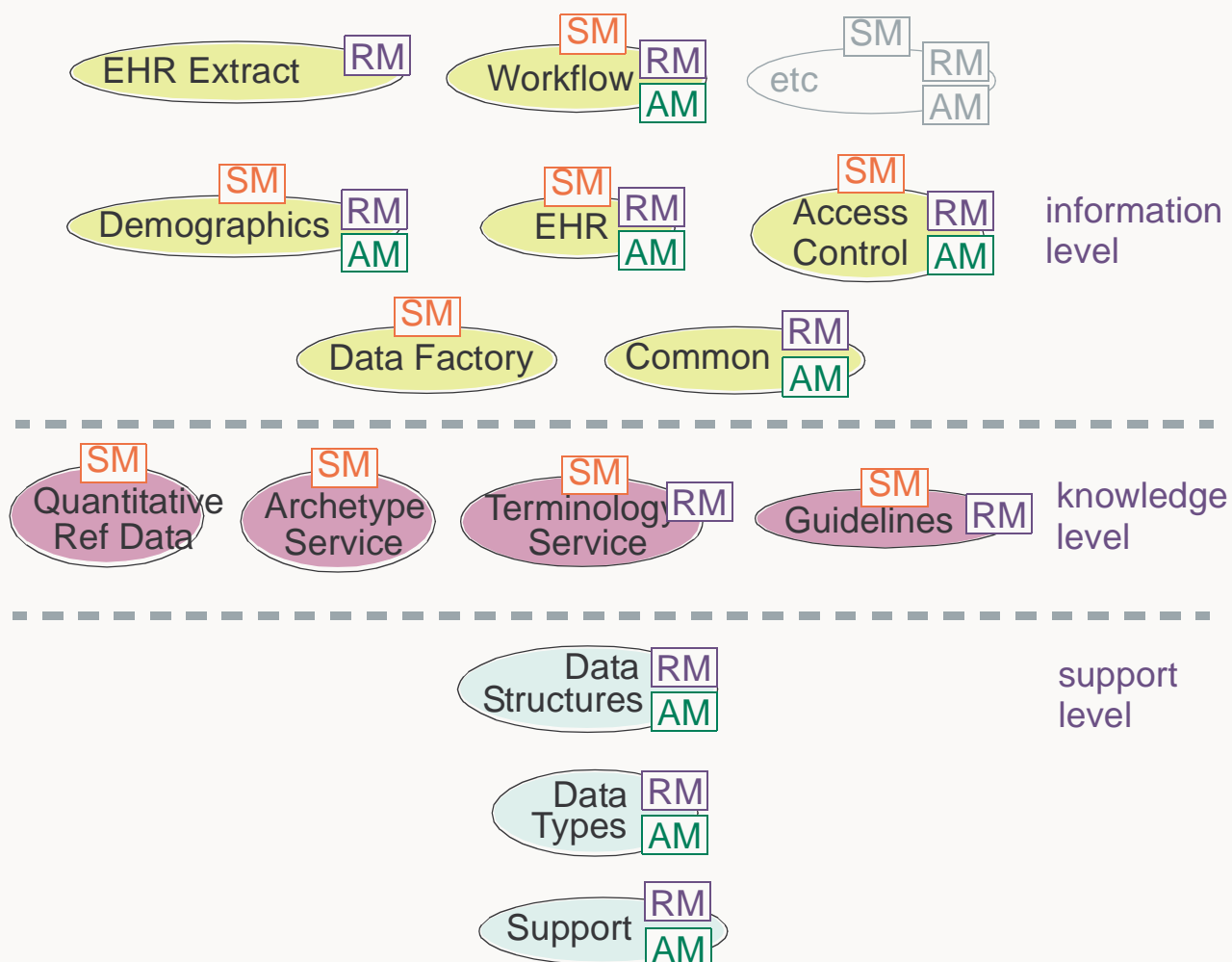


FIGURE 3 *openEHR* Model Universe

4.5 openEHR Package Structure

FIGURE 5 illustrates the overall package structure of the *openEHR* Reference Models. Three major namespaces are defined: RM, AM and SM. Packages describing detailed semantics will appear in one of these namespaces. Other top-level namespaces might be added in the future by *openEHR*; implementors will no doubt add their own.

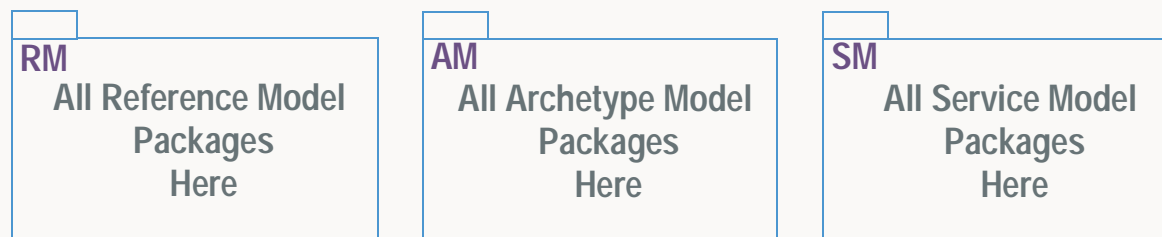


FIGURE 4 Global Package Structure of *openEHR*

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. Each outer package corresponds to one document¹. Where packages are nested, the inner packages cannot exist outside of their parent package. The archetype models have a nearly identical package structure within the AM namespace..

The package structure in FIGURE 5 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.

4.6 Reference Model Packages

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

4.6.1 Support Level

Several support models provide formal semantics used in other models, as follows.

Support

This package describes the most basic concepts, required by all other packages, and is comprised of the Definitions and External packages. The former contains symbolic constants, while the latter describes references and identifiers that enable any object in a system to refer to an object in another system (such as enabling an EHR to refer to a demographic entity in a demographic server).

Data Types

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. These 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

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

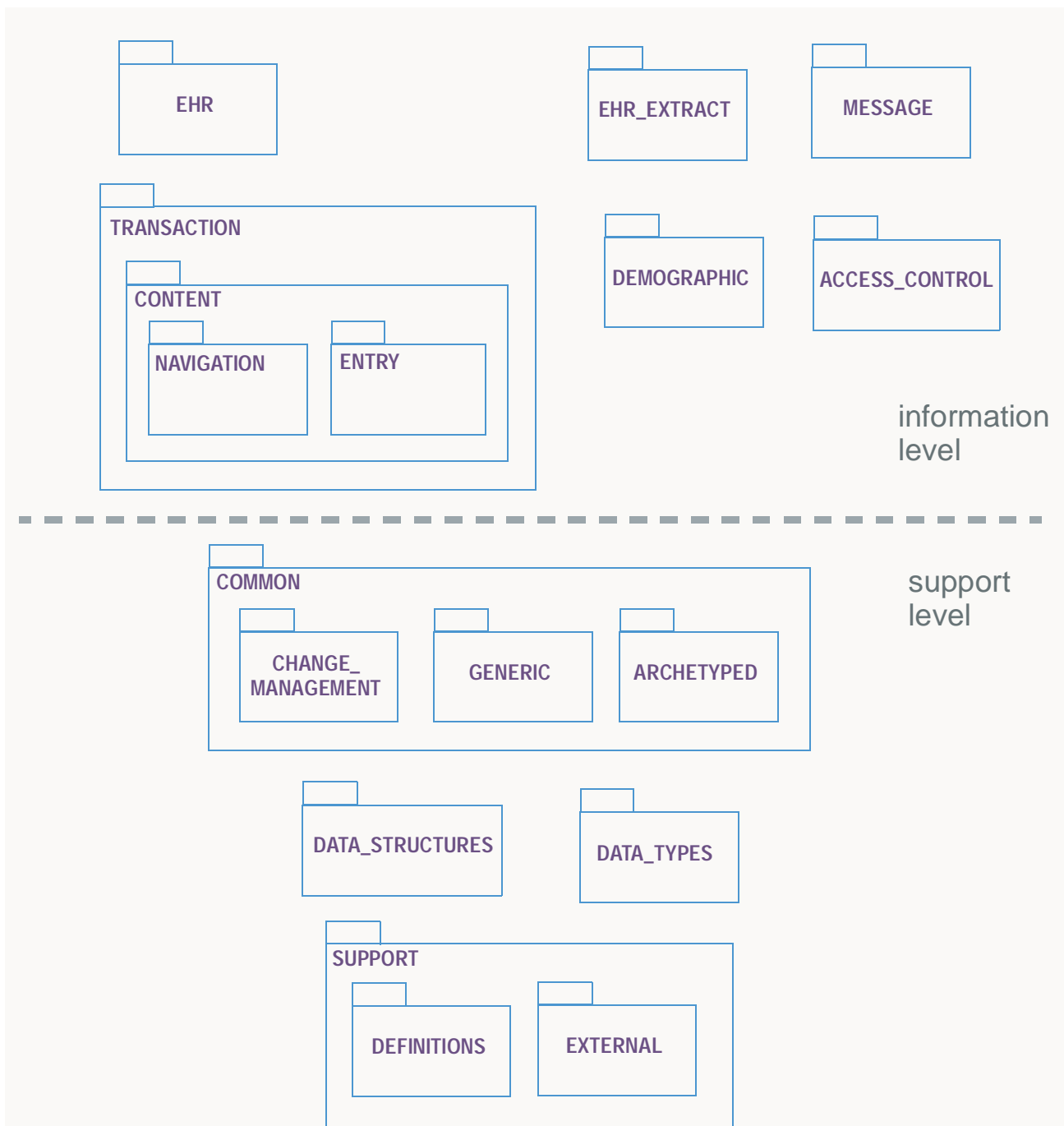


FIGURE 5 Reference Model Package Structure of openEHR

Encapsulated data: multimedia,

Basic types: boolean, state variable

The Data Types Archetype Model defines constraint semantics on objects of all the above data types.

Data Structures

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.

The Data Structures Archetype Model defines constraint semantics on these data structures, allowing archetypes to include constraint specifications defining particular clinical structures, such as “biochemistry results” and “person name”.

Common

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.

4.6.2 Information Level

The Information level includes models describing semantics of production information, i.e. health records, demographic data, access control data and so on.

EHR

The EHR Reference and Archetype models define respectively the EHR architecture and constraint models. The EHR Reference Model includes the well-known containment and context semantics of the concepts `EHR`, `TRANSACTION`, `ORGANISER`, and `ENTRY`. The associated archetype model defines the semantics of constraints on each of these types, in particular providing the model for `ORGANISER` and `ENTRY` archetypes, as well as the semantics of the composition of archetypes between the layers. The EHR service model defines the public interface of an EHR service, including support for queries to access any part of any health record, and an interface to create new EHR data.

The structure of the *openEHR* EHR reference model has a number of origins:

Ontological: the coarse structure is defined by a set of ontological layers starting with the level 0 reference ontology (expressed in terminologies). See [3].

Context Analysis: the finer structure is based on the analysis of information contexts described in [3].

Modelling Principles: the structure has also been guided by good software principles of low coupling and layers of abstraction. This has led to the generic data types and data structure layers, which support a number of other layers.

The context analysis has given rise to a model in which every context level is explicitly modelled, and there are no classes for any context layer. Table 1 summarises the relationship between ontological levels, context levels and the *openEHR* reference model root classes.

Table 1 Structure of the *openEHR* EHR Reference Model

Ontological Level	Context	Reference model	Reference model root class
0 - Principles	Values	<code>DATA_TYPES</code>	<code>DATA_VALUE</code>

Table 1 Structure of the *openEHR* EHR Reference Model

Ontological Level	Context	Reference model	Reference model root class
1 - Descriptive	Structure	DATA_STRUCTURES	STRUCTURE
	Temporal		HISTORY<T>
	Clinical Statement	EHR	ENTRY
2 - Organising	Organising		ORGANISER
3 - Thematic	Clinical session		TRANSACTION

EHR Extract

The EHR Extract model incorporates the same TRANSACTION semantics from the EHR reference model, and incorporates demographic and access control concepts as well, to provide a definition of a self-contained extract of an 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 more flexible way of including the arbitrary demographic attributes allowed in the OMG HDTF PIDS standard.

The demographic service model allows any other service to reference any required demographic entity. This will be heavily based on the OMG 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. The workflow archetype model provides the semantics for defining archetypes for particular clinical processes, such as a vaccination recall, hospital drug administration, monitoring, and patient education processes.

Access Control

Access control is defined by three models. The reference model describes the generic semantics of role-based access, consent, policies and authorisation. The archetype model describes semantics allowing particular policies and consent models to be described by actual archetypes. The service interface allows any consent or other access control entity to be defined by users, as well as access from any other service, including the EHR and demographic services.

4.6.3 Knowledge Level

Terminology Interface

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 responsibilities 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 ter-

minology-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.

Archetype Server

The archetype service defines the interface to the online repository 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.

Guidelines

The guidelines service provides access to computerised clinical guidelines.

Quantitative Reference Data

This service defines access to clinical reference data, quantitative unit systems, conversion algorithms and so on.

4.7 Archetype Model Packages

4.7.1 Overview

As described in [2], an archetype model is created for each reference model for which it is desired to create archetypes. Archetypes are both domain concept models, and structural constraint definitions, so each archetype model defines constraint semantics within a structure resembling that of the underlying reference model very closely. Each key class in a reference model has a correspondent in the archetype model. The relationship between a few semantic constructs in a reference model and its archetype model, such as relationships, are slightly different. The archetype package structure in the AM namespace is shown in FIGURE 6.

4.7.2 Naming

Archetype classes with a correspondent in the Data Types RM are named with the same name preceded by “C_” indicating “constraint for”. Class features which are direct homologues for features in the reference model classes are named similarly, with a preceding “c_”. This is done so that other features added to archetype classes are clearly distinct from those features required to express constraints. All such features carry data describing a constraint. Where archetypes are persisted, all “c_” features should be persisted.

4.7.3 Archetype Metadata

The AM.COMMON package describes archetype metadata, including purpose, description, lifecycle status, author and other details.

4.7.4 Data Types

The data types are the lowest level construct in any reference model. Accordingly, the data types archetype model defines types which are mostly custom-designed. This is not surprising; consider that types such as DV_CODED_TEXT and DV_QUANTITY: these embody quite complex semantics. In a similar way, their archetype counterparts C_DV_CODED_TEXT and C_DV_QUANTITY express complex semantics which have been discovered largely by experience.

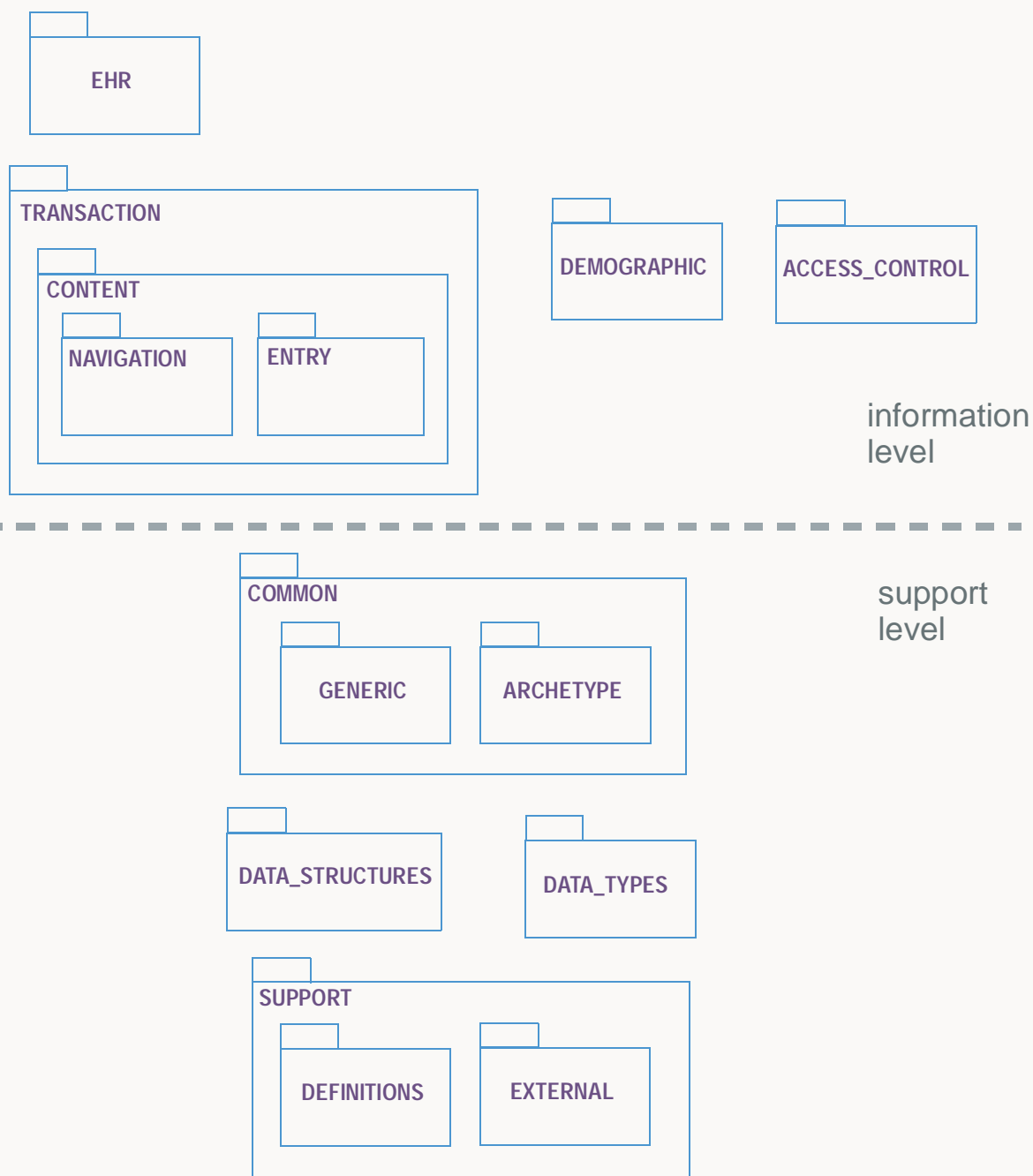


FIGURE 6 Archetype Model Package Structure of *openEHR*

4.7.5 Other Levels

This is in contrast with higher levels of constructs, such as ENTRY, ORGANISER etc in the EHR reference model, whose semantics are essentially that of containment. Consequently, the archetype classes for these constructs are largely automatically derivable from their reference model counterparts.

4.8 Service Model Packages

To Be Continued:

4.9 Validation Tools

All models are written in UML and currently validated using the Eiffel language. Eiffel is the object-oriented formalism with the closest semantics to UML/OCL (OCL = Object Constraint Language) of production languages available today, and implements multiple inheritance, generic (template) types, Design by Contract - invariants, preconditions and postconditions, and agents, and has a set of tools for not only describing models but turning models into software. It is thus able to be used for building reference specifications and implementations of all models. *openEHR* uses both ISE (<http://www.eiffel.com>) and GNU (<http://smarteiffel.loria.fr>) versions of Eiffel.

XMI files are generated from the validated Eiffel models and imported into the Rational Rose UML tool (<http://www.rational.com>); other tools are being experimented with, including Objecteering (<http://www.softteam.fr>).

The use of Eiffel as a verification tool does not imply that it should be used for any particular implementation - the *openEHR* specifications are being used with all major implementation technologies. As for any other implementation language, Eiffel has its own ITS (although it is clearly simpler than most others).

Other tools which could be used include Object-Z tools (see e.g. links from <http://www.zuser.org/>). However, as these are (unfortunately) less well-known and understood, and the more formal mathematical nature of the notation is unattractive to many developers, it was not chosen.

In the future it is expected that OCL tools will be available to fully validate models, in which case they can be used to fulfill the same function as Eiffel at the moment.

5 Implementation Technology Specifications

5.1 Overview

ITSs are effectively profiles of mapping and transformation rules from the “full-strength” semantics of the abstract models to a given technology, and may describe mappings of:

- naming 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 7 illustrates the implementation technology specification space. Each specification documents the mapping from the standard object-oriented semantics used in the *openEHR* abstract models, and also provides an expression of each of the abstract models in the ITS formalism.

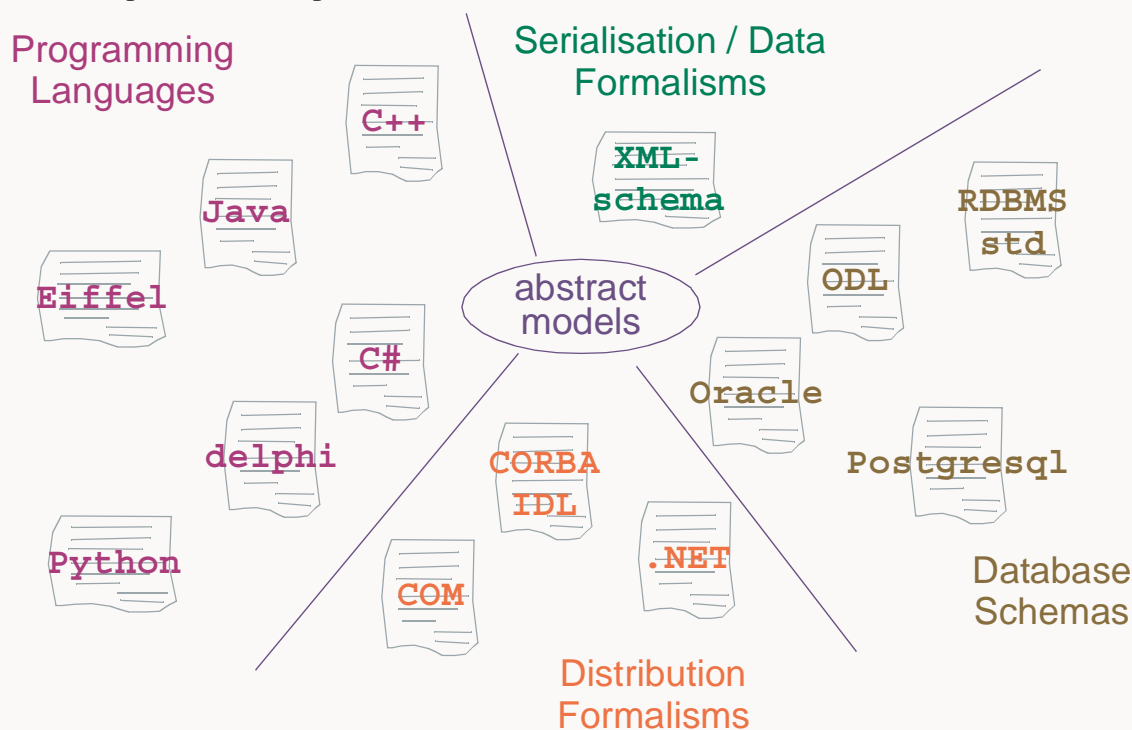


FIGURE 7 Implementation Technology Roadmap

5.2 Programming Languages

5.3 Serialisation Formalisms

5.4 Distribution Formalisms

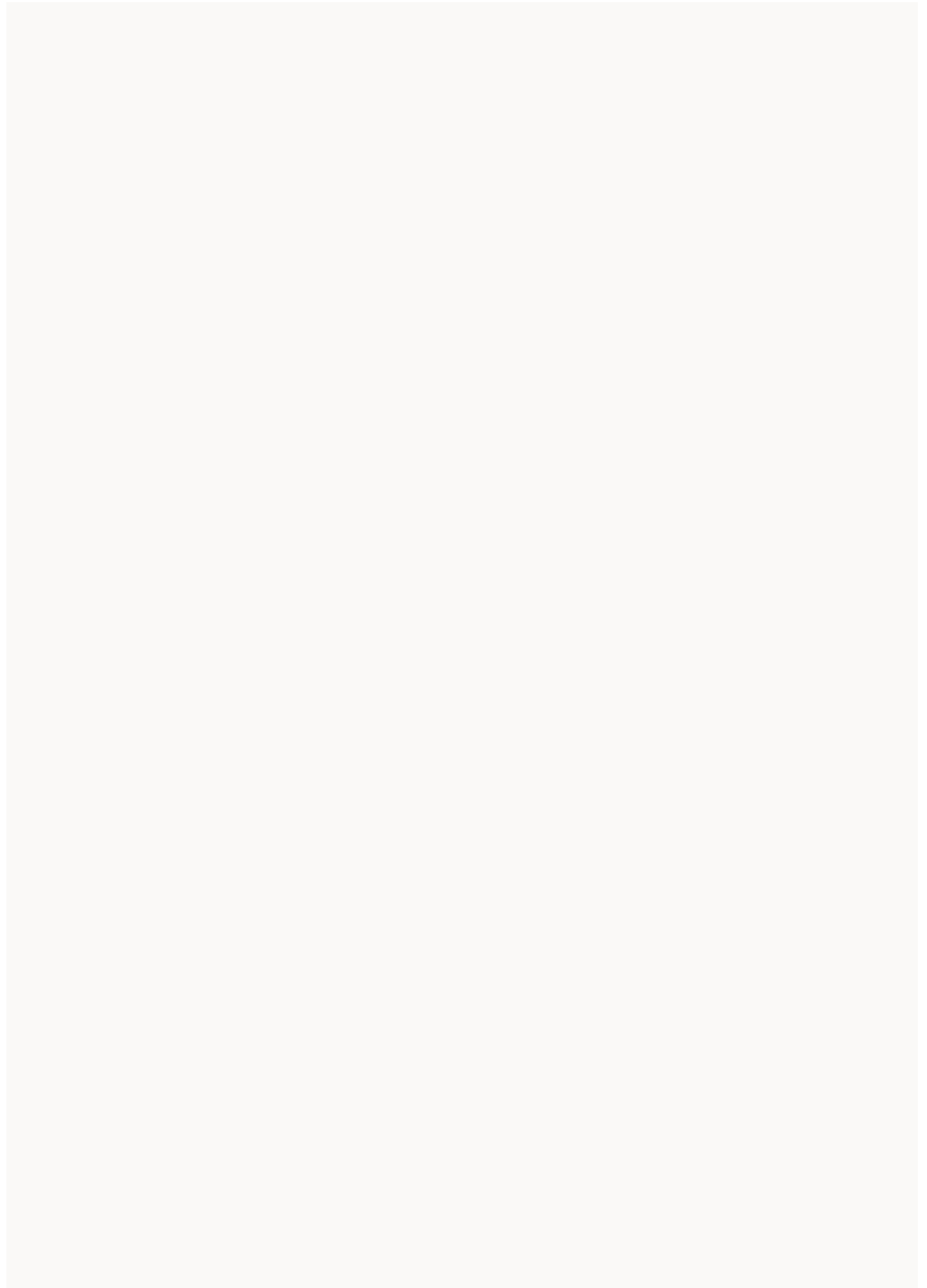
5.5 Database Schemas

6 Relationship to Standards

- 6.1 CEN ENV 13606 EHCRA**
- 6.2 CEN ENV 13940 - Continuity of Care**
- 6.3 CEN HISA 12xxx**
- 6.4 HL7 version 3**
 - 6.4.1 Messaging**
 - 6.4.2 Clinical Document Architecture (CDA)**
- 6.5 OMG HDTF Standards**
- 6.6 ISO TC 251 TS 18308**

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.deepthought.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 Bommel 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