REFERENCE MODEL

The *open*EHR Data Structures Archetype Model

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

Revision: 0.5

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
Patrons	To Be Announced

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

Copyright Notice

Amendment Record

Issue	Details	Who	Date
0.5	CR-000019. Add HISTORY & STRUCTURE supertype. CR-000028. Change name of STRUCTURE class to avoid clashes.	T Beale H Frankel	03 Mar 2004
0.4.1	CR-000013. Rename key classes according to CEN ENV13606. CR-000041. Visually differentiate primitive types in openEHR documents.	S Heard, D Kalra, T Beale, D Lloyd	10 Oct 2003
0.4	CR-000003, CR-000004 changes. Changed package naming, improved heading structures. (Formally validated).	T Beale	20 Mar 2003
0.3	Removed existential invariants. **Formally validated using ISE Eiffel 5.2.**	T Beale	25 Feb 2003
0.2	Further development.	T Beale	28 Dec 2002
0.1	Taken from Common RM.	T Beale	11 Oct 2002

Acknowledgements

The work reported in this paper has been funded in by a number of organisations, including The University College, London; The Cooperative Research Centres Program through the Department of the Prime Minister and Cabinet of the Commonwealth Government of Australia; Ocean Informatics, Australia.

Table of Contents

1 Introduction

1.1 Purpose

This document describes the formal archetype semantics for the common data structures used in *open*EHR reference models. It is based on "The *open*EHR Data Structures Reference Model" version 1.0.

The intended audience includes:

- Standards bodies producing health informatics standards
- Software development groups using *open*EHR
- Academic groups using *open*EHR
- The open source healthcare community

1.2 Related Documents

Prerequisite documents for reading this document include:

- The *open*EHR Modelling Guide
- The *open*EHR Data Structures Reference Model

1.3 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 and HTML formats at http://www.openEHR.org/doculist.htm. New versions are announced on openehr-announce@openehr.org.

1.4 Peer review

Known omissions or questions are indicated in the text with a "to be determined" paragraph, as follows:

TBD_1: (example To Be Determined paragraph)

Areas where more analysis or explanation is required are indicated with "to be continued" paragraphs like the following:

To Be Continued: more work required

Reviewers are encouraged to comment on and/or advise on these paragraphs as well as the main content. Please send requests for information to info@openEHR.org. Feedback should preferably be discussed on one of the appropriate mailing lists, openehr-technical@openehr.org or openehr-clinical@openehr.org.

2 Background

2.1 Requirements

2.2 Design Principles

3 The AM.DATA_STRUCTURE Package

3.1 Overview

The DATA_STRUCTURE archetype package contains two packages: the ITEM_STRUCTURE package and the HISTORY package. The first includes constraint classes for the Structure package classes, while the latter describes constraint classes for the History package classes. The DATA_STRUCTURE package is illustrated in FIGURE 1.

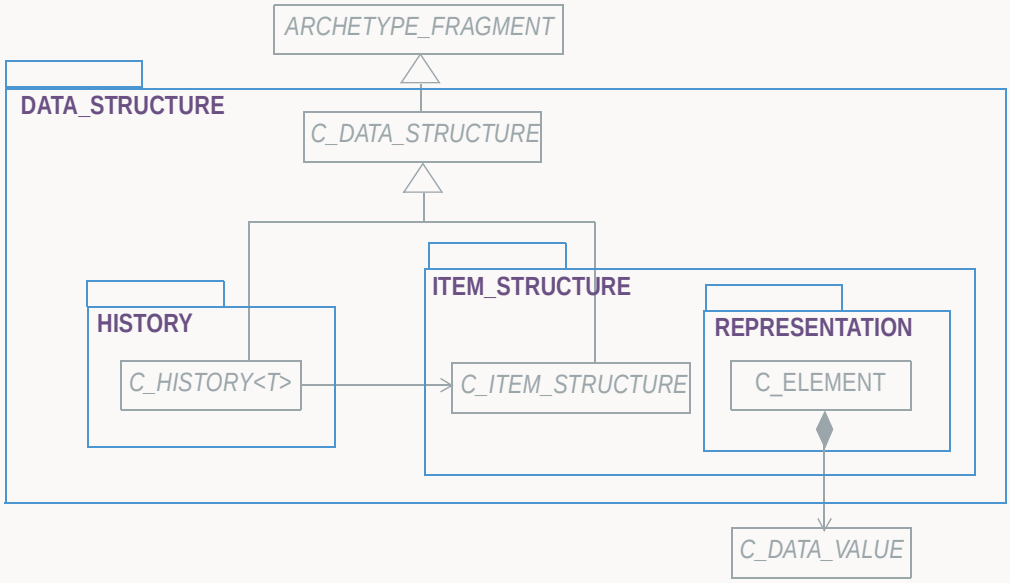

FIGURE 1 AM.DATA_STRUCTURE Package

4 AM.DATA_STRUCTURE. ITEM_STRUCTURE Package

4.1 Overview

The class model for the ITEM_STRUCTURE package is illustrated in FIGURE 2. The semantics of the classes define constraints for the corresponding reference model data structures, namely Tree, List, Table, Single and so on.

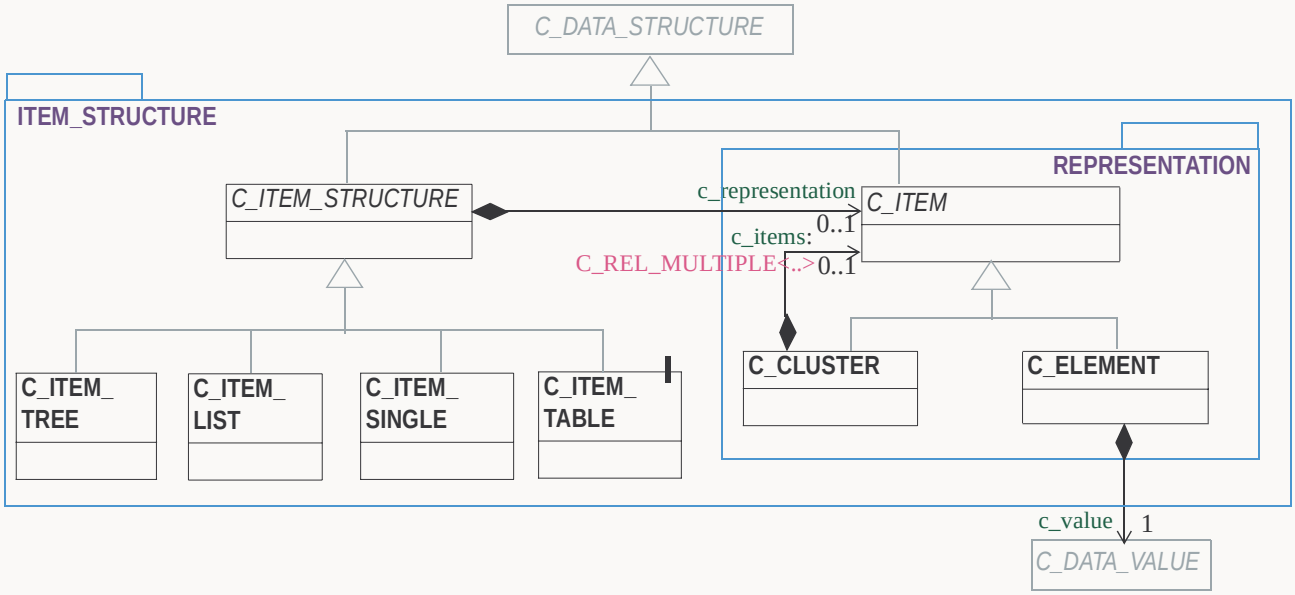

FIGURE 2 AM.DATA_STRUCTURE.ITEM_STRUCTURE Package

4.2 Class Descriptions

4.2.1 C_ITEM_STRUCTURE Class

CLASS	*C_ITEM_STRUCTURE (abstract)*	
Purpose	Abstract parent class of spatial data constraint types.	
Inherit	`C_DATA_STRUCTURE`	
Abstract	**Signature**	**Meaning**
	c_representation: `C_ITEM`	
Invariants	***c_representation_valid***: c_representation /= Void	

4.2.2 C_ITEM_SINGLE Class

CLASS	C_ITEM_SINGLE	
Purpose	Constrainer type for single values.	
Use	Used to represent any data which is logically a single value, such as a person's height or weight.	
Inherit	C_ITEM_STRUCTURE	
Attributes	**Signature**	**Meaning**
	+c_item: C_ELEMENT	

4.2.2.1 C_ITEM_SINGLE Archetype Path

In the following path structure, the *meaning* of the C_ITEM_SINGLE object acts as the root-name.

- the item: "|" <C_ITEM_SINGLE.meaning>, e.g. "|weight"

4.2.3 C_ITEM_LIST Class

CLASS	C_ITEM_LIST	
Purpose	Constrainter structure for LIST_S objects.	
Use	Used to define constraints on clinical, demographic or other List_s structures, such as a measuring protocol or a multi-part street address.	
Inherit	C_ITEM_STRUCTURE	
Attributes	**Signature**	**Meaning**
	c_representation: C_CLUSTER	
	c_item_count: INTERVAL <Integer>	Constrain the number of items in a LIST_S.
	c_items: C_LIST <C_ELEMENT>	Constrain the actual members in a list.
Functions	**Signature**	**Meaning**
	c_names: C_LIST <C_DV_TEXT>	
Invariants		

4.2.3.1 C_ITEM_LIST Paths

In the following path structure for C_lists, the *meaning* attribute of the LIST_S object acts as the root-name.

- whole list: "|" <C_ITEM_LIST.meaning>, e.g. "|BP protocol"

- nth list item: "|" <C_ITEM_LIST.meaning> "|" "item="<n>, e.g. "|BP protocol|item=2"
- named list item: "|" <C_ITEM_LIST.meaning> "|" <item.meaning>, e.g. "|BP protocol|cuff"

4.2.4 C_ITEM_TABLE Class

CLASS	C_ITEM_TABLE	
Purpose	Constrainer structure for TABLE_S objects.	
Use	Used to describe constraints for tabular data such as visual acuity result, or numerous other investigations results.	
Inherit	C_ITEM_STRUCTURE	
Attributes	**Signature**	**Meaning**
	c_representation: C_CLUSTER	
	c_row_count: INTERVAL<Integer>	Allowed number of rows
	c_column_count: INTERVAL <Integer>	Allowed number of columns
	c_row_names: C_LIST<C_DV_TEXT>	Required row names
	c_column_names: C_LIST<C_DV_TEXT>	Required column names
Invariants		

4.2.4.1 TABLE_S Paths

The following path patterns are legal for tables.

- whole table: "|<TABLE_S.name>", e.g. "|root"
- column:"|<TABLE_S.name>|<column-name>", e.g. "|vision|left eye"
- row: "|<TABLE_S.name>|<row-name>", e.g. "|vision|colour"
- cell: "|<TABLE_S.name>|<column-name>|<row-name>", e.g. "|vision|left eye|acuity w/ glasses"

4.2.4.2 C_ITEM_TABLE Example

4.2.5 C_ITEM_TREE Class

CLASS	C_ITEM_TREE
Purpose	Constrainer type for navigable tree structure.
Inherit	C_ITEM_STRUCTURE

CLASS	C_ITEM_TREE	
Attributes	**Signature**	**Meaning**
	c_representation: C_CLUSTER	
Functions	**Signature**	**Meaning**

4.2.5.1 C_ITEM_TREE Paths

Tree paths are of the following form.

- whole tree: "|<TREE_S.name>", e.g. "|biochemistry"
- node: "|<TREE_S.name>|<node-name>...<node-name>",
  e.g. "|biochemistry|lipid studies"
- leaf value: "|<TREE_S.name>|<node-name>...<node-name>|<leaf-name>",
  e.g. "|biochemistry|lipid studies|LDL cholesterol"

4.2.5.2 C_ITEM_TREE Example

5 AM.DATA_STRUCTURE.ITEM_STRUCTURE. REPRESENTATION Package

5.1 Overview

5.2 Class Descriptions

5.2.1 C_ITEM Class

CLASS	*C_ITEM (abstract)*	
Purpose	The abstract parent of `C_CLUSTER` and `C_ELEMENT` classes.	
Inherit	`ARCHETYPE_FRAGMENT`	
Attributes	**Signature**	**Meaning**

5.2.2 C_CLUSTER Class

CLASS	C_CLUSTER	
Purpose	The grouping variant of `ITEM`, which may contain further instances of `ITEM`, in an ordered list.	
Inherit	`C_ITEM`	
Attributes	**Signature**	**Meaning**
	items : `C_LIST<C_ITEM>`	Ordered list of items - `C_CLUSTER` or `C_ELEMENT` objects - under this `C_CLUSTER`.
Invariants	***c_items_valid***: items /= Void ***implies not*** items.empty	

5.2.3 C_ELEMENT Class

CLASS	C_ELEMENT	
Purpose	Constrainer class for `ELEMENT` instances.	
Inherit	`C_ITEM`	
Attributes	**Signature**	**Meaning**

CLASS	C_ELEMENT	
	c_value : C_REL_SINGLE<C_DATA_VALUE>	Allowed data values of this leaf.
Invariants	***c_value_valid***: c_value /= Void	

6 AM.DATA_STRUCTURE.HISTORY Package

6.1 Overview

The HISTORY Package classes are shown in FIGURE 3. These classes express constraints for instances of `HISTORY<T>` and its descendants.

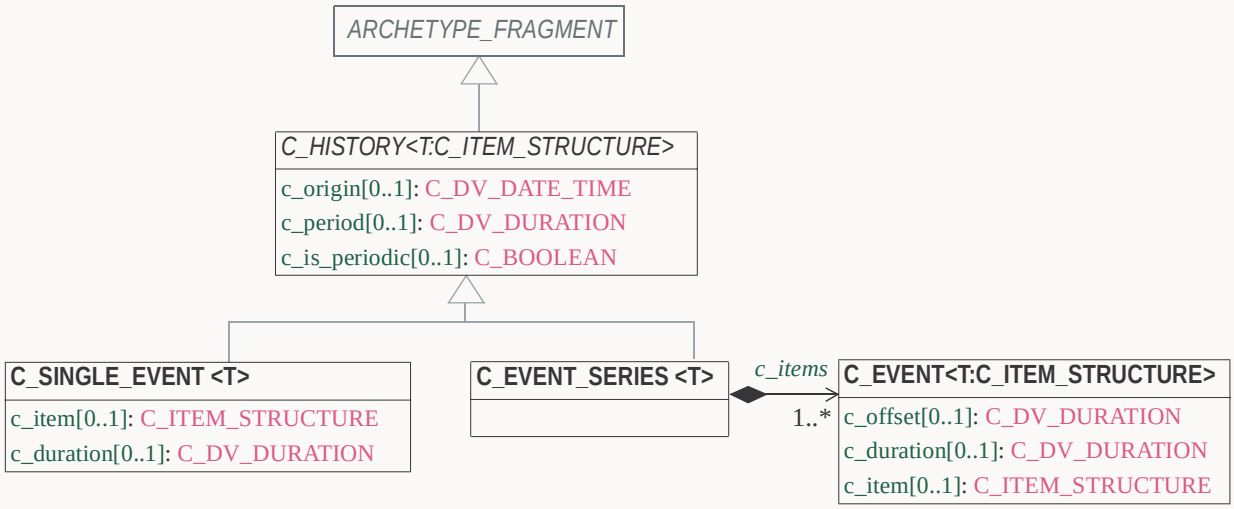

FIGURE 3 AM.DATA_STRUCTURE.HISTORY Classes

6.2 Class Descriptions

6.2.1 C_HISTORY<T> Class

CLASS	*C_HISTORY<T:C_ITEM_STRUCTURE> (abstract)*	
Purpose	Constrainer class for instances of `HISTORY`.	
Inherit	`C_ARCHETYPE_FRAGMENT`	
Attributes	**Signature**	**Meaning**
	c_origin: `C_DV_DATE_TIME`	Constraint on when origin can occur. Almost never needs to be constrained.
	c_is_periodic: `C_BOOLEAN`	Constraint on whether series is periodic or not.
	c_period: `C_DV_DURATION`	Constraint on periodicity, if c_is_periodic includes the value True.
Invariants		

6.2.2 C_EVENT_SERIES<T:C_ITEM_STRUCTURE> Class

CLASS	C_EVENT_SERIES <T:C_ITEM_STRUCTURE>	
Purpose	Constrainer type for instances of `EVENT_SERIES`.	
Inherit	`C_HISTORY<T>`	
Attributes	**Signature**	**Meaning**
	c_items: `C_LIST <C_EVENT<T>>`	List of event constraints
Invariants		

6.2.3 C_EVENT<T:C_ITEM_STRUCTURE> Class

CLASS	C_EVENT<T:C_ITEM_STRUCTURE>	
Purpose	Constrainer type for instances of `EVENT<T>`.	
Inherit	`ARCHETYPE_FRAGMENT`	
Attributes	**Signature**	**Meaning**
	c_item: `T`	Constrainer for item allowed at this event position
	c_offset: `C_DV_DURATION`	Constrainer for offset of this event
	c_duration: `C_DV_DURATION`	Constrainer for duration of this event
Invariants		

6.2.4 C_SINGLE_EVENT<T:C_ITEM_STRUCTURE> Class

CLASS	C_SINGLE_EVENT<T:C_ITEM_STRUCTURE>	
Purpose	Constrainer for instances of `SINGLE_EVENT<T>`.	
Inherit	`C_HISTORY<T>`	
Attributes	**Signature**	**Meaning**
	c_item: `T`	Constrainer for item allowed at this event position
	c_duration: `C_DV_DURATION`	Constrainer for duration of this event
Invariants		

6.2.5 History Paths

History paths include the following possibilities:

- whole history by name: "|" C_HISTORY.name, e.g. "|history"
- whole history by time: "|origin=<dt>", e.g. "|origin=2001-05-10 16:45:00"
- event: "|" C_HISTORY.meaning "|" C_EVENT.meaning, e.g. "|history|event"

A References

A.1 General

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 *et al*. *Design Principles for the EHR*. See http://www.deepthought.com.au/openEHR.

3 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.

A.2 European Projects

4 Dixon R., Grubb P.A., Lloyd D., and Kalra D. *Consolidated List of Requirements. EHCR Support Action Deliverable 1.4*. European Commission DGXIII, Brussels; May 200159pp Available from http://www.chime.ucl.ac.uk/HealthI/EHCR-SupA/del1-4v1_3.PDF.

5 Dixon R, Grubb P, Lloyd D. *EHCR Support Action Deliverable 3.5: "Final Recommendations to CEN for future work"*. Oct 2000. Available at http://www.chime.ucl.ac.uk/HealthI/EHCR-SupA/documents.htm.

6 Dixon R, Grubb P, Lloyd D. *EHCR Support Action Deliverable 2.4 "Guidelines on Interpretation and implementation of CEN EHCRA"*. Oct 2000. Available at http://www.chime.ucl.ac.uk/HealthI/EHCR-SupA/documents.htm.

7 Ingram D. *The Good European Health Record Project*. Laires, Laderia Christensen, Eds. Health in the New Communications Age. Amsterdam: IOS Press; 1995; pp. 66-74.

8 *Deliverable 19,20,24: GEHR Architecture*. GEHR Project 30/6/1995

A.3 CEN

9 ENV 13606-1 - *Electronic healthcare record communication - Part 1: Extended architecture*. CEN/ TC 251 Health Informatics Technical Committee.

10 ENV 13606-4 - *Electronic Healthcare Record Communication standard Part 4: Messages for the exchange of information*. CEN/ TC 251 Health Informatics Technical Committee.

A.4 OMG

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

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

A.5 Software Engineering

13 Meyer B. *Object-oriented Software Construction*, 2nd Ed.
Prentice Hall 1997

14 Fowler M. *Analysis Patterns: Reusable Object Models*. Addison Wesley 1997

15 Fowler M, Scott K. *UML Distilled (2nd Ed.).* Addison Wesley Longman 2000

A.6 Resources

16 IANA - http://www.iana.org/.

END OF DOCUMENT