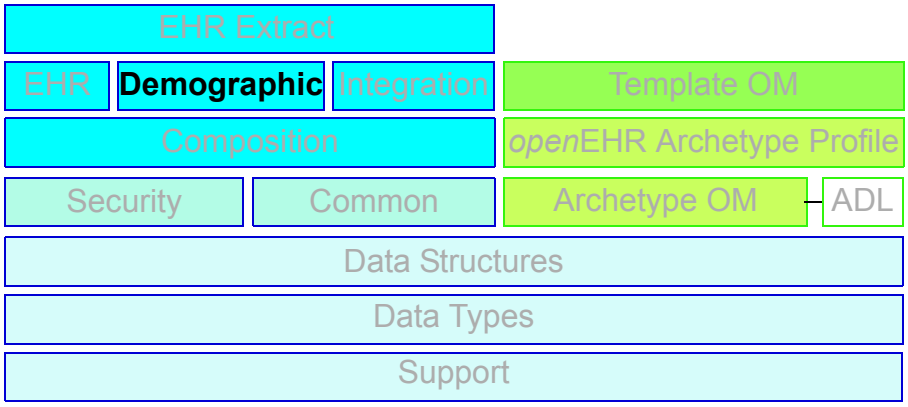
openEHR

The *open*EHR Reference Model

Demographic Information Model

Issuer: openEHR Specification Program		
Revision: 2.0.2	*Pages:* 25	*Date of issue:* 04 Aug 2008

Keywords: EHR, reference model, demographics, openehr



Amendment Record

Issue	Details	Raiser	Completed
RELEASE 1.0.2			
2.0.2	**SPEC-257**: Correct minor typos and clarify text. Make PARTY_IDENTITY.*details* and ADDRESS.*details* mandatory in class definitions.	C Ma, R Chen, T Cook	04 Aug 2008
RELEASE 1.0.1			
2.0.1	**CR-000200**. Correct Release 1.0 typographical errors.	D Lloyd	23 Feb 2006
RELEASE 1.0			
2.0	**CR-000189**. Add LOCATABLE.*parent*. New invariant in PARTY. **CR-000190**. Rename VERSION_REPOSITORY to VERSIONED_OBJECT. **CR-000194**: Correct anomalies with LOCATABLE.*uid* **CR-000161**. Support distributed versioning.	S Heard T Beale H Frankel T Beale T Beale H Frankel	25 Jan 2006
RELEASE 0.96			
RELEASE 0.95			
1.4.7	**CR-000133**. Remove details /= Void invariant from PARTY	R Chen	12 Mar 2005
1.4.6	**CR-000048**. Pre-release review of documents. Corrected STRUCTURE to be ITEM_STRUCTURE. Make ACTOR.*languages* a List not a Set.	D Lloyd	22 Feb 2005
1.4.5	**CR-000024**. Revert *meaning* to STRING and rename as *archetype_node_id*. **CR-000118**. Make package names lower case.	S Heard, T Beale T Beale	10 Jan 2005
RELEASE 0.9			
1.4.4	**CR-000041**. Visually differentiate primitive types in *open*EHR documents.	D Lloyd	04 Oct 2003
1.4.3	**CR-000013**. Rename key classes, according to CEN ENV 13606.	S Heard, D Kalra, T Beale	15 Sep 2003
1.4.2	**CR-000035**. Clarify circular relationships between PARTY and PARTY_REL.	Z Tun	14 Aug 2003
1.4.1	**CR-000003**. Removed ARCHETYPED and VERSIONABLE classes.	T Beale, Z Tun	18 Mar 2003
1.4	**Formally validated using ISE Eiffel 5.2**. Minor corrections to invariants.	T Beale	25 Feb 2003
1.3.1	**Review by H Frankel, MCA.** Corrections to diagrams and class texts. Improved PARTY_RELATIONSHIP semantics. Added Patient instance example. Made *time_validity* attributes optional.	T Beale	13 Feb 2003

Issue	Details	Raiser	Completed
1.3	Corrections to diagrams and class texts. Inheritance changed to ARCHETYPED for most key classes. Some instance examples added.	Z Tun, T Beale	08 Jan 2003
1.2	General modifications, addition of CAPABILITY class.	T Beale, D Lloyd	22 Oct 2002
1.1	Renamed CONTACT_DESCRIPTOR to CONTACT. Removed CONTACT.*role*. Renamed PARTY_ROLE to ROLE. Changed CONTACT.*address* to *addresses*. Renamed SPATIAL to STRUCTURE. Introduced PARTY and ACTOR classes.	T Beale	18 Sep 2002
1.0	Created from EHR RM.	T Beale	28 Aug 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 architecture of the *open*EHR Demographic Information Model. The semantics are drawn from previous work in GEHR, existing models in CEN 13606 and the HL7v3 RIM, and other work done in Australia.

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;
- Medical informaticians and clinicians intersted in health information.

1.2 Related Documents

Prerequisite documents for reading this document include:

- The *open*EHR Architecture Overview
- The *open*EHR Support Information Model
- The *open*EHR Data Types Information Model
- The *open*EHR Common Information Model

Other documents describing related models, include:

- The *open*EHR EHR Information Model

1.3 Status

The status of this specification is 'stable'. It is available at http://www.openehr.org/releases/trunk/architecture/rm/demographic_im.pdf.

The latest version of this document can be found at https://github.com/openEHR/specifications-RM/blob/master/publishing/demographic_im.pdf.

New versions are announced on the openehr-announce mailing list.

1.4 Peer review

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. Feedback should be provided either on the technical mailing list, or on the specifications issue tracker.

1.5 Conformance

Conformance of a data or software artifact to an *open*EHR Reference Model specification is determined by a formal test of that artifact against the relevant *open*EHR Implementation Technology Specification(s) (ITSs), such as an IDL interface or an XML-schema. Since ITSs are formal, automated derivations from the Reference Model, ITS conformance indicates RM conformance.

2 Demographic Package

2.1 Overview

The demographic model illustrated in FIGURE 1 is a generalised model of the facts one might expect to see in a demographic server. The purpose of the model is as a specification of a demographic service, either standalone, or a "wrapper" service for an existing system such as a patient master index (PMI). In the latter situation, it is used to add the required *open*EHR semantics, particularly versioning, to an existing service.

The general design is based on the scheme of party and accountability described by Fowler [19], and is influenced by clinical adaptations including work done in Australia [11] and the HL7v3 RIM [15] (itself influenced by the Fowler models).

One of the main design criteria of the model is that it expresses attributes and relationships of demographic entities which exist *regardless* of particular clinical involvements or participations in particular events. Participations are meaningful only within the context of the health record or other relevant model where they record context-specific relationships between demographic entities and events in the real world.

Another criterion is that instances of the classes in the model must be serialisable into an EHR Extract in an unambgiuous way. This requires that each PARTY be a self-contained hierarchy of data, in the same way as distinct COMPOSITIONs in the EHR model are distinct hierarchies in an Extract. In order to ensure this condition, PARTY_RELATIONSHIPs must be implemented correctly, so as to prevent endless traversal of all PARTY objects through their relationships, when serialising. See Party Relationships below for details.

2.1.1 Archetyping

The model is designed to be used with archetypes, hence the generic nature of all entities. Every class containing an attribute of the form *details*:STRUCTURE is a completely archetypable structure. As a result, archetypes can be defined for concepts such as particular kinds of PERSON, ORGANISATION; for actual ROLEs such as "health care practitioner", and for party identities and addresses.

2.1.2 Names and Addresses

Classes have been included for PARTY_IDENTITY and ADDRESS, even though they contain only a link to details, in the form of the generic STRUCTURE class. This is not strictly necessary - it could have been done simply using appropriately named attributes in the classes PARTY and CONTACT - but is done to provide a place to add specific semantics in future releases of the model. It is also expected to make software development easier, since it provides explicit classes to which behaviour and other implementation attributes can be added. Lastly, it allows the notions of PARTY_IDENTITY and ADDRESS to be explicitly used in archetype-authoring tools.

Instances of PARTY_IDENTITY, linked to PARTY by the attribute *identities* are intended to express the names of people, organisations, and other actors - that is names which are "owned" by the party, e.g. self-declared (in the case of institutions and companies) or by virtue of social relations (names given by parents, tribes etc). Identifiers of Parties given by other organisations, or the state are not represented in this way, and should be recorded in the PARTY.*details* structure instead (see below).

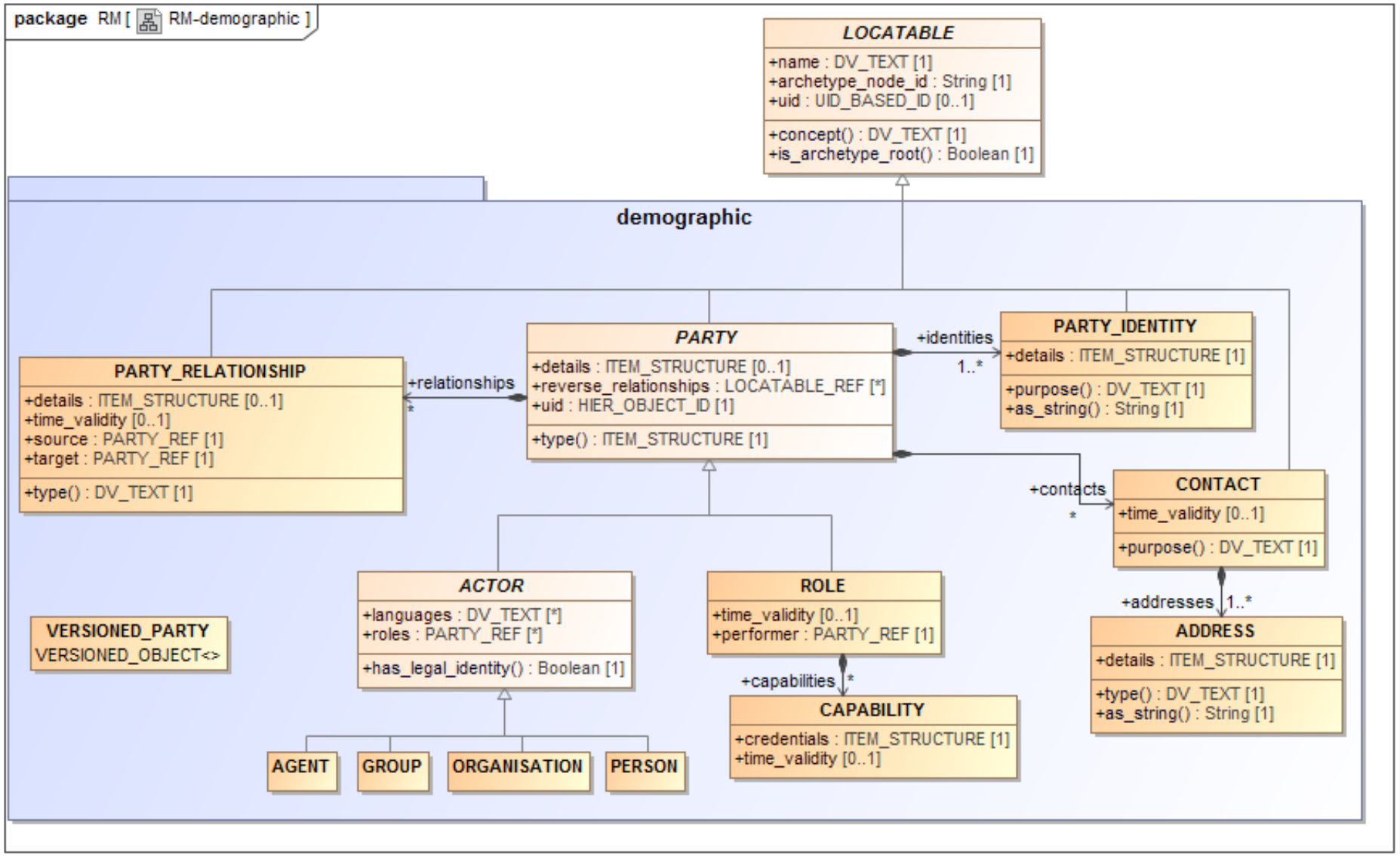


FIGURE 1 rm.demographic Package

2.1.3 Party Identification

Identifiers of Parties given by organisations or the state are treated as any other attribute of a Party, i.e. recorded as part of the data in the PARTY.*details* structure. Identifiers of Party instances in the system are provided in the same way as identifiers of Compositions in the EHR: via the uid attribute (type OBJECT_VERSION_ID) of the containing VERSION. These identifiers are used in all cross-references between Parties, and between Parties and Perty_relationships.

2.1.4 Party Relationships

Relationships between parties in the real world may be expressed using PARTY_RELATIONSHIP objects, as illustrated in FIGURE 2.

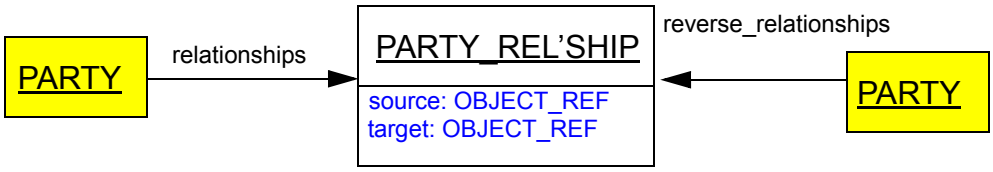


FIGURE 2 General Relationship Model

Relationships are considered *directional*, hence the use of the attribute names *source* and *target*, however, these names are otherwise neutral, and give no indication as to the meaning of the relationships, such as which party is responsible and which accountable (for comparison, see the demographic models of Fowler [19]). Accordingly, each Party involved in a relationship includes it in its *relationships* list, if it is at the source end, or in the *reverse_relationships* list, if at the target end.

The usual way to determine the directionality of a relationship between two Parties is usually by which Party's actions caused the relationship to come into being. For example, a relationship representing the concept "patient", between a health consumer and a health care organisation would have the consumer as source and the organisation as target.

PARTY_RELATIONSHIPs are stored as part of the data of the PARTY designated as the source. This means that the *relationships* attribute is by value, while *reverse_relationships* is by references, as are PARTY_RELATIONSHIP.*source* and *target*. The actual kind of reference is via the use of OBJECT_REFs containing HIER_OBJECT_IDs to denote the Version container of a Party, rather than OBJECT_VERSION_IDs, which would denote particular versions. Logically this implements the semantic that such relationships in the real world are between *continuants*, i.e. the real Parties, not just one of their version instances in a demographic system.

2.1.5 Versioning Semantics

The class PARTY and its descendants ACTOR and ROLE are all potentially versioned in a demographic system. A Version of a PARTY includes all the compositional parts, such as identities, contacts, Party relatoinships of which it is the source. Every Party is stored in its own Version container.

2.2 Class Definitions

2.2.1 PARTY Class

CLASS	*PARTY (abstract)*	
Purpose	Ancestor of all party types, including real world entities and their roles. A party is any entity which can participate in an activity. The *name* attribute inherited from `LOCATABLE` is used to indicate the actual type of party (note that the actual names, i.e. identities of parties are indicated in the *identities* attribute, not the *name* attribute).	
Inherit	`LOCATABLE`	
Attributes	**Signature**	**Meaning**
1..1 (redefined)	**uid**: `HIER_OBJECT_ID`	Identifier of this Party.
1..1	**identities**: `Set<PARTY_IDENTITY>`	Identities used by the party to identify itself, such as legal name, stage names, aliases, nicknames and so on.
0..1	**contacts**: `Set<CONTACT>`	Contacts for this party.
0..1	**relationships**: `Set<PARTY_RELATIONSHIP>`	Relationships in which this role takes part as source.
0..1	**reverse_relationships**: `Set<LOCATABLE_REF>`	Relationships in which this role takes part as target.
0..1	**details**: `ITEM_STRUCTURE`	All other details for this party.
Functions	**Signature**	**Meaning**
	type: `DV_TEXT`	Type of party, such as “PERSON”, “ORGANISATION”, etc. Role name, e.g. “general practitioner”, “nurse”, “private citizen”. Taken from inherited *name* attribute.
Invariants	***Uid_valid***: uid /= Void ***Type_valid***: type = name ***Identities_valid***: identities /= Void **and then not** identities.is_empty ***Contacts_valid***: contacts /= Void **implies not** contacts.is_empty ***Relationships_validity***: relationships /= Void **implies** (**not** relationships.is_empty ***and then*** relationships.for_all(source = Current) ***Reverse_relationships_validity***: reverse_relationships /= Void **implies** (**not** reverse_relationships.empty ***and then*** reverse_relationships.for_all(item | repository(“demographics”).all_party_relationships.has_object(item) **and then** repository(“demographics”).all_party_relationships.object(item).target = Current)) ***Is_archetype_root***: is_archetype_root ***No_parent***: parent = Void	

2.2.2 PARTY_IDENTITY Class

CLASS	PARTY_IDENTITY	
Purpose	An identity "owned" by a PARTY, such as a person name or company name, and which is used by the party to identify itself. Actual structure is archetyped.	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
1..1	**details**: ITEM_STRUCTURE	The value of the indentity. This will often taken the form of a parsable string or a small structure of strings.
Functions	**Signature**	**Meaning**
1..1	**purpose**: DV_TEXT	Purpose of identity, e.g. "legal", "stagename", "nickname", "tribal name", "trading name". Taken from value of inherited *name* attribute.
	as_string: String	Identity in the form of a single string.
Invariants	***Purpose_valid***: purpose = name ***Details_exists***: details /= Void	

2.2.3 CONTACT Class

CLASS	CONTACT	
Purpose	Description of a means of contact of a party. Actual structure is archetyped.	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
0..1	**time_validity**: DV_INTERVAL<DV_DATE>	Valid time interval for this contact descriptor.
1..1	**addresses**: List<ADDRESS>	A set of address alternatives for this purpose and time validity.
Functions	**Signature**	**Meaning**
1..1	**purpose**: DV_TEXT	Purpose for which this contact is used, e.g. "mail", "daytime phone", etc. Taken from value of inherited *name* attribute.
Invariants	***Purpose_valid***: purpose = name ***Addresses_exists***: addresses /= Void ***and then not*** addresses.empty	

2.2.4 ADDRESS Class

CLASS	ADDRESS	
Purpose	Address of contact, which may be electronic or geographic.	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
1..1	**details**: ITEM_STRUCTURE	The details of the address, in the form of a ITEM_STRUCTURE. This may take the form of a ITEM_SINGLE, whose data item is a parsable string or a list or tree of many parts.
Functions	**Signature**	**Meaning**
1..1	**type**: DV_TEXT	Type of address, e.g. “electronic”, “locality”. Taken from value of inherited *name* attribute.
	as_string: String	Address in the form of a single string.
Invariants	***Type_valid***: type = name ***Details_exists***: details /= Void	

2.2.5 ACTOR Class

CLASS	*ACTOR (abstract)*	
Purpose	Ancestor of all real-world types, including people and organisations. An actor is any real-world entity capable of taking on a role.	
Inherit	PARTY	
Attributes	**Signature**	**Meaning**
0..1	**roles**: Set<PARTY_REF>	Identifiers of the Version container for each Role played by this party.
0..1	**languages**: List<DV_TEXT>	Languages which can be used to communicate with this actor, in preferred order of use (if known, else order irrelevant).
Functions	**Signature**	**Meaning**
1..1	**has_legal_identity**: Boolean	True if one there is an identity with purpose “legal identity”
Invariants	***Roles_valid***: roles /= Void ***implies not*** roles.is_empty ***Languages_valid***: languages /= Void ***implies not*** languages.is_empty ***Legal_identity_exists***: has_legal_identity	

2.2.6 PERSON Class

CLASS	PERSON	
Purpose	Generic description of persons. Provides a dedicated type to which Person archetypes can be targeted.	
Inherit	ACTOR	
Attributes	**Signature**	**Meaning**
Invariants		

2.2.7 ORGANISATION Class

CLASS	ORGANISATION	
Purpose	Generic description of organisations. An organisation is a legally constituted body whose existence (in general) outlives the existence of parties considered to be part of it.	
Inherit	ACTOR	
Attributes	**Signature**	**Meaning**
Invariants		

2.2.8 GROUP Class

CLASS	GROUP	
Purpose	A group is a real world group of parties which is created by another party, usually an organisation, for some specific purpose. A typical clinical example is that of the specialist care team, e.g. “cardiology team”. The members of the group usually work together.	
Inherit	ACTOR	
Attributes	**Signature**	**Meaning**
Invariants		

2.2.9 AGENT Class

CLASS	AGENT	
Purpose	Generic concept of any kind of agent, including devices, software systems, but not humans or organisations.	
Inherit	ACTOR	
Attributes	**Signature**	**Meaning**
Invariants		

2.2.10 ROLE Class

CLASS	ROLE	
Purpose	Generic description of a role performed by an actor. The role corresponds to a competency of the party. Roles are used to define the responsibilities undertaken by a party for a purpose. Roles should have credentials qualifying the performer to perform the role.	
Use	Roles correspond to concepts like “general practitioner”, “nurse” and so on.	
Inherit	PARTY	
Attributes	**Signature**	**Meaning**
0..1	**capabilities**: List <CAPABILITY>	The capabilities of this role.
0..1	**time_validity**: DV_INTERVAL <DV_DATE>	Valid time interval for this role.
1..1	**performer**: PARTY_REF	Reference to Version container of Actor playing the role.
Invariants	***Capabilities_valid***: capabilities /= Void ***implies not*** capabilities.empty ***Performer_exists***: performer /= Void	

2.2.11 CAPABILITY Class

CLASS	CAPABILITY
Purpose	Capability of a role, such as “ehr modifier”, “health care provider”. Capability should be backed up by credentials.
Use	

CLASS	CAPABILITY	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
1..1	**credentials**: ITEM_STRUCTURE	The qualifications of the performer of the role for this capability. This might include professional qualifications and official identifications such as provider numbers etc.
0..1	**time_validity**: DV_INTERVAL <DV_DATE>	Valid time interval for the credentials of this capability.
Invariants	***Credentials_exists***: credentials /= Void	

2.2.12 PARTY_RELATIONSHIP Class

CLASS	PARTY_RELATIONSHIP	
Purpose	Generic description of a relationship between parties.	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
1..1 (redefined)	**uid**: HIER_OBJECT_ID	Identifier of this Party.
0..1	**details**: ITEM_STRUCTURE	The detailed description of the relationship
0..1	**time_validity**: DV_INTERVAL <DV_DATE>	Valid time interval for this relationship.
1..1	**source**: PARTY_REF	Source of relationship.
1..1	**target**: PARTY_REF	Target of relationship.
Functions	**Signature**	**Meaning**
1..1	**type**: DV_TEXT	Type of relationship, such as “employment”, “authority”, “health provision”
Invariants	***Uid_valid***: uid /= Void ***Type_validity***: type = name ***Source_valid***: source /= Void **and then** source.relationships.has(Current) ***Target_valid***: target /= Void **and then not** target.reverse_relationships.has(Current)	

2.3 Instance Examples

In the following instance examples, the values of the attributes *uid*, *source*, *target*, and *reverse_relationships* are not meant to be taken as literally valid OBJECT_IDs - for the purposes of clarity, simple integers have been used.

2.3.1 Parties

2.3.1.1 Person

FIGURE 3 illustrates a possible set of instances for a PERSON, with home and work contact information. There are separate archetypes for the PERSON, each ADDRESS, and each PARTY_IDENTITY. In the following figure, "meaning" is the meaning from the value of the *archetype_node_id* attribute, functionally derived from the archetype local ontology.

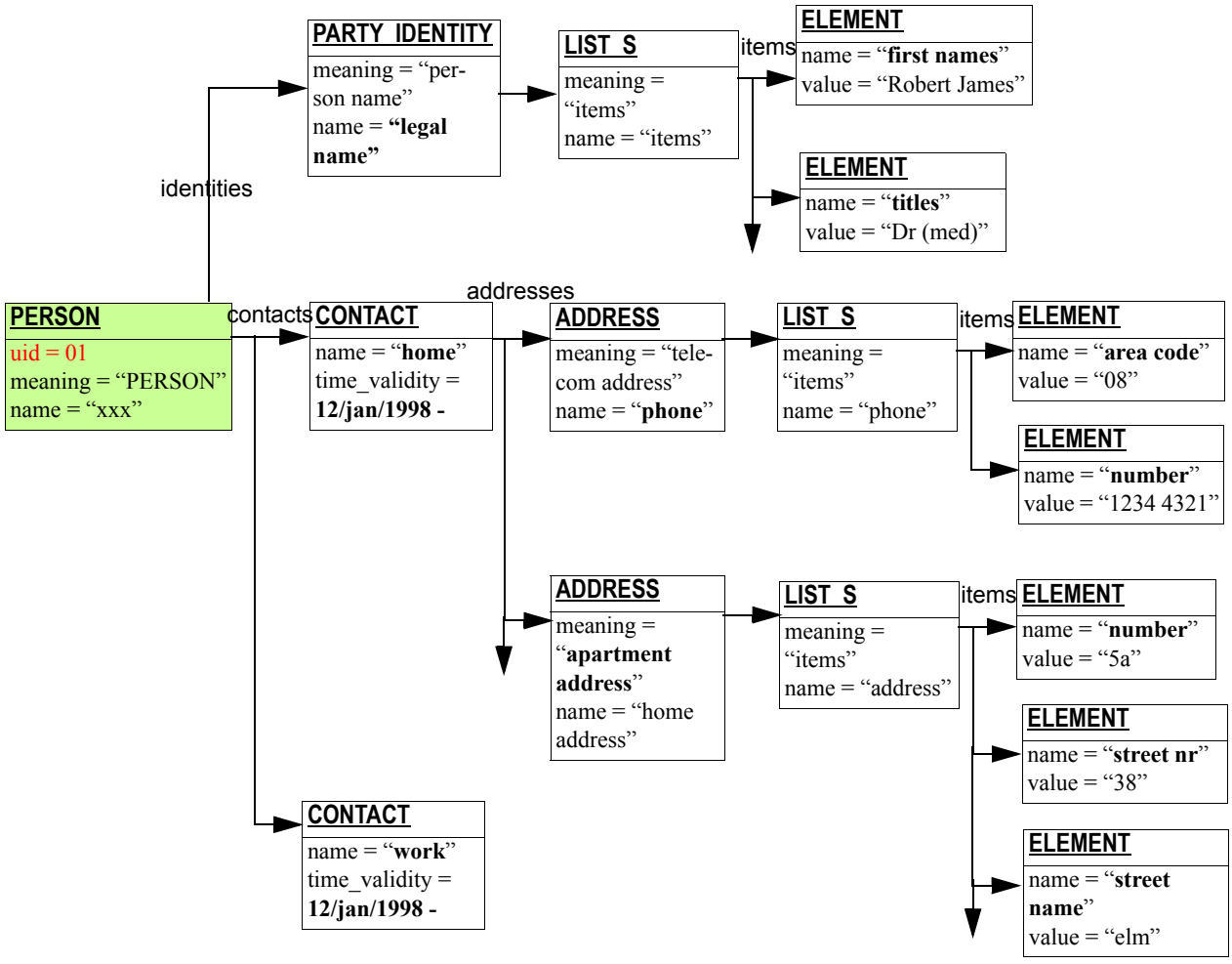


FIGURE 3 Person Demographic Information

2.3.1.2 Clinician

Credentials

2.3.1.3 Health Care Facility

2.3.1.4 Group

FIGURE 4 illustrates the demographic information for a cadiac surgery team in a hospital. The group includes a head surgeon, anaesthetist, assistant surgeon, and presumably others (not shown). Each of

these members of the team have an employment relationship with the hospital (shown only for one surgeon, in the interests of clarity).

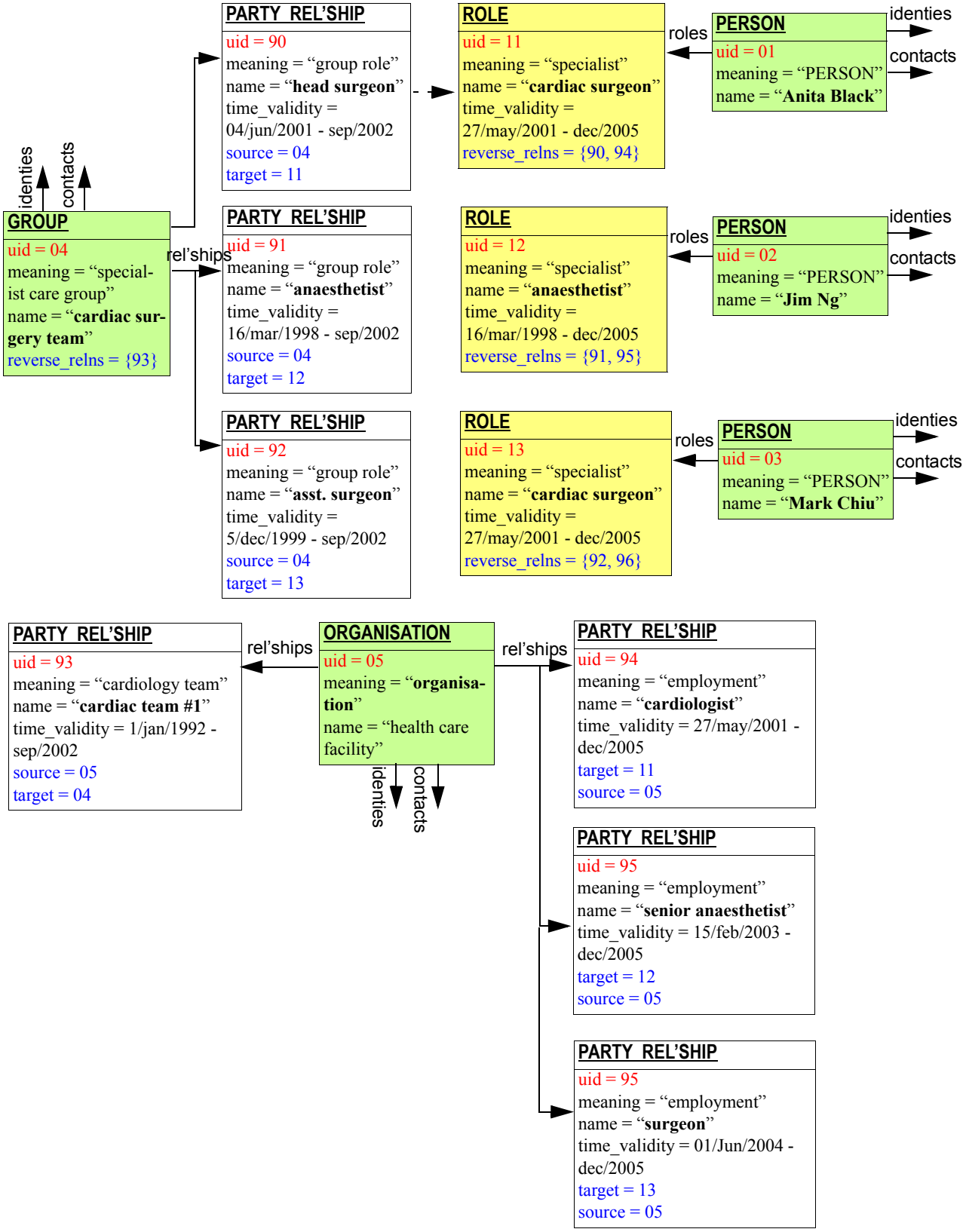


FIGURE 4 Group Demographics

2.3.2 Relationships

2.3.2.1 Familial Relationship

2.3.2.2 Employment Relationship

2.3.2.3 Patient

FIGURE 5 shows a simple way of representing the patient relationship between a person and a health care organisation.

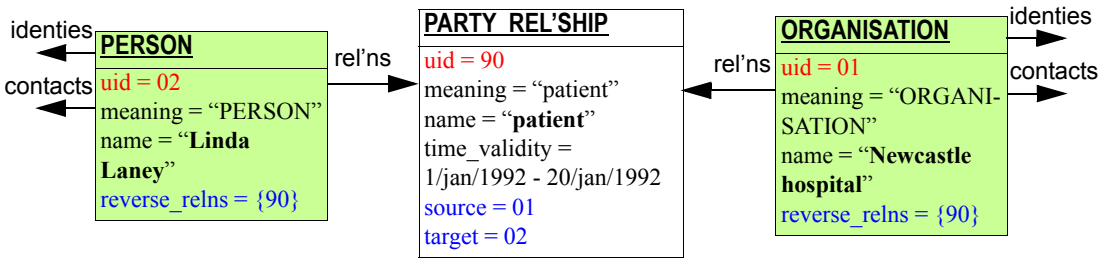


FIGURE 5 Simple Patient Relationship

FIGURE 6 shows the same logical relationship, but with both Parties acting through Roles, representing their status as healthcare consumer and healthcare provider respectively. Each of these Roles has associated credentials which document its official nature within the healthcare system.

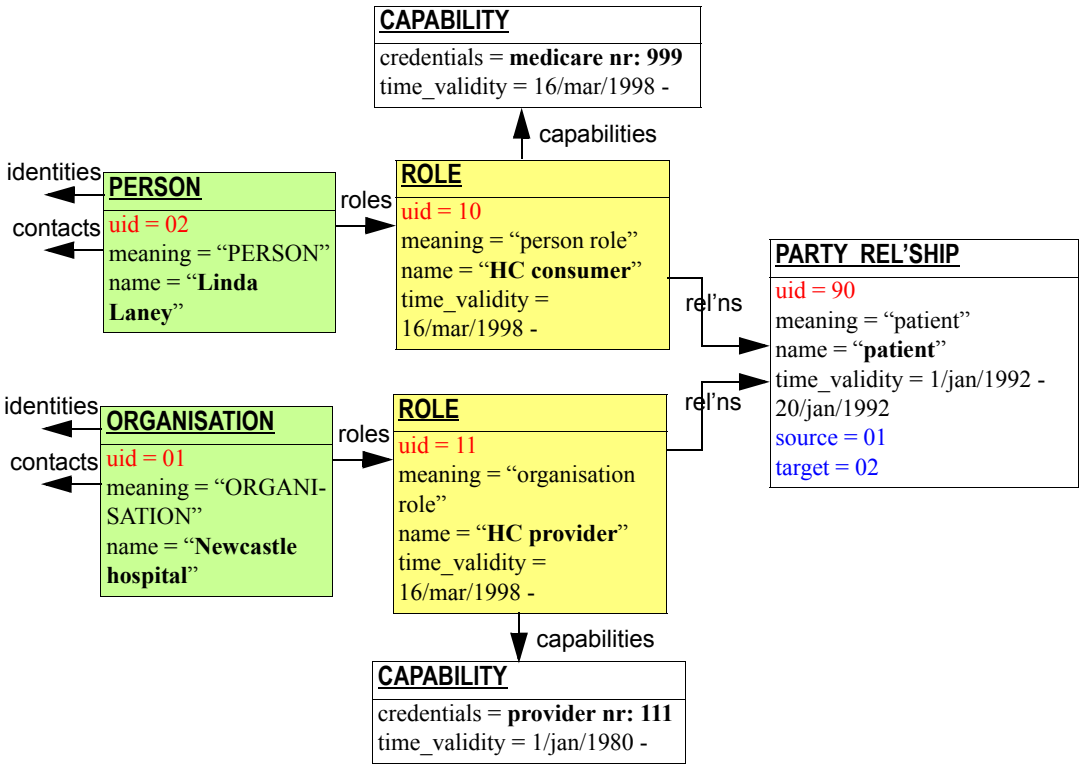


FIGURE 6 Patient Relationship with Roles and Credentials

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 Gray J, Reuter A. *Transaction Processing Concepts and Techniques*. Morgan Kaufmann 1993.

4 Sowa J F. *Knowledge Representation: Logical, philosophical and Computational Foundations*. 2000, Brooks/Cole, California.

5 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.

6 Sottile P.A., Ferrara F.M., Grimson W., Kalra D., and Scherrer J.R. The holistic healthcare information system. Toward an Electronic Health Record Europe 1999. Nov 1999; 259-266.

A.2 CEN

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

8 ENV 13606-2 - *Electronic healthcare record communication - Part 2: Domain term list*. CEN/ TC 251 Health Informatics Technical Committee.

9 ENV 13606-3 - *Electronic healthcare record communication - Part 3: Distribution rules*. 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.3 GEHR Australia

11 Beale T. *Northern Territory Health Demographic Model*.

12 Heard S. *GEHR Project Australia, GPCG Trial*. Available at http://www.gehr.org/gpcg/ehra.htm.

13 Beale T, Heard S. *GEHR Technical Requirements*. See http://www.gehr.org/technical/requirements/gehr_requirements.html.

A.4 OMG

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

A.5 HL7

15 HL7 version 3 2nd Ballot specification. Available at http://www.hl7.org.

A.6 Software Engineering

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

17 Walden K, Nerson J. *Seamless Object-oriented Software Architecture*.
Prentice Hall 1994

18 Gamma E, Helm R, Johnson R, Vlissides J. *Design patterns of Reusable Object-oriented Software*
Addison-Wesley 1995

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

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

21 Booch G, Rumbaugh J, Jacobsen I. *The Unified Modelling Language User Guide*. Addison esley 1999.

A.7 Resources

22 Arden Syntax. http://www.cpmc.columbia.edu/arden/

23 Asbru / The Asgaard Project. http://smi-web.stanford.edu/projects/asgaard/

24 Digital Imaging ad Communications in Medicine (DICOM). http://medical.nema.org/dicom.html.

25 EON ref required

26 GLIF (Guideline Interchange Format). http://www.glif.org/.

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

28 ProForma language for decision support. http://www.acl.icnet.uk/lab/proforma.html.

29 SynEx project, UCL. http://www.chime.ucl.ac.uk/HealthI/SynEx/.

END OF DOCUMENT