openEHR

Release 1

(in development)

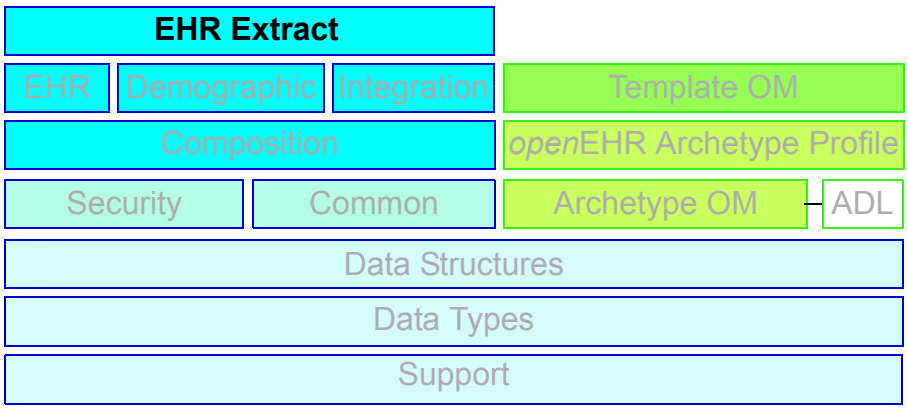
The *open*EHR Reference Model

Extract Information Model

Editors: {T Beale, H Frankel}[a]		
Revision: 2.1	*Pages:* 55	*Date of issue:* 29 Mar 2012

a. Ocean Informatics

Keywords: EHR, reference model, extract, openehr



© 2003-2010 The *open*EHR Foundation.

The *open*EHR Foundation is an independent, non-profit community, facilitating the 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

Copyright Notice

© Copyright openEHR Foundation 2001 - 2012
All Rights Reserved

1. This document is protected by copyright and/or database right throughout the world and is owned by the openEHR Foundation.
2. You may read and print the document for private, non-commercial use.
3. You may use this document (in whole or in part) for the purposes of making presentations and education, so long as such purposes are non-commercial and are designed to comment on, further the goals of, or inform third parties about, openEHR.
4. You must not alter, modify, add to or delete anything from the document you use (except as is permitted in paragraphs 2 and 3 above).
5. You shall, in any use of this document, include an acknowledgement in the form: "© Copyright openEHR Foundation 2001-2012. All rights reserved. www.openEHR.org"
6. This document is being provided as a service to the academic community and on a non-commercial basis. Accordingly, to the fullest extent permitted under applicable law, the openEHR Foundation accepts no liability and offers no warranties in relation to the materials and documentation and their content.
7. If you wish to commercialise, license, sell, distribute, use or otherwise copy the materials and documents on this site other than as provided for in paragraphs 1 to 6 above, you must comply with the terms and conditions of the openEHR Free Commercial Use Licence, or enter into a separate written agreement with openEHR Foundation covering such activities. The terms and conditions of the openEHR Free Commercial Use Licence can be found at http://www.openehr.org/free_commercial_use.htm

© 2003-2010 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

Amendment Record

Issue	Details	Who	Completed
RELEASE 1.x			
2.1	**SPEC-313**. Resolve various open modelling questions in existing Extract spec. **SPEC-314**. Correct modelling inconsistency of every EXTRACT_CHAPTER being related to a single Entity	T Beale	29 Mar 2012
RELEASE 1.0.2			
RELEASE 1.0.1			
2.0	**CR-000189**. Add LOCATABLE.*parent*. New invariant in EHR_EXTRACT. **CR-000186**: Upgrade EHR_EXTRACT to Release-1.0. Major redevelopment **CR-000219**: Use constants instead of literals to refer to terminology in RM.	T Beale T Beale R Chen	20 Feb 2007
RELEASE 1.0			
RELEASE 0.95			
1.3.5	**CR-000118**. Make package names lower case.	T Beale	10 Dec 2004
RELEASE 0.9			
1.3.4	**CR-000041**. Visually differentiate primitive types in openEHR documents.	D Lloyd	04 Oct 2003
1.3.3	**CR-000013**. Change key class names, according to CEN ENV 13606.	S Heard, D Kalra, D Lloyd, T Beale	15 Sep 2003
1.3.2	**CR-000003**, **CR-000004** (changes to versioning and LOCATABLE). MESSAGE_CONTENT now inherits from LOCATABLE.	T Beale, Z Tun	18 Mar 2003
1.3.1	**Formally validated using ISE Eiffel 5.2.** Revised structure of MESSAGE class to align better with CEN 13606-4. Renamed EHR_EXTRACT.*hca_authorising* to *originator*, similar to 13606.	T Beale	26 Feb 2003
1.3	**Changes post CEN WG meeting Rome Feb 2003.** Added attestations to X_TRANSACTION class. Significantly improved documentation of requirements, comparison to CEN 13606-4.	T Beale, S Heard, D Kalra, D Lloyd	07 Feb 2003
1.2.2	Minor corrections to diagrams and class definitions.	Z Tun	08 Jan 2003
1.2.1	Added senders_reference to conform to CEN 13606-4:2000 section 7.4.	T Beale	04 Jan 2003
1.2	Rewritten and restructured as two packages.	T Beale	07 Nov 2002
1.1	Moved part of EHR_EXTRACT into MESSAGE. Allow for multi-level archetypable Folder structures.	T Beale, D Kalra, D Lloyd	07 Oct 2002
1.0	Taken from EHR RM.	T Beale	07 Oct 2002

© 2003-2010 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

Acknowledgements

Thanks to...

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

© 2003-2010 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

Table of Contents

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

1 Introduction

1.1 Purpose

This document describes the architecture of the *open*EHR EHR Extract Information Model. This model formally defines the concepts of 'extract request', 'extract', various kinds of content including *open*EHR and non-*open*EHR, and a message wrapper. It covers use cases including EHR system communication, other clinical content messaging, and EHR system synchronisation, as well as providing an equivalent of the ISO 13606 EHR Extract.

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 Architecture Overview
- The *open*EHR Modelling Guide
- The *open*EHR Support Information Model
- The *open*EHR Data Types Information Model
- The *open*EHR Data Structures Information Model
- The *open*EHR Common Information Model
- The *open*EHR EHR Information Model
- The *open*EHR Demographic Information 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 format at http://www.openehr.org/svn/specification/TRUNK/publishing/architecture/rm/ehr_extract_im.pdf. New versions are announced on openehr-announce@openehr.org.

THIS DOCUMENT IS UNDER ACTIVE DEVELOPMENT AND IS NOT YET SUBJECT TO ARB CONTROL.

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

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

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 provided on the mailing list openehr-technical@openehr.org, or by private email.

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.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

2 Requirements

2.1 Overview

This section describes the requirements that the *open*EHR Extract Information Model (IM) is designed to meet. Requirements are expressed in terms of a description of the assumed operational environments (which acts as a design constraint), a set of use cases, functional and security requirements. The family of use cases describe coarse-grained information import to and export from health information systems, using *open*EHR standardised information structures as the *lingua franca*. The Extract IM is neutral with respect to the communication technology used between systems: the information structures can equally be used in a web services environment or in a messaging environment, including secure email. The concrete method of communication is therefore not a factor in the scenarios described here.

2.2 Operational Environment

2.2.1 *open*EHR Environments

The assumed operational *open*EHR environment for *open*EHR Extracts is shown in FIGURE 1. In this figure, a Request for "information from the records of one or more 'subjects'" is created by a Requesting system. A *subject* record may be a patient EHR, a Person record in a demographic system, or any other logically meaningful top-level entity. Responding system(s) reply in the form of one or more Extracts. The Request/Response interaction is enabled by a transport mechanism and possibly other services. This be be in the form of comprehensive middleware, web services, or simple point-to-point protocols such as SMTP (email) transport.

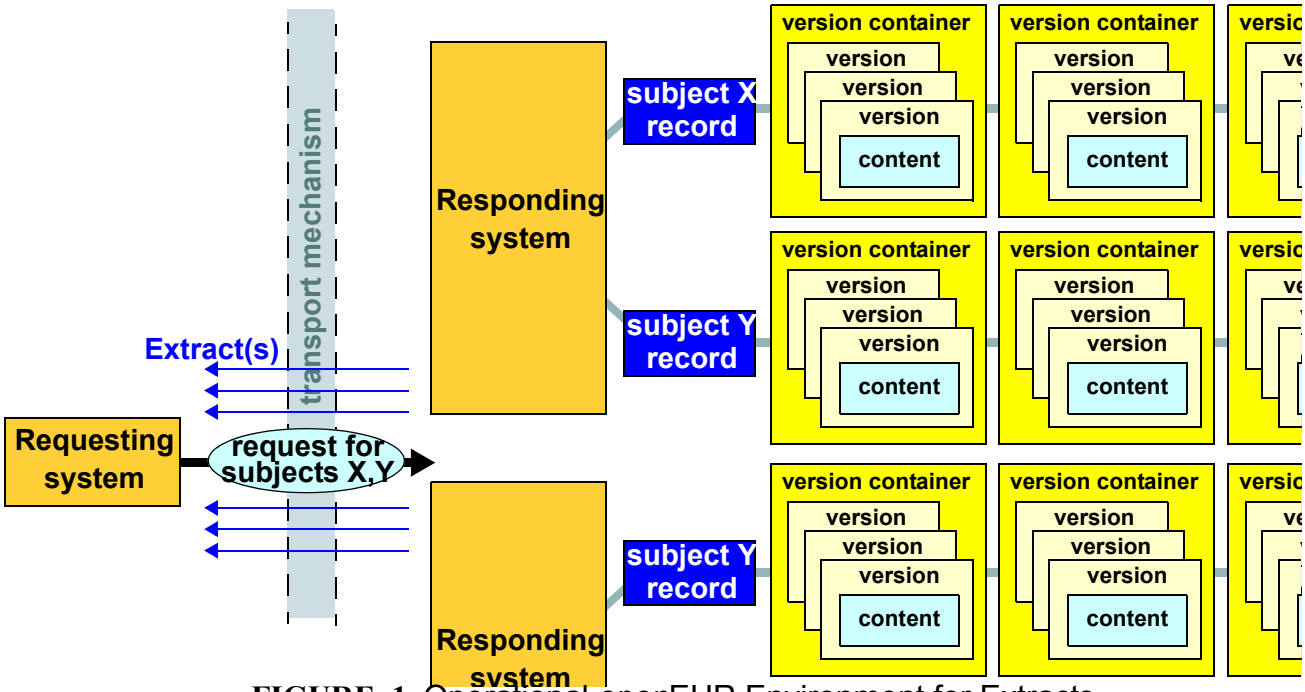


FIGURE 1 Operational *open*EHR Environment for Extracts

Information in Responding systems is assumed to be in the following form.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

- Each such system contains one or more Subject records (e.g. EHRs); there may be records for the same Subject in more than one system.
- Each Subject record consists of one or more Version containers, each of which contains the version history of a particular piece of content. For an EHR, distinct containers are used for persistent Compositions (e.g. “medications list”, “problem list”) and event Compositions (Compositions created due to patient encounters and other events in the clinical care process).
- Each Version within a Version container corresponds to one version of the content managed by that container, in other words the state of a particlar content item at some point in time when it was committed by a user.
- Groups of Versions, each one from a different Version container within a Responding system correspond to Contributions, i.e. the *open*EHR notion of a “change-set”. Any particular Contribution corresponds to the set of Versions committed at one time, by a particular user to a particular system.

The above relationships reveal a hierarchy of potential 1:N relationships in the information accessible to the requesting system, with Contributions forming an alternative view of the content. At each level of the hierarchy, a system of identifiers is needed, e.g. to distinguish Subjects, to distinguish Versions and so on. In some specific circumstances, some of these may be reduced to 1:1 relationships, e.g. there may be no versioning, removing the need for specific identifiers for versions of an item.

2.2.2 Non-*open*EHR Environments

The *open*EHR Extract defined in this specification can be used in non-*open*EHR environments, where the aim is to define messages whose content is expressed as templated archetypes. In general, not much can be assumed about the internal data architecture of such systems. For the purposes of this specification, the existence of two levels of information is assumed:

- ‘record’ or equivalently ‘patient’ - i.e. a division of information on the basis of subject of care;
- ‘document’ (Composition in *open*EHR), which is the coarsest grain item making up a record.

Regarding versioning in non-*open*EHR systems, it is assumed that some systems may support basic concepts including:

- document version;
- document version set - an identifier of a group of versions of the same logical item;
- document type / schema type - an identifier of some kind of model, schema or content type of a given document;
- document type version - the version of document type.

A typical environment in which Extracts can be used to send legacy information in archetyped form is one in which cross-enterprise communications are required, including discharge summaries and referrals. The content of such messages may be defined in terms of archetypes, and then templated in order to define the total content of the message.

2.2.3 Location of Information

In more advanced environments, there may be a health information location service which obviates the need for any knowledge on the part of the requestor about which systems contain information on a particular Subject of interest (e.g. a certain patient); in simpler environments, the requesting system

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

may need to explicitly identify the target systems of the request. FIGURE 2 illustrates a direct request and a request mediated by a location service.

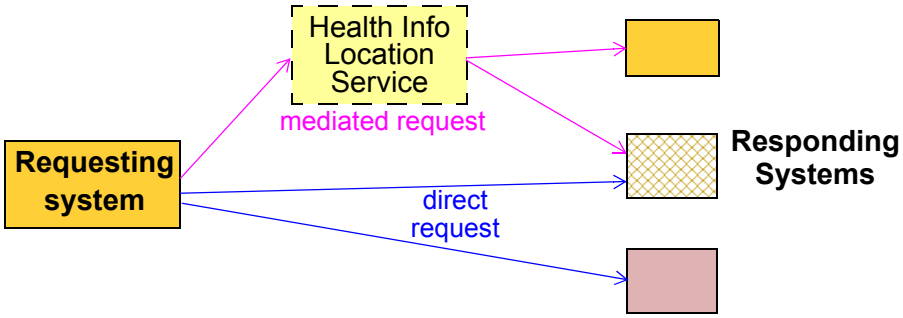


FIGURE 2 Health Information Reference Environment

This specification assumes that the EHR Request and Extract is between the Requesting system and each Responding system, even if the list of relevant Responding systems has been generated by a location service. In other words, this Extract specification does not encompass the idea of a compendium of Extracts from multiple Responding systems.

2.2.4 Granularity of Extract Data

In FIGURE 1 the lowest level of information shown in a Responding system is simply marked 'content'. This corresponds to top-level information structures such as Compositions, Folder trees, Parties etc in *open*EHR. Each such content item potentially contains a whole hierarchy of information items, only some of which is generally of interest to the Requestor. The typical database idea of a "query result" is usually expected to return only such fine-grained pieces. However, the Extract specification here only allows for a granularity of Compositions ('documents' etc), rather than fine-grained query responses which are dealt with by other means. This is because the primary use case of an Extract is to make parts of an EHR available somewhere else, rather than to intelligently query the record *in situ* and return the result.

2.2.5 Time

Versioned health record systems are 'bitemporal' systems, because they include two notions of time. Times mentioned in the data relate to *real world* events or states, such as the time of diagnosis of diabetes, or the date of discharge of a patient from a hospital. Real world time is distinquished from *system time*, which is the time of events in the information system itself, such as committal of Contributions. Both real world time and system time may need to be specified in a Request.

2.3 Use Cases

The following sections describe typical use cases that the Request/Extract model must satisfy.

2.3.1 Single Patient, *Ad Hoc* Request

A key clinical use case is the need to obtain some or all of a patient's EHR from a remote system or systems. The request is most likely to be made due to the patient having been referred to another provider (including discharge to tertiary care), but other reasons such as falling ill while travelling will also figure.

The request might be made to an identified system, such as the system of the referring provider, or it may be made to all systems containing data on the given patient, if a health information location service is available.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

The contents of the request may be specified in different ways, either by the clinician and/or the software, as follows:

- get this patient's entire EHR from an identified source for the first time;
- get all changes to this patient's EHR from specified (e.g. referring or GP) system since the last time this was done by me;
- get persistent Compositions such as "current medications", "problem list" and "allergies and interactions";
- get Compositions matching a specific query, such as "blood sugar level measurements during the last six months".

The meaning of time is content-dependent. For Observations in an *open*EHR EHR, the sample times might be specified; for an Evaluation that documents a diagnosis, times for date of onset, date of last episode, date of resolution could all be specified.

2.3.2 Multiple Patient, Batch Send

A very common requirement for pathology laboratories is to be able to send result data for tests done for a number of patients, on a periodic batch basis, to a known receiver such as a hospital, clinic or state health system. The batch send is usually seen as a "standing request" by the receiver, and may be on a periodic basis (e.g. send all the results available every hour), an "as-available" basis, or according to some other scheme.

Such data are currently often sent as HL7v2, Edifact or other similar messages.

To Be Continued: patient per message? Send trigger?

To Be Continued: Extract may be sent unsolicited - i.e. no Request

2.3.3 Previous Versions and Revision Histories

In some circumstances, a request may be made for versions other than the latest of certain content. This might happen due to a study or medico-legal investigation that needs to establish what information was visible in certain systems at an earlier time. For example there may be a need to determine if the problem list, list of known allergies, and patient preferences were all compatible with the medications list at some earlier time.

As part of querying for previous versions, the revision histories of Versioned containers might be requested, in order to allow a user or software agent to determine which Versions are of interest.

2.3.4 Systematic Update and Persisted Requests

In larger healthcare delivery environments such as state and regional health services, patients are routinely treated by multiple providers, some or all of which are part of a large distributed clinical computing environment. They may visit various clinics, specialists and hospitals, each of which has its own patient record for each patient. However, there is usually a need for centralised aggregation of patient data within the overall health authority, with updating required on a routine basis.

In such situations, the general requirement is for a request for update, typically for more than one patient, to be made once, and for it to be acted upon repeatedly until further notice. Specific requirements may include:

- periodic updates of changes since last update, with a specified period;
- event-driven updates, whereby an update occurs when a certain event occurs in the server, e.g. "any change to the EHR", "any change to medications or allergies" and so on.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

For these situations, the request can be persisted in the server. Even for one-off *ad hoc* requests, the requestor may require the request to be persisted in the server, so that it can be referred to simply by an identifier for later invocations.

2.3.5 Sharing of non-EHR *open*EHR Data

There will be a need to be able to request information from *open*EHR systems and services other than the EHR, such as demographics and workflow, as they are developed. One likely purpose for such requests is for import from *open*EHR systems into non-*open*EHR systems, for example from an *open*EHR demographics service to an existing hospital Patient Master index.

It should be possible to use the same general form of request as used for an EHR. However, instead of specifying Extracts of patient records (EHRs), the data shared in these other cases will be whichever top-level business objects are relevant for that kind of service, e.g. PARTYs for the demographic service and so on.

2.3.6 Provision of data from non-*open*EHR Systems

One of the major uses of an *open*EHR system is as a shared EHR system, aggregating data from various existing systems in a standardised form. Data from such systems may be provided in different ways, including various messaging forms (HL7, Edifact), various kinds of EMR document (CEN EN13606, HL7 CDA), and other formats that may or may not be standardised.

The developers of some such systems may decide to provide an *open*EHR-compatible export gateway, capable of serialising various data into *open*EHR structures, particularly the Composition / `GENERIC_ENTRY` form (see *open*EHR Integration IM) which is highly flexible and can accommodate most existing data formats. Types of non-*open*EHR systems that may supply *open*EHR Extracts in this fashion include pathology systems and departmental hospital systems, such as radiology, (RIS), histopathology and so on.

Extract Requests might be specified in *open*EHR form (i.e. according to this document) or in some other form, such as web service calls or messages; either way, the logical request is the same, i.e. which patients, which content, which versions, and the update basis. The responses must be some subset of the *open*EHR Extract presented in this document.

2.3.7 Patient Access to Health Data

Direct access by patients to their own health data outside of clinical encounters is a common aspiration of e-health programmes around the world. It seems clear that there will be various modes by which such access will occur, including:

- patient carrying USB stick or other portable device containing some or all of health record;
- access from home PC via secure web service to EHR, in a manner similar to online banking;
- access to EHR data in the form of encrypted email attachments on home PC either sent unsolicited (e.g. a scan cc:d to patient by imaging lab) or by request of the patient;
- access to EHR in the waiting rooms of doctors' surgeries, clinics etc via kiosks or normal PCs.

Both the USB stick and email scenarios involve asynchronous access of EHR information, and can be addressed by the EHR Extract.

In the case of a portable device, the most obvious need is for the device to act as a synchronising transport between a home PC containing a copy of patient / family EHRs, and the EHR systems at various clinics that the patient visits. To achieve this, when changes are made at either place (in the

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

home record or in the record held at a clinic), it should be possible to copy just the required changes to the device. In *open*EHR terms, this corresponds to copying Contributions made since the last synchronisation.

The email attachment scenario is more likely to involve Extracts containing either information requested by the patient in a similar manner as for the *ad hoc* clinical requests described above (e.g. most recent test result) or laboratory information in the form of an *open*EHR Extract, destined for integration into an *open*EHR EHR. In the latter case, the information is likely to be in the form of Compositions containing `GENERIC_ENTRIES`, built according to legacy archetypes, although it could equally be in "pure" *open*EHR form (i.e. Compositions containing proper *open*EHR Observations).

2.3.8 Move of Entire Record

A patient's entire EHR may be moved permanently to another system for various reasons, due to the patient moving, a permanent transfer of care, or re-organisation of data centres in which EHRs are managed. This is known as *change of custodianship* of the record, and is distinct from situations in which copying and synchronisation take place.

The record is deleted (possibly after archival) from active use at the sending system. In these situations, the usual need is for an interoperable form of the complete EHR (including all previous versions) to be exported from the existing system and sent to the destination system, since in general the two systems will not have the same implementation platform, versioning model and so on. In some cases, the implementations may be identical, allowing a copy and delete operation in native representation could be used.

2.3.9 System Synchronisation

Mirroring

Two *open*EHR systems containing the same kind of data (i.e. EHR, demographic etc) may contain records that are intended to be logical "mirrors" (i.e. clones) of each other. In some cases, an entire system may be a clone of another, i.e. a "mirror system". Mirrored records are purely read-only, and in all cases, the mirror record or system is a slave of its source, and no local updates occur. Synchronisation is therefore always in one direction only.

To maintain the information in mirrored records, an efficient update mechanism is required. The *open*EHR Contribution provides the necessary semantic unit because it is the unit of change to any record in a system, it can also be used as the unit of update to the same record in a mirroring system.

The Virtual EHR

If changes are allowed to multiple systems that also systematically synchronise, a "virtual EHR" exists. This term indicates that the totality of changes taken together form a complete EHR, even if any any particular instant in any given system, not all such changes are visible. The virtual EHR is the usual situation in any large-scale distributed e-health environment. Synchronisation might be on an *ad hoc* or systematic basis, and may or may not be bidirectional. The difference between a synchronisation request and any other kind of request is that the request is specified not in terms of a user query but in terms of bringing the record up to date, regardless of what changes that might require.

Due to the way version identification is defined in *open*EHR (see Common IM, `change_control` package), the virtual EHR is directly supported, and synchronisation is possible simply by copying Versions from one place to another and adding them to the relevant Versioned container at the receiver end. In a large health computing environment, cloning and mirroring might be used systematically to achieve a truly de-centralised system.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

The defining condition of this use case is that one or more (possibly all) records in a system are maintained as perfect copies of the same records in other systems, with possible delays, depending on when and how updating occurs.

2.3.10 Communication between non-*open*EHR EMR/EHR systems

Since the *open*EHR Extract represents a generic, open standardised specification for representing clinical information, there is no reason why it cannot be used for systems that not otherwise implement *open*EHR. In this case, the Extract content is most likely to consist of Compositions and Generic_entries, and may or may not contain versioning information, depending on whether versioning is supported in the generating systems.

2.4 Technical Requirements

2.4.1 Specification of Content

Content is specifiable in terms of matching criteria. This can take two forms: lists of specific top-level content items, or queries that specify top-level items in terms of matching subparts.

To Be Continued:

Queries are expressed in the Archetype Query Language.

To Be Continued:

2.4.2 Specification of Versions

The *open*EHR Extract supports detailed access to the versioned view of data. Which versions of the content should be returned can be specified in a number of ways:

- as a version time of the source EHR at which all content should be taken
- in more specific terms, such as:
  - time window of committal;
  - with or without revision history, or revision history only;
  - all, some, latest versions of each content item;
- in terms of identified Contributions, Contributions since a certain time etc.

To Be Continued:

2.4.3 Completeness of Data

Information transferred in an EHR Extract needs to be self-standing in the clinical sense, i.e. it can be understood by the requestor without assuming any other means of access to the responding system. In general, this means that for references of any kind in the transferred EHR data, the Extract needs to either contain the reference target, or else it must be reasonable to assume that the requestor has independent access, or else doesn't need it.

References to Other Parts of the Same EHR

In *open*EHR, there are two kinds of cross reference within an EHR: `LINKs` (defined in `LOCATABLE`), and hyperlinks (`DV_TEXT`.*hyperlink*). Both of these use a `DV_EHR_URI` instance to represent the link target. The contents of the URI are defined in the Architecture Overview.

To Be Continued: Are EHR ids always included in URIs?

In some cases, referenced items within the same EHR will need to travel with the originally requested item, in order for the latter to make sense. For example, a discharge summary or referral might refer

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

(via LINKs) to other Compositions in the EHR, such as Medication list, Problem list, and lab reports. On the other hand, there may be links within the requested item to objects that are not required to be sent in an Extract.

Which links should be followed when building an Extract can be specified in terms of:

- link depth to follow, i.e. how many jumps to continue from the original item; a value of 0 means “don’t follow any links”.

In addition, for LINK instances, the following can be specified:

- link type to follow, i.e. only follow links whose *type* attribute matches the specification.

References to Other EHRs

References to items in other EHRs may occur within an EHR, e.g. to the EHR of a parent, other relation, organ donor etc. There is no requirement for such links to be followed when constructing EHR Extracts.

References to Resources Outside the EHR

Computable references can also be made to external items from within an *open*EHR EHR. Instances of the data type DV_URI occurring either on their own or in a DV_TEXT hyperlink are typically used to refer to resources such as online guidelines and references. Instances of DV_MULTIMEDIA can contain DV_URI instances that refer to multimedia resources usually within the same provider enterprise, e.g. radiology images stored in a PACS. Since URIs are by definition globally unique, they remain semantically valid no matter where the data containing them moves. However, there is no guarantee that they can always be resolved, as in the case of a URI referring to a PACS image in one provider environment when transferred to another. This is unlikely to be a problem since images are usually represented in the EHR as a small (e.g. 200kb) JPG or similar, and it is almost never the intention to have original image sets (which may be hundreds of Mb) travel with an EHR. Requests to access original images would be made separately to a request for EHR Extracts.

References to Demographic Entities

Two kinds of demographic entities are referred to throughout an *open*EHR EHR. Individual providers and provider institutions are referenced from PARTY_PROXY objects in the record, via the *external_ref* attribute, which contains references to objects within a demographic repository, such as an *open*EHR demographic repository, a hospital MPI or a regional or national identity service. The PARTY_IDENTIFIED subtype of PARTY_PROXY can in addition carry human readable names and other computational identifiers for the provider in question.

The second kind of demographic reference, found in the PARTY_SELF subtype of PARTY_PROXY, is to the EHR subject (i.e. patient), and may or may not be present in the EHR, depending on the level of security in place. Where it is present, it is to a record in a demographic or identity system.

For the contents of an EHR Extract to make sense to the receiver, referenced demographic items may have to be included in the Extract, if the receiver has no access to a demographic system containing the entities. Whether patient demographics are included is a separate matter, since the requestor system already knows who the patient is, and may or may not need them. The requestor should therefore be able to specify whether the Extract includes referenced demographic entities other than the subject, and independently, whether subject demographics should be included.

Archetypes and Terminology

Another kind of “reference” is terminology codes, stored in instances of the data type DV_TEXT (via the *mapping* attribute) and DV_CODED_TEXT.*defining_code*. In *open*EHR systems, all coded terms (i.e. instances of DV_CODED_TEXT) carry the text value of the code, in the language of the locale of

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

the EHR. For normal use, this is typically sufficient. However, for the purposes of decision support or other applications requiring inferencing, the terminology itself needs to be available. This specification assumes that where the requestor requires inferencing or other terminology capabilities, independent access to the complete terminology will be obtained.

The same assumption is made with respect to archetypes whose identifiers are mentioned in the EHR data or meta-data: archetype are not themselves included in Extracts, and have to be resolved separately.

2.4.4 Security and Privacy

Security becomes one of the most important requirements for the EHR Extract for the obvious reason of its exposure in potentially uncontrolled environments outside the (supposedly) secure boundaries of requesting or responding systems. The general requirement is that the contents of an Extract are based on:

- the access control rules defined in the `EHR_ACCESS` object at source;
- any other access rules defined in policy services or other places;
- authentication of the requesting user.

Digital signing should be used based on the (preferably certified) public key of the requestor. Notarisation might also be used to provide non-repudiable proof of sending and/or receiving Extracts, although this is outside the scope of this specification.

2.4.5 Update Basis

In addition to specifying the content a basis for update also needs to be specified. The simplest possible case is that of an *ad hoc* one-off query.

More complex cases are periodic update and event-driven update.

Persistent Request

To Be Continued:

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

3 Design Overview

3.1 Abstract Communication Model

The *open*EHR Extract model uses two design ideas. Firstly, the notion of a Request and an Extract (the reply) are distinguished. Extracts may include a copy of the Request to which the Extract is a reply, indicating what is actually in the Extract, which may differ from what was requested. Secondly, the common semantics of Requests and Extracts are modelled in a generic way, with a number of specialised Request and Extract types being based on the common classes. Different concrete types of Extract are thus used to satisfy particular groups of requirements, rather than trying to make one kind of Extract perform all possible tasks. FIGURE 3 illustrates key Extract communication scenarios, along with the various concrete Extract types defined by the model.

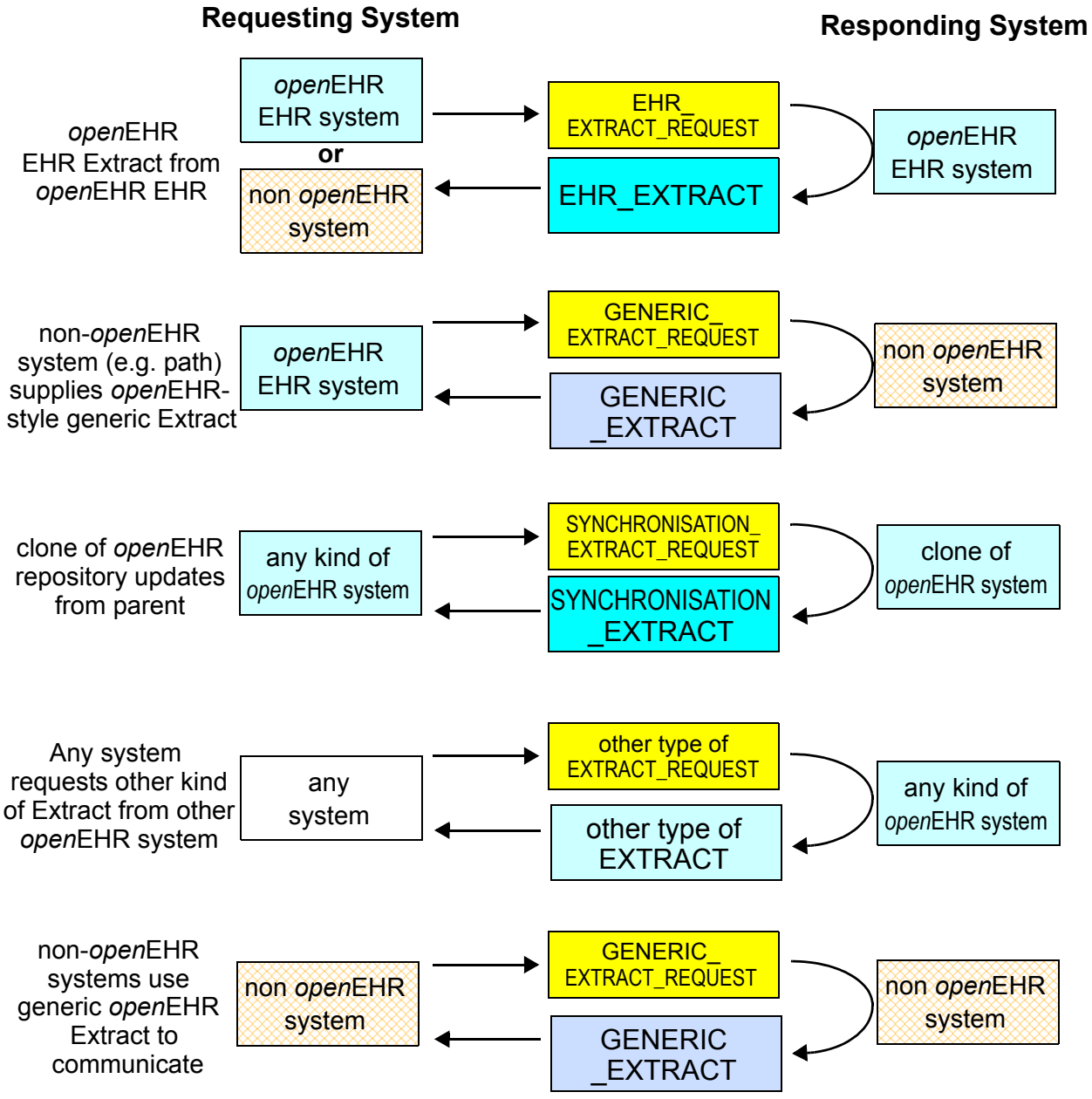


FIGURE 3 Use cases for *open*EHR Extracts

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

The EHR Extract is just one concrete type of Extract. The other types include:

- *Generic Extract*: an Extract type designed for use with non-*open*EHR systems communicating to *open*EHR systems, and users of the ISO 13606 Extract specification. The Generic Extract assumes the absolute minimum about what is in a system, while remaining withing the *open*EHR type system;
- *Synchronisation Extract*: an Extract used for updating mirrored EHRs across systems; uses Contributions as the grouping concept.

Other types of Extract may be defined in the future.

The pattern of requests and replies may be controlled in time. The simplest situation is single request, single reply. Other variants include:

- send request, persist in server, and use 'action' requests to obtain an Extract, which may have differing content each time;
- send request indicating a repeat period, at which server automatically sends replies;
- send request indicating a trigger event on which server should send replies.

3.2 Content Model & Representation

The general situation of any environment from which content needs to be extracted and sent to another system is that three categories of data may be needed: the 'key' clinical and administrative information, typically EHR content (e.g. patient blood pressure history); demographic information (indicating who the patient and professionals are that are mentioned in the clinical information; and relevant meta-data. These types of information almost always reside in different parts of the source system environment, but need to be combined within the Extract. The model defined in this specification allows for content to be flexibly aggregated in an Extract. The model therefore consists of a generic Extract containment structure into which specific archetyped content can be plugged, with the whole structure being templated. Each such template corresponds to one extract type, i.e. one message type. This is visualised in FIGURE 4.

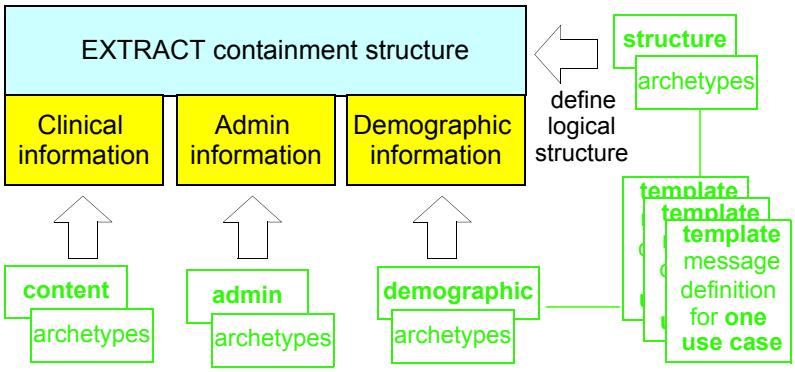


FIGURE 4 Archetyping of Extracts

The templates can be used to generate content in at least two ways. The default approach is to use the template for a given extract type, say 'referral', to create standard *open*EHR content according to the published *open*EHR EHR Extract schema (which largely re-uses the main *open*EHR EHR and demographic schemas). In this approach, every Extract message conforms to the standard schema - an XML Extract document is 'standard generic *open*EHR XML' in this case.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

An alternative approach converts each template into its own schema, to which all instances of that template conform. Different templates, e.g. discharge summaries and referrals define distinct schemas. In this method, the XML of any message is specific to its own schema. If changes are required in the message content, the schema usually has to be re-generated. The two methods are illustrated in FIGURE 5.

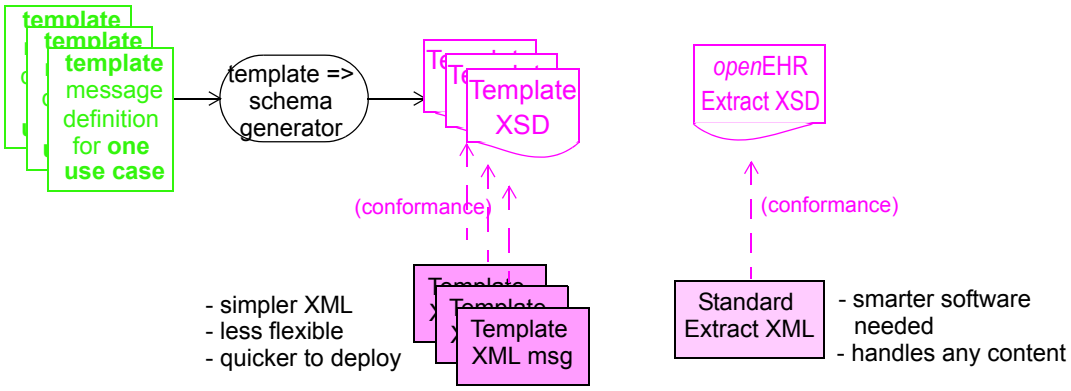


FIGURE 5 Representation options

Which method is used depends on the requirements of the environment and other factors, such as the stability of the template structures. The sections of this document describing specific kinds of extract show details of the concrete data representations involved.

3.3 Package Structure

The `rm.extract` package defines the semantics of Extracts from *open*EHR data sources, including EHRs. FIGURE 6 illustrates the package structure of the `rm.extract package`.

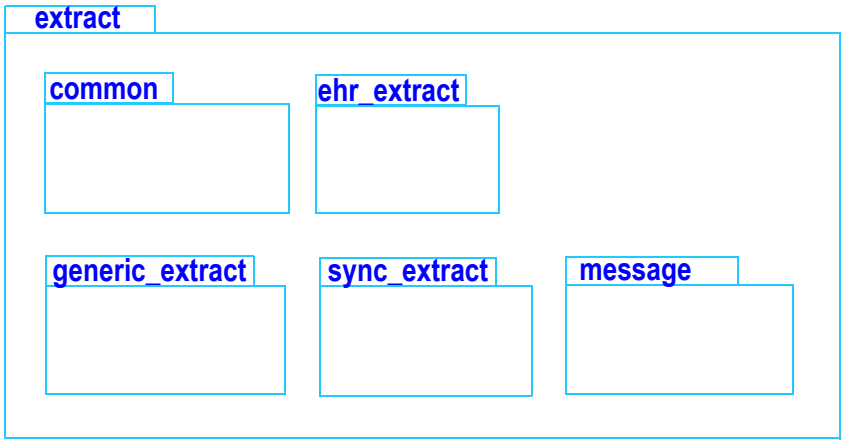


FIGURE 6 rm.extract Package

The sub-packages are as follows:

- `common`: semantics common to all Extracts;
- `ehr_extract`: semantics for the EHR Extract type;
- `generic_extract`: defines semantics of the Generic Extract type;
- `synchronisation_extract`: defines semantics of the Synchronisation Extract type;

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

- `message`: simple model of a message containing an Extract.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

4 Extract.common Package

4.1 Overview

The `rm.extract.common` package defines the semantics common to all kinds of Extract requests and Extracts. Requests and Extracts can be implemented as messages, or used as types in a webservice environment. In the latter, the Extract request semantics would most likely be replaced by equivalent service function definitions.

The Request contains a detailed specification of the repository content required in the Extract. A Request is not always needed, and an Extract may be sent unsolicited. Requests can be made persistent and/or event-driven, supporting various periodic and continuous update scenarios. Each request may specify the data from one or more entities, e.g. EHRs or subjects.

The Extract reply may optionally include a copy of the request. Its main content is in the form of *chapters*, under which *folders* and *content items* contain the requested content, or as much of it as was possible to retrieve. Specific types of content item, including *open*EHR content, generic EMR content, and demographic content are defined in various sub-packages, How the content is arranged within folders in each chapter, and across chapters is managed by the use of archetypes and templates for the Extract/chapter structure. FIGURE 7 illustrates the `rm.extract.common` package.

The typical instance structure of the `EXTRACT_REQUEST` and `EXTRACT` types is illustrated in FIGURE 8.

4.2 Design

4.2.1 Extract Request

The `EXTRACT_REQUEST` class consists of an *update_spec* specifying rules for update (one-off, periodic, event-driven etc), and a *extract_spec*, indicating what information from the target repository is required. The latter consists of an optional *version_spec*, indicating which versions (default is latest) and a *manifest*, specifying which entities (records or subjects, and optionally which items from those entities should be included.

Content Criteria Specification

The *extract_spec* part of the Request applies to all content in the Extract. The attributes are as follows:

- *extract_type*: what kind of Extract this is, e.g. |openehr-ehr|, |openehr-demographic|, |openehr-synchronisation|, |openehr-generic|, |generic-emr|, etc;
- *include_multimedia*: a flag indicating whether inline binary objects are included or not;
- *link_depth*: value indicating how many levels of link to follow when constructing the content of the Extract; the default is 0, meaning ‘don’t follow’;
- *criteria*: specifyies queries defining the required content of each entity record;
- *other_details*, an archetypable structure including further extract details.

The *criteria* attribute defines in the form of generic queries (i.e. queries that apply sensibly to any record) which items are to be retrieved from each entity’s record. Each query is expressed as a `DV_PARSABLE` instance, enabling any formalism to be used. The *open*EHR archetype query formalisms (AQL) could be used for example. Query expressions use variables such as $ehr to mean the current EHR from the *manifest* list. Queries may be as simple as the following:

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

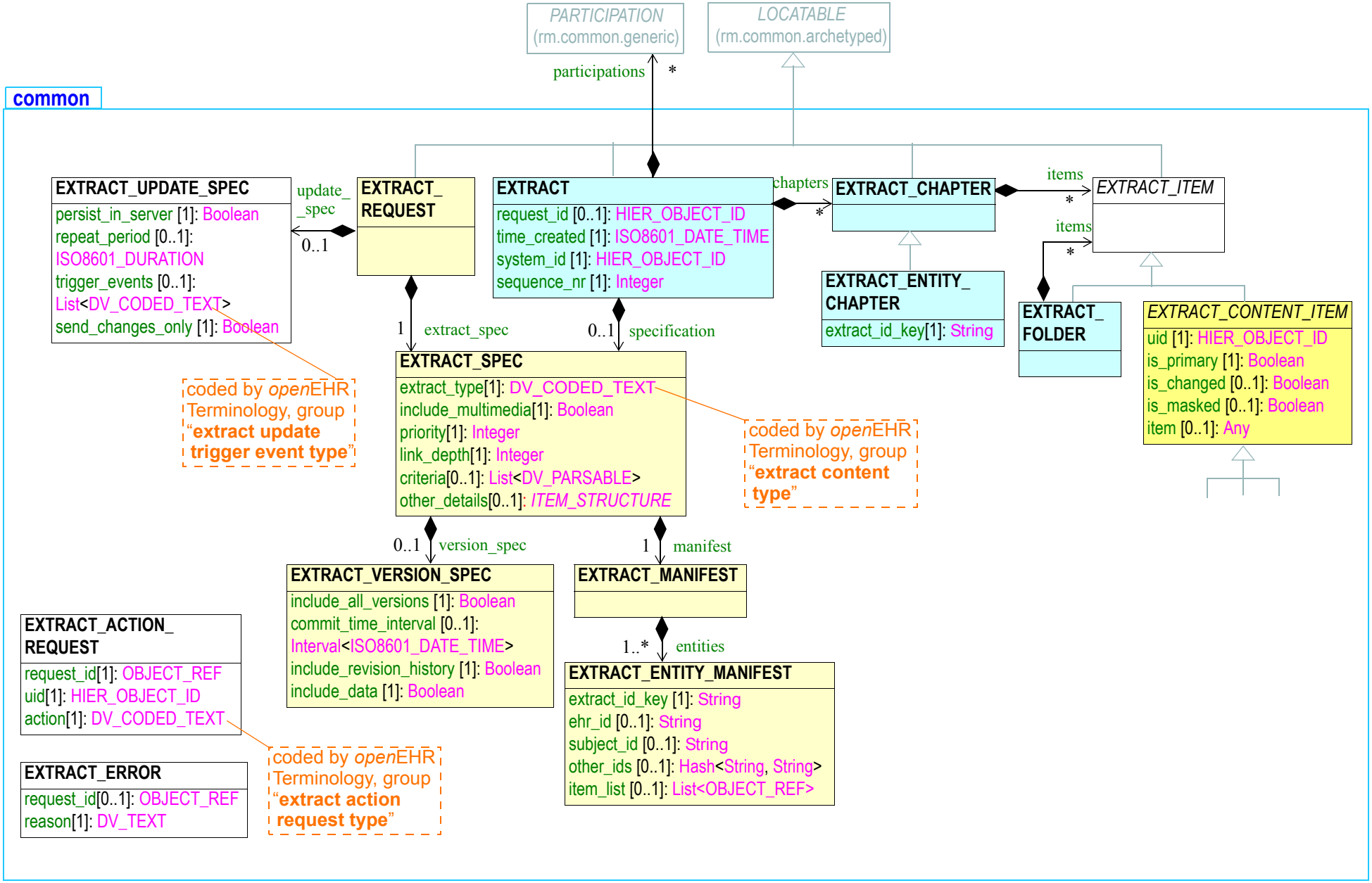


FIGURE 7 rm.extract.common Package

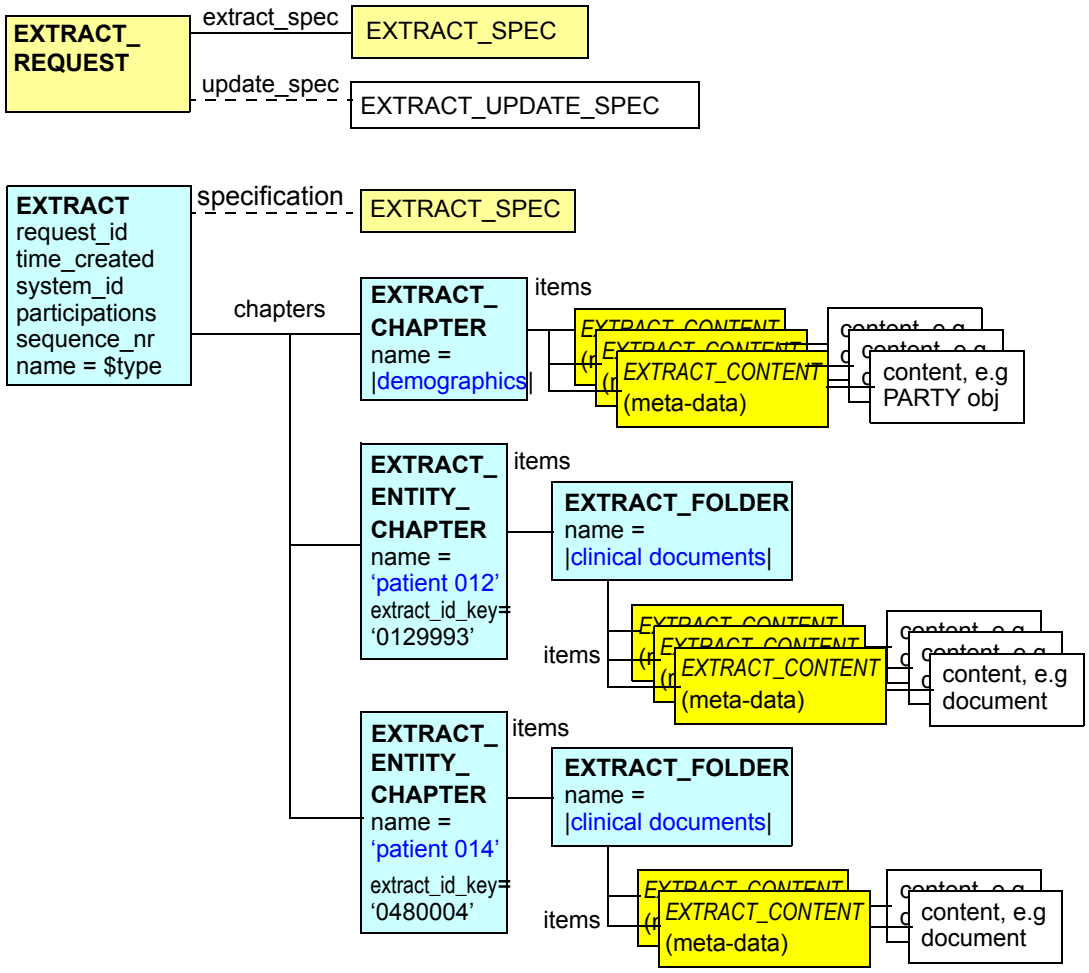


FIGURE 8 Abstract Request and Extract structure

- `SELECT * FROM $ehr` - get all items in the record
- `SELECT /ehr_access FROM $ehr` - retrieve the `EHR_ACCESS` object in an EHR
- `SELECT /ehr_status FROM $ehr` - retrieve the `EHR_STATUS` object in an EHR

More sophisticated queries can be used to obtain items corresponding to a specific criteria, e.g.:

- SELECT .... - retrieve last 6 months' worth of blood glucose measurements
- ... - retrieve ongoing medications list
- ..... - retrieve items relating to active pregnancy
- ..... - retrieve all GP encounter notes since 12-03-2005

To Be Continued:

Update Specification

The *update_spec* part of the Request specifies how the Request is to be processed by the server. The default situation, requiring no *update_spec* at all, is a one-off request. Other alternatives include:

- repeat Reqest with a defined period and/or trigger events;
- persisting the Request in the server so that the requestor can make later repeat requests by referring to a previously stored request by its identifier.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

An *ad hoc* repeat of a request persisted earlier is made using an EXTRACT_ACTION_REQUEST with *action* set to |repeat|. This specifies only the identifier of a request previously specified using an EXTRACT_REQUEST. Since requests are uniquely identified for all time, there can be no error in the identification.

The Manifest

The *manifest* that decides the scope of records to be retrieved. In the simplest case, only a single entity and no specific items will be identified (the content will be determined by the criteria in the EXTRACT_SPEC object); this will be used in the vast majority of single-patient request scenarios. However, some scenarios will be of a batch update nature, including pathology lab result update and situations where patient records are requested corresponding to a list of referrals received by a hospital since the last such update. In these cases, the request manifest may identify a number of entities (i.e. records or patients), each of which will be allocated a separate chapter in the resulting Extract.

The *item_list* of each entity manifest identifies individual items using OBJECT_REF instances each containing the HIER_OBJECT_ID identifier of a particular top-level object such as a Composition, or Party. This mechanism for identifying contents of an Extract is only expected to be used when a specific identifier is known, rather than when items corresponding to filtering criteria are requested. The latter are specified using the *criteria* attribute.

Version Specification

A Extract request in its simplest form has no version specification, corresponding to the assumption of 'latest available version' for each matched item. However, in some situations there is a need to retrieve previous versions, or information about versions. The version specification part of the Extract request allows this to be done. The possibilities available are as follows.

include_all_versions: the whole version 'stack' of each item matched by the *manifest* and retrieval *criteria* should be returned. Note that the result of this will be all available versions from the repository in question, which is in general not the same as all versions that have ever been created, since versions in the same logical version tree may exist at other repositories due to local modifications that have not been propagated to the target repository of the Extract Request.

include_revision_history: this flag indicates whether the *revision_history* of a VERSIONED_OBJECT is to be included in the Extract form of the version

4.2.2 Extract

Content Specification

The EXTRACT consists of a *request_id*, an optional *participations* list, an optional *specification* and a set of *chapters*, containing the retrieved content. The *participations* attribute denotes parties that were responsible for creating the Extract. The *specification* takes the same form as the *extract_spec* of the EXTRACT_REQUEST, but in an EXTRACT instance, indicates the *actual* contents of the Extract. This will usually differ in from the request specification in that it will list in the manifest section the actual entities (and the identifiers they are known by in the receiver system) retrieved, whereas the request may not specify any entities (it may rely on query criteria), or it may specify a different set of entities than were retrievable in the receiver system.

Content

The content of an Extract is contained within the *chapters* attribute, in the form of one of more instances of the EXTRACT_CHAPTER class or its subtype EXTRACT_ENTITY_CHAPTER. Within an EXTRACT_CHAPTER the *content* attribute contains a folder structure made of EXTRACT_FOLDER

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

objects which in turn contain the content items. The folder structure is defined by archetypes, and can be used to separate the parts of a health record, or group items according to some other scheme e.g. episode. A specific kind of chapter, the EXTRACT_ENTITY_CHAPTER, is used to carry content associated with *a single entity*, i.e. patient. This means that an Extract may contain data relating to multiple entities, i.e. multiple patients.

The EXTRACT_CONTENT_ITEM class is subtyped in later sections of this specification to define metadata for specific kinds of content.

4.2.3 Participations and Demographic Referencing

A key difference between an 'extract' or in fact any kind of message carrying information out of a system environment is that the Extract combines in one place information typically found in multiple services in the original environment. Two kinds of information routinely included in Extracts are clinical (i.e. EHR / EMR or similar) and demographic data regarding participations of the subject and other parties in the activities documented in the clinical information.

Within the *open*EHR EHR model, a 'participation' is defined by the PARTICIPATION class in the rm.common.generic package. Actual participation instances include a performer object (a subtype of PARTY_PROXY), and optionally an external reference object (type PARTY_REF). The latter contains a reference to the demographic entity within a demographic service, provider registry or similar.

Within the Extract, the content of PARTICIPATION instances is copied faithfully, including any inline *performer* information. If there was an *external_ref* present, pointing to a demographic entity that is also to be included in the Extract, this reference is rewritten to point to a local copy of the demographic entity (using a GUID), within the Entity chapter of the Extract.

The result of this is that for source data using external references to refer to common demographic entities, the logical structure is completely preserved within the Extract, with just the reference being rewritten to the GUID values used in the Extract (which may well come from the original demographic database or service).

A typical arrangement is to dedicate a single chapter to all demographic entities referenced within the Extract, optionally with an internal Folder structure. Some example arrangements are illustrated in the EHR and generic Extract sections below.

4.3 Class Descriptions

4.3.1 EXTRACT_REQUEST Class

CLASS	EXTRACT_REQUEST	
Purpose	Generic model of a Request for an Extract, containing an Extract specification.	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
1..1	**extract_spec**: EXTRACT_SPEC	Specification details of the request.
0..1	**update_spec**: EXTRACT_UPDATE_SPEC	Update details of the request.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	EXTRACT_REQUEST	
1..1 (redefined)	**uid**: HIER_OBJECT_ID	Identifier of this Request, generated by requestor.
Invariants	***Extract_spec_valid***: extract_spec /= Void ***Uid_exists***: uid /= Void	

4.3.2 EXTRACT_ACTION_REQUEST Class

CLASS	EXTRACT_ACTION_REQUEST	
Purpose	Generic model of a Request for an Extract, containing an Extract specification.	
Inherit	LOCATABLE	
Attributes	Signature	Meaning
1..1	**request_id**: OBJECT_REF	Identifier of previous EXTRACT_REQUEST.
1..1	**action**: DV_CODED_TEXT	Requested action: cancel, resend, send new
Invariants	***Request_id_valid***: request_id /= Void ***Action_valid***: action /= Void	

4.3.3 EXTRACT_SPEC Class

CLASS	EXTRACT_SPEC	
Purpose	Specification of an Extract's contents. Subtypes can be used to add details specific to the type of Extract. The specification consists of attributes specifying the directory, and two further groups of attributes in their own classes, namely a version specfication (which versions of information items are to be included) and a manifest (which entities are to be included in the extract).	
Use	Used in a request to specify an Extract, as well as to describe what is contained in an Extract.	
Attributes	Signature	Meaning
1	**extract_type**: DV_CODED_TEXT	Coded term indicating the type of content required, e.g. • |openehr-ehr| • |openehr-demographic| • |generic-emr| • |other|
1	**include_multimedia**: Boolean	Indicates whether inline instances of DV_MULTIMEDIA in the source data are included or not.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	EXTRACT_SPEC	
1..1	**link_depth**: `Integer`	Degree of links to follow emanating from content items specified for inclusion. The kind of links to follow is dependent on the type of Extract. All items at the target end of followed links at the given depth are also included in the extract; `EXTRACT_CONTENT_ITEM.`*is_primary* is used to differentiate. - 0 = don't follow; - 1 = follow first degree links; - 2 = follow 2nd degree links; - .... - n = follow nth degree links
1	**priority**: `Integer`	Requested priority of this request to be handled by server. Priority schemes are likely to be local, and use values agreed by both ends. TBD: alternative is standard coded terms
0..1	**criteria**: `List<DV_PARSABLE>`	Queries specifying the contents of this Extract.
1..1	**manifest**: `EXTRACT_MANIFEST`	Specification of entities (e.g. records) to include in the Extract.
0..1	**version_spec**: `EXTRACT_VERSION_SPEC`	Specification of which versions of information items to include in the Extract. If Void, the default is latest versions only (which is reasonable for non-versioning systems as well).
0..1	**other_details**: *ITEM_STRUCTURE*	Other specification items. Archetypable.
Invariants	***Extract_type_valid***: extract_type /= Void ***Link_depth_valid***: link_depth >= 0 ***Manifest_valid***: manifest /= Void	

4.3.4 EXTRACT_MANIFEST Class

CLASS	EXTRACT_MANIFEST	
Purpose	Specification of the candidate entities and optionally top-level items to be included in the Extract.	
Attributes	**Signature**	**Meaning**

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	EXTRACT_MANIFEST	
1..1	**entities**: `List <EXTRACT_ENTITY_MANIFEST>`	List of entity manifests uids of items included in the Extract; for *open*EHR data, these are uids identifying the version containers.
Invariants	***Entities_valid***: entities /= Void **and then** not entities.is_empty	

4.3.5 EXTRACT_ENTITY_MANIFEST Class

CLASS	EXTRACT_ENTITY_MANIFEST	
Purpose	The manifest for one entity, identifying the entity and optionally specifying top-level items to be included in the Extract. The list actually included may be modified by the *version_spec* part of the specification, and also by the *link_depth* attribute. In repeat (standing order) requests, the final inclusion may be modified by the *send_changes_only* flag of the *update_spec*. Various identifiers may be provided for the entity; these are to be used by the receiver system to locate the entity. The *extract_id_key* attribute is used to record the identifier that will be used throughout the Extract for this entity, including in instances of `EXTRACT_ENTITY_IDENTIFIER`.	
Attributes	**Signature**	**Meaning**
1	**extract_id_key**: `String`	Identifier by which this entity is known in the Extract. May be one of the other identifiers, e.g. *ehr_id* or *subject_id*, or it may be something else, including a simple integer.
0..1	**ehr_id**: `String`	EHR / EMR identifier for the entity at the target system.
0..1	**subject_id**: `String`	Subject (i.e. patient or similar) identifier for the entity at the target system.
0..1	**other_ids**: `Hash<String, String>`	Other identifiers that may be used to find the entity at the target system, keyed by type. May include medicare numbers, drivers license number, tax number etc.
0..1	**item_list**: `List<OBJECT_REF>`	List of uids of items included in the Extract; for *open*EHR data, these are uids identifying the version containers.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	EXTRACT_ENTITY_MANIFEST
Invariants	***Extract_id_key_valid***: extract_id_key /= Void **and then not** extract_id_key.is_empty ***Ehr_id_valid***: ehr_id /= Void **implies not** ehr_id.is_empty ***Subject_id_valid***: subject_id /= Void **implies not** subject_id.is_empty ***Other_ids_valid***: other_ids /= Void **implies not** other_ids.is_empty ***Item_list_valid***: item_list /= Void **implies not** item_list.is_empty

4.3.6 EXTRACT_VERSION_SPEC Class

CLASS	EXTRACT_VERSION_SPEC	
Purpose	Specification of what versions should be included in an Extract. By default, only latest versions are included in the Extract, in which case this part of the Extract specification is not needed at all. The attributes *include_all_versions* and *commit_time_interval* are used to modify this; the former forces all versions to be included; the latter limits the versions to be those latest versions committed in the time interval, or if *include_all_versions* is True, all versions committed in the time interval.	
Attributes	**Signature**	**Meaning**
0..1	**commit_time_interval**: `Interval <ISO8601_DATE_TIME>`	Specifies commit time interval of items to source repository to include in Extract. By default, only latest versions whose commit times fall in the range are included. If *include_all_versions* is True, then the range includes all versions committed within the interval.
1..1	**include_all_versions**: `Boolean`	True if all versions of each item in the Extract are included.
1..1	**include_revision_history**: `Boolean`	True if revision histories of the items in the Extract are included. If included, it is always the full revision history.
1..1	**include_data**: `Boolean`	True if the data of items matched by the content spec should be included. This is the default. If False, only revision history is included in serialised versions. Turning this option on in *open*EHR systems causes `X_VERSIONED_OBJECTs` to have *revision_history* set, but *versions* Void. Useful for interrogating a server without having to look at any content data. In other systems it may or may not have a sensible meaing.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	EXTRACT_VERSION_SPEC
Invariants	***Includes_revision_history_valid***: **not** include_data **implies** include_revision_history.

4.3.7 EXTRACT_UPDATE_SPEC Class

CLASS	EXTRACT_UPDATE_SPEC	
Purpose	Specification of the how the request should be processed by server. The request can be persisted in the server, meaning that a) it can be re-activated by the requesting system simply by indicating Request id, and b) that a changes-only pattern of Extract updates can be set up. To achieve this, the server has to remember what was send in the previous reponse. The update mode may be event-driven and periodic update or a mixture of both. The candidate items to be sent each time are the result of re-evaluating the content and versioning parts of the specification; what is actually sent is determined by the *send_changes_only* flag.	
Attributes	**Signature**	**Meaning**
1..1	**persist_in_server**: `Boolean`	If True, this Request is persisted in the server until further notice.
1..1	**send_changes_only**: `Boolean`	If True, send only items that are changed (including logical deletions) or new since last send. For persist_in_server Requests only.
0..1	**repeat_period**: `ISO8601_DURATION`	Period for resending update Extracts in response to original Request.
0..1	**trigger_events**: `List<DV_CODED_TEXT>`	Set of Event names that will cause sending of update Extracts. Event types include: • |any_change| - any change in content items matched by content specification, e.g. new versions of persistent compositions. If the content list allows matching of any, or a wide range of archetypes, this event type will match any additions to the record. • |correction| - match error corrections only, including deletions. • |update| - match updates (i.e. new versions) of included content items.
Invariants	***Overall_validity***: repeat_period /= Void **or** trigger_events /= Void ***Trigger_events_validity***: trigger_events /= Void **implies not** trigger_events.is_empty ***Send_changes_only_validity***: send_changes_only **implies** persist_in_server	

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

4.3.8 EXTRACT Class

CLASS	EXTRACT	
Purpose	Generic model of an Extract of some information from a repository; the generic parameters select which kind of specification and content the Extract carries.	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
1..1 (redefined)	**uid**: HIER_OBJECT_ID	Identifier of this Extract.
0..1	**request_id**: HIER_OBJECT_ID	Reference to causing Request, if any.
1..1	**sequence_nr**: Integer	Number of this Extract response in sequence of responses to Extract request identified by *request_id*. If this is the sole response, or there was no request, value is 1.
0..1	**specification**: EXTRACT_SPEC	The specification that this Extract actually conforms to; might not be identical with the specification of the corresponding request.
0..1	**participations**: List<PARTICIPATION>	Participations relevant to the creation of this Extract.
1..1	**chapters**: List<EXTRACT_CHAPTER>	The content extracted and serialised from the source repository for this Extract.
Invariants	***Uid_exists***: uid /= Void ***Participations_valid***: participations /= Void **implies not** participations.is_empty ***Specification_valid***: specification /= Void ***Sequence_nr_valid***: sequence_nr >= 1 ***Chapters_valid***: chapters /= Void	

4.3.9 EXTRACT_CHAPTER Class

CLASS	EXTRACT_CHAPTER	
Purpose	One content chapter of an Extract; contains information relating to only one entity.	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
0..1	**items**: List<EXTRACT_ITEM>	The information content of this chapter.
Invariants	***Content_valid***: content /= Void	

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

4.3.10 EXTRACT_ENTITY_CHAPTER Class

CLASS	EXTRACT_ENTITY_CHAPTER	
Purpose	Type of chapter that contains information relating to a single demographic entity.	
Inherit	EXTRACT_CHAPTER	
Attributes	**Signature**	**Meaning**
1..1	**entity_identifier**: EXTRACT_ENTITY_IDENTIFIER	Reference to entity, usually a demographic entity such as a patient that the content of this chapter relates to.
Invariants	***Entity_identifier_valid***: entity_identifier /= Void	

4.3.11 EXTRACT_ITEM Class

CLASS	*EXTRACT_ITEM (abstract)*	
Purpose	Abstract parent of Extract Folder and Content types.	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
Invariants		

4.3.12 EXTRACT_FOLDER Class

CLASS	EXTRACT_FOLDER	
Purpose	Folder in local Folder structure in an Extract. Empty Folders are allowed.	
Inherit	EXTRACT_ITEM	
Attributes	**Signature**	**Meaning**
0..1	**items**: List<EXTRACT_ITEM>	List of Folders and content items in this Folder.
Invariants	***Items_valid***: items /= Void **implies not** items.is_empty	

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

4.3.13 EXTRACT_CONTENT_ITEM Class

CLASS	*EXTRACT_CONTENT (abstract)*	
Purpose	Abstract model of a wrapper for one content item in an Extract, containing various meta-data. Indicates whether it was part of the primary set and what its original path was. Intended to be subtyped for wrappers of specific types of content.	
Inherit	`EXTRACT_ITEM`	
Attributes	**Signature**	**Meaning**
1 (redefined)	**uid**: `HIER_OBJECT_ID`	Unique id of this Extract item; used for all referencing from elsewhere in Extract, including from Extract Folders.
1	**is_primary**: `Boolean`	True if the content item carried in this container was part of the primary set for the Extract, i.e. not added due to link-following.
0..1	**is_changed**: `Boolean`	True if the content item carried in this container is an update due to change since last send, in repeat sending situations.
0..1	**is_masked**: `Boolean`	True if the content of this item has not been included due to insufficient access rights of requestor.
0..1	**item**: `Any`	Content object.
Invariants	***Uid_valid***: uid /= Void ***Item_validity***: is_masked **xor** item /= Void	

4.3.14 X_PARTICIPATION Class

CLASS	X_PARTICIPATION	
Purpose	Simplified form of `PARTICIPATION` used in Extracts, where the performer is a direct reference to the target demographic information elsewhere in the Extract.	
Inherit	`PARTICIPATION`	
Attributes	**Signature**	**Meaning**
1	**performer**: `String`	UID of the target Extract item.
Invariants	***Performer_valid***: performer /= Void	

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

5 Ehr_extract Package

5.1 Overview

The rm.extract.ehr_extract package defines an *open*EHR-specific variant of an EXTRACT_ITEM, which consists of a form of the VERSIONED_OBJECT<T> class from the rm.common.change_control package suitable for use in Extracts. The Extract form of the class is X_VERSIONED_OBJECT<T>, and consists of attributes which replicate functional values from the EHR form of the class. This class is further subtyped via binding to non-generic types corresponding to the top-level object types of the *open*EHR EHR.

FIGURE 9 illustrates the rm.extract.ehr_extract package. FIGURE 10 illustrates a typical instance structure of an EHR Extract.

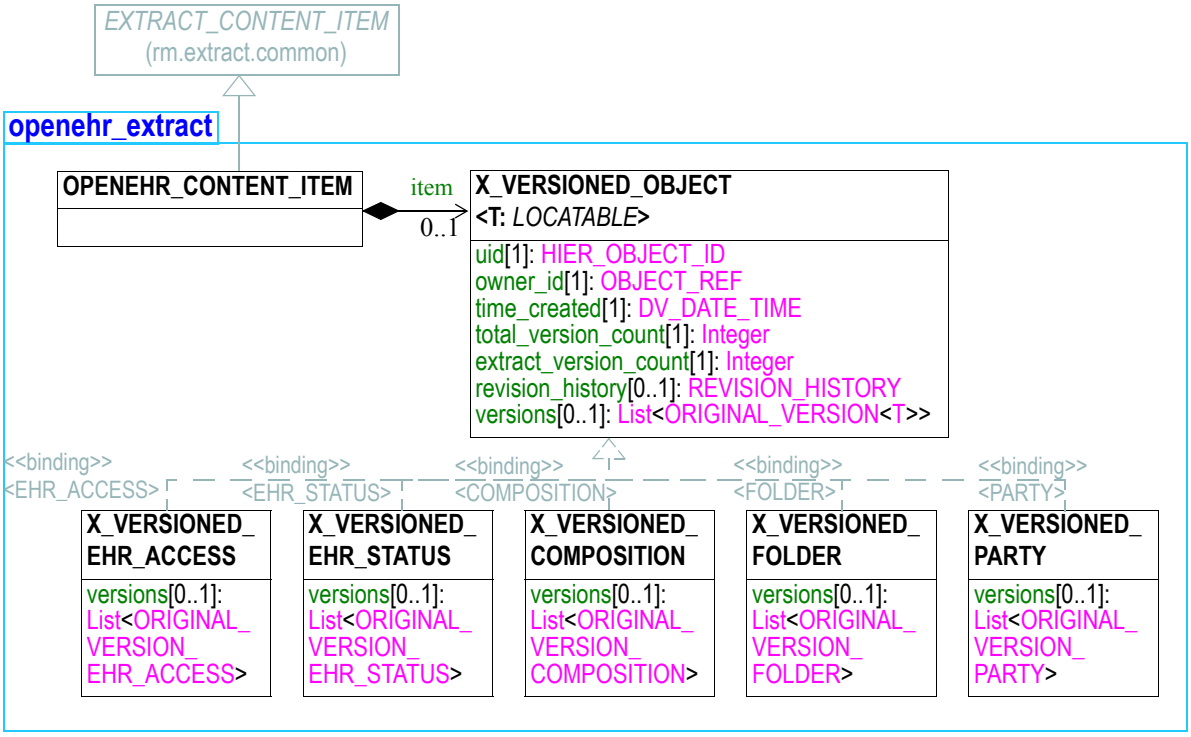


FIGURE 9 rm.extract.ehr_extract Package

5.2 Design

*open*EHR Extract Item

Items from an *open*EHR sytem included in an EHR Extract are always top-level objects (Composition, Directory, Ehr_access, Party, etc, and descendants) are expressed in the serialisable form of descendants of X_VERSIONED_OBJECT<T>. This type provides a standard interoperable way to serialise all or part of VERSIONED_OBJECTs for lossless transmission between systems, regardless of the likely implementation differences in versioning at each end. Accordingly, X_VERSIONED_OBJECT turns most functional properties of VERSIONED_OBJECT into data attributes. The two attributes of most interest are *revision_history*, which enables the optional inclusion of the complete revision history from the original VERSIONED_OBJECT, and *versions*, which allows any or all of the versions to

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

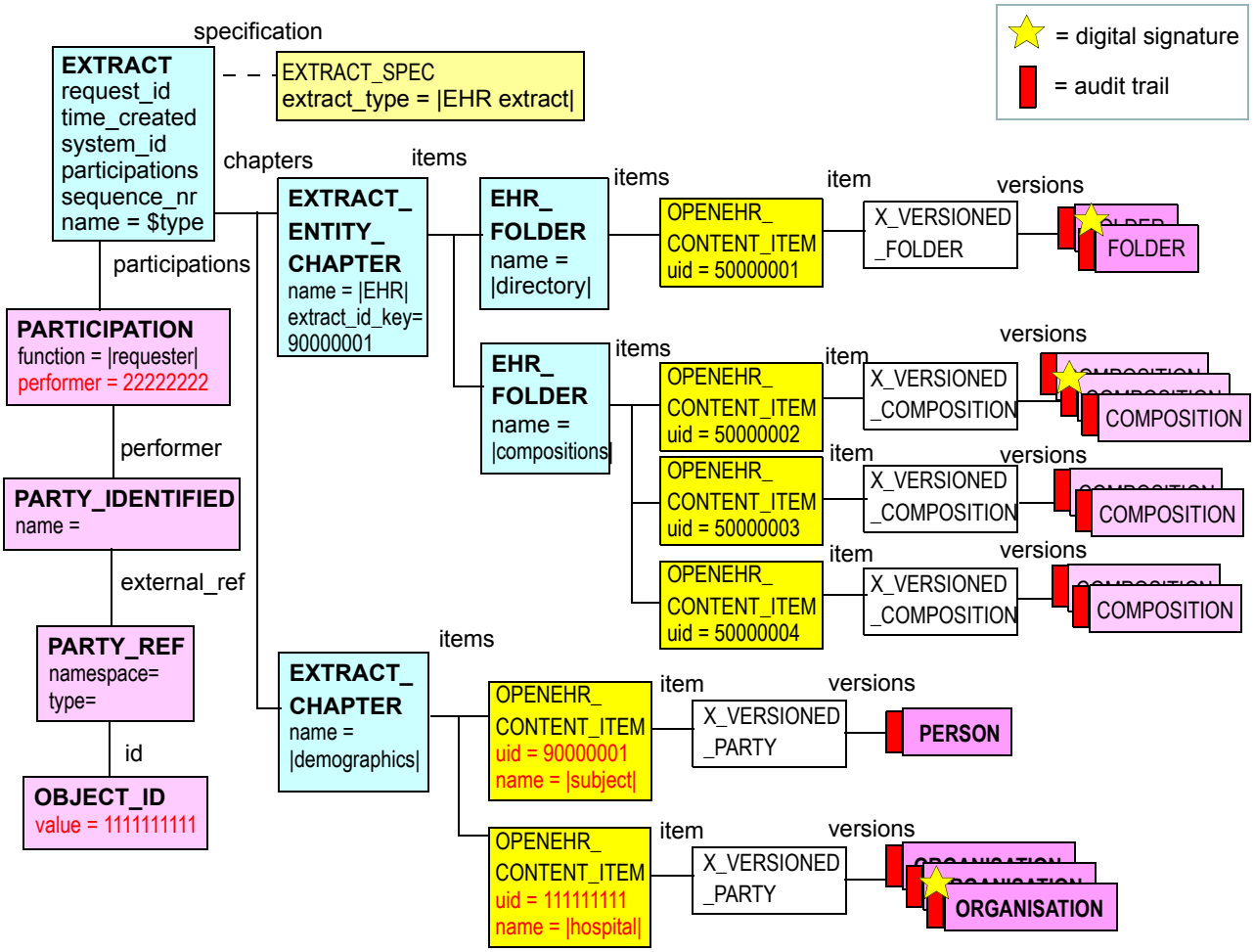


FIGURE 10 Typical EHR Extract structure

be included form the original. The revision history can be requested on its own using by setting the *includes_data* flag of the version specification to False.

In most scenarios, *versions* will be included, and *revision_history* excluded. Each item in `X_VERSIONED_OBJECT.`*versions* consists of a wrapped copy of an `ORIGINAL_VERSION` from the corresponding `VERSIONED_OBJECT` object in the source system.

EHR Extract Structure

The overall structure of the *open*EHR EHR for a subject can be replicated with an archetyped Folder structure that creates the groupings of *directory*, *compositions*, *ehr_access*, and *ehr_status* that are found in an *open*EHR EHR. This use of folders is not mandatory, but is likely to be useful in most cases, and is the structure shown in FIGURE 10 above.

Demographic Referencing

Within the content an EHR Extract, wherever `PARTICIPATION` and `PARTY_PROXY` structures occur (for the latter, the `PARTY_SELF` and `PARTY_IDENTIFIED` descendants), they are preserved intact, for reasons of fidelity. However, the final 'pointer', i.e. the `OBJECT_ID.`*value* may be rewritten within the Extract to correctly refer to the intended demographic. A typical `S_PARTICIPATION` structure is illustrated at the bottom of FIGURE 10. In this figure, the `OBJECT_ID.`*value* has been rewritten to refer to the uid (11111111) of the first OPENEHR_CONTENT_ITEM object shown in FIGURE 10 above.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

5.3 Class Descriptions

5.3.1 OPENEHR_CONTENT_ITEM Class

CLASS	OPENEHR_CONTENT_ITEM	
Purpose	Form of EHR EXTRACT_ITEM containing *open*EHR serialised VERSIONED_OBJECTs.	
Inherit	EXTRACT_CONTENT_ITEM	
Attributes	**Signature**	**Meaning**
0..1 (redefined)	**item**: X_VERSIONED_OBJECT<T>	Versioned *open*EHR content object.
Invariants		

5.3.2 X_VERSIONED_OBJECT Class

CLASS	X_VERSIONED_OBJECT <T: LOCATABLE>	
Purpose	Variety of Extract content that consists is a sharable data-oriented version of VERSIONED_OBJECT<T>.	
Attributes	**Signature**	**Meaning**
1..1	**uid**: HIER_OBJECT_ID	Uid of original VERSIONED_OBJECT.
1..1	**owner_id**: LOCATABLE_REF	Owner_id from original VERSIONED_OBJECT, which identifies source EHR.
1..1	**time_created**: DV_DATE_TIME	Creation time of original VERSIONED_OBJECT.
1..1	**total_version_count**: INTEGER	Total number of versions in original VERSIONED_OBJECT at time of creation of this X_VERSIONED_OBJECT.
1..1	**extract_version_count**: INTEGER	The number of Versions in this extract for this Versioned object, i.e. the count of items in the *versions* attribute. May be 0 if only revision history is requested.
0..1	**revision_history**: REVISION_HISTORY	Optional revision history of the original VERSIONED_OBJECT. If included, it is the complete revision history.
0..1	**versions**: List <ORIGINAL_VERSION<T>>	0 or more Versions from the original VERSIONED_OBJECT, according to the Extract specification.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	X_VERSIONED_OBJECT <T: LOCATABLE>
Invariants	***Uid_valid***: uid /= Void ***Owner_id_valid***: owner_id /= Void ***Time_created_valid***: time_created /= Void ***Total_version_count_valid***: total_version_count >= 1 ***Extract_version_count_valid***: extract_version_count >= 0 ***Versions_valid***: versions /= Void **implies not** versions.is_empty

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

6 Generic_extract Package

6.1 Overview

The `rm.extract.generic_extract` package defines a kind of Extract designed to be used by non-*open*EHR systems (including EHR/EMR systems) that want to send data to another system using *open*EHR structures defined by archetypes and templates. Such systems typically do not natively contain standardised data of any kind, and usually have quite variable and idiosyncratic support for versioning. FIGURE 11 illustrates the `rm.extract.generic_extract` package.

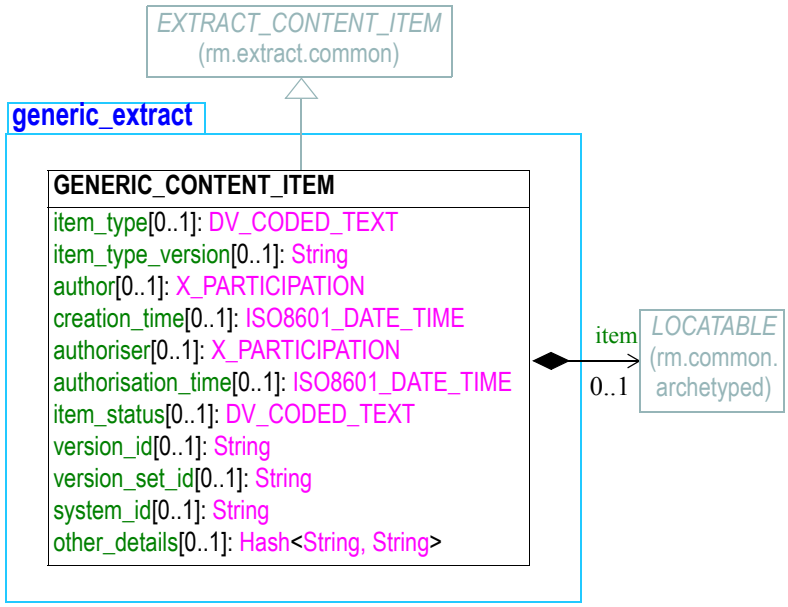


FIGURE 11 rm.extract.generic_extract Package

FIGURE 12 illustrates the instance structure of a typical `GENERIC_EXTRACT`. The containment structure is shown in blue, and content item wrappers (meta-data) in yellow.

6.2 Design

6.2.1 Structure

The `GENERIC_CONTENT_ITEM` subtype of `EXTRACT_CONTENT_ITEM` defines meta-data more or less of which can be populated by typical legacy systems, as well as document sources such as IHE/XDS repositories. Various groups of meta-data are defined: document type / status information, commit audit details and document lifecycle information.

The document meta-data items are as follows:

- *item_type*: a coded term indicating the content 'model' or schema, which may be a published standard, and/or some kind of archetype, template or similar content control artefact;
- *item_type_version*: version of the *item_type* model.

The next group of items defines details of creation of the original information. In some systems, this is the same as 'committal', i.e. the act of creating and committing the original data to the system. In

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

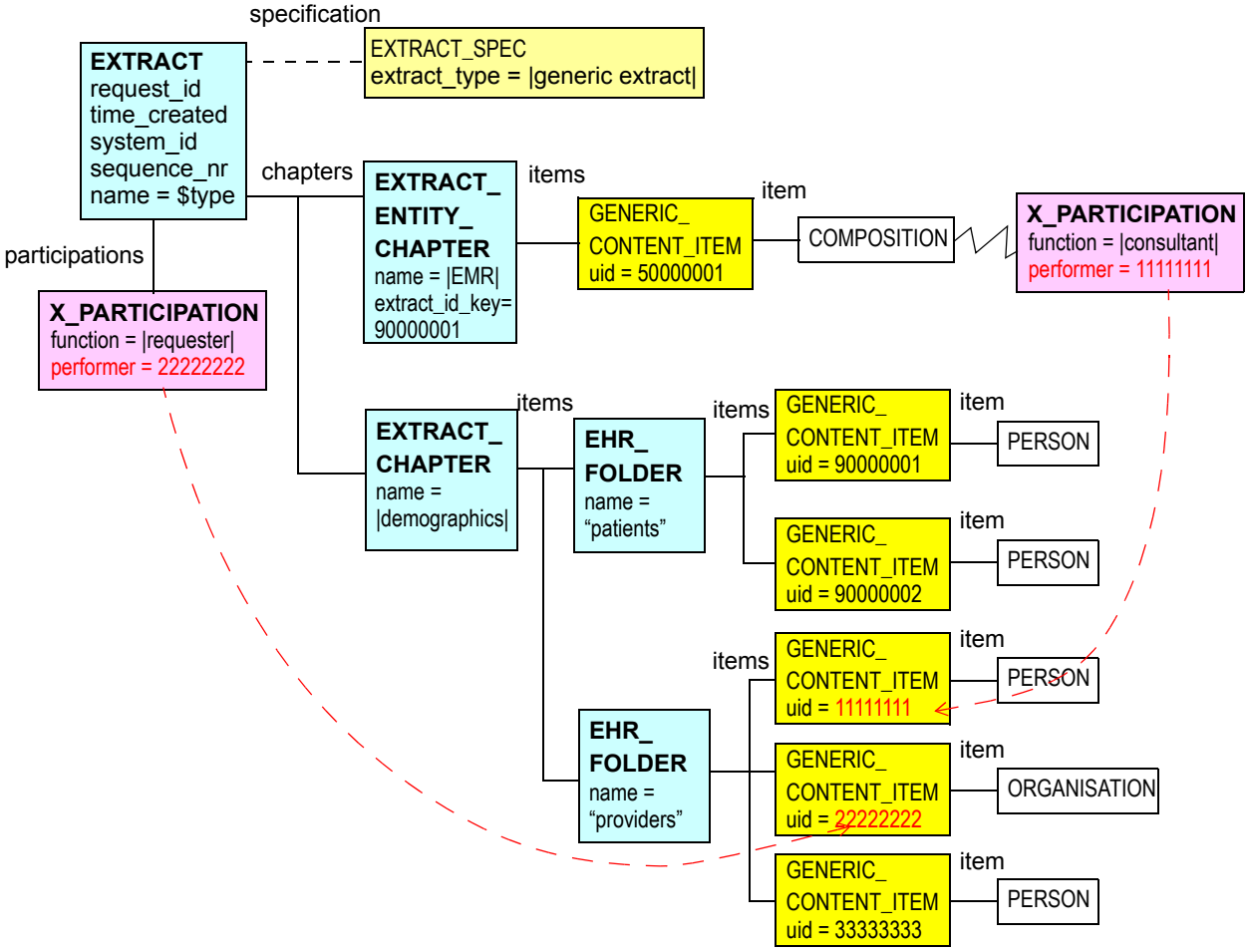


FIGURE 12 Typical generic extract structure

other cases, the information in the extract is being assembled for the first time for the purposes of the extract, and the audit information therefore corresponds to this act. The audit details are as follows:

- *author*: identity of the author of the original content - this is a UID reference to a demographic item in another part of the current extract;
- *creation_time*: date/time when the content was created;
- *authoriser*: identity of a professional who authorised this content, if relevant - this is a UID reference to a demographic item in another part of the current extract;
- *authorisation_time*: date/time when the content was authorised;
- *item_status*: lifecycle status of the content;
- *version_id*: version of this particular instance of the content item when it was created and/or committed, in environments where versioning is supported;
- *version_set_id*: 'version set' identifier, i.e. identifier of group of versions which together constitute a set of versions in time of a given logical content item;
- *system_id*: system where the content was created / committed to / extracted from;
- *other_details*: other meta-data, in the form of a keyed list of Strings.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

All of these items are optional, reflecting their varying availability in different source systems. An archetype or template of the Extract may force some or all to be mandatory for specific kinds of Extract.

The *item* attribute carries content which may be any *open*EHR LOCATABLE structure, including things like COMPOSITION, any kind of ENTRY including GENERIC_ENTRY, ITEM_TREE, or even CLUSTER. This is assumed to have been converted by some data transformation process from data within the original system.

Demographic Referencing

Within the content a generic Extract, participations are usually expressed using the X_PARTICIPATION variant (which is a descendant of the PARTICIPATION class defined in the rm.common.generic package). This is because the *open*EHR structures (such as COMPOSITION etc) used within the Extract are not copies of openEHR data, but constructed on the fly for the purposes of the Extract. In this situation, there is no need to use the more complex S_PARTICIPATION structure.

6.3 Class Descriptions

6.3.1 GENERIC_CONTENT_ITEM Class

CLASS	GENERIC_CONTENT_ITEM	
Purpose	Single item in generic extract.	
Inherit	EXTRACT_CONTENT_ITEM	
Attributes	**Signature**	**Meaning**
0..1	**item**: LOCATABLE	Content item.
0..1	**item_type**: DV_CODED_TEXT	Identifier of model or schema used to create the content.
0..1	**item_type_version**: String	Version of model or schema used to create the content item.
0..1	**author**: String	Reference to a demographic entity elsewhere in this Extract representing the author of the item version. The reference should be a UID corresponding to the UID of a GENERIC_CONTENT_ITEM containing the demographic information.
0..1	**creation_time**: ISO8601_DATE_TIME	Time of creation of this item version on the original system. This may be an earlier commit time, or it may be the time at which the item was created during the Extract generation process.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	GENERIC_CONTENT_ITEM	
0..1	**authoriser**: `String`	Reference to a demographic entity elsewhere in this Extract representing an authoriser of the item version, if relevant. The reference should be a UID corresponding to the UID of a `GENERIC_CONTENT_ITEM` containing the demographic information.
0..1	**authorisation_time**: `ISO8601_DATE_TIME`	Time of authorisation of this item version on the original system where relevant.
0..1	**item_status**: `DV_CODED_TEXT`	Coded lifecycle status of the item.
0..1	**version_id**: `String`	Version id of this item in original system.
0..1	**version_set_id**: `String`	Version set id of this item in original system, where applicable.
0..1	**system_id**: `String`	Identifier of EMR or other system from which the item was created / extracted. Typically in the form of a domain name.
0..1	**other_details**: `Hash<String, String>`	Other details about the content item.
Invariants	***Item_type_valid***: /= Void ***Item_type_version_valid***: item_type_version /= Void **implies not** item_type_version.is_empty ***Author_valid***: author /= Void **implies not** author.is_empty ***Authoriser_valid***: authoriser /= Void **implies not** authoriser.is_empty ***Version_id_valid***: version_id /= Void **implies not** version_id.is_empty ***Version_set_id_valid***: version_set_id /= Void **implies not** version_set_id.is_empty ***System_id_valid***: system_id /= Void **implies not** system_id.is_empty ***Other_details_valid***: other_details /= Void **implies not** other_details.is_empty	

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

7 Synchronisation Extracts

TBD_1: this section still be to be reviewed

7.1 Overview

FIGURE 13 illustrates an Extract variant designed for synchronising two *open*EHR systems. The specification only allows for a list of Contributions, or Contributions since a certain Contribution; it also allows the actual versions to be included or excluded. If they are excluded, you can get just Contributions on their own - i..e find out what the other system has got.

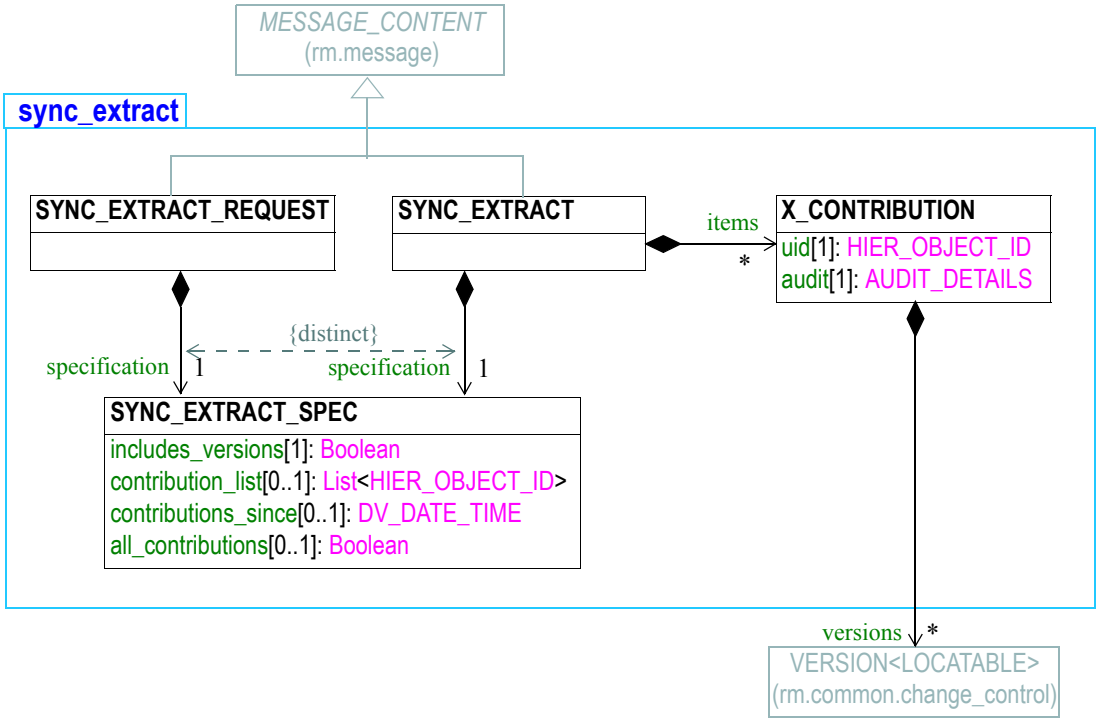


FIGURE 13 rm.extract.synchronisation_extract Package

7.2 Class Descriptions

7.2.1 SYNC_EXTRACT_REQUEST Class

CLASS	SYNC_EXTRACT_REQUEST	
Purpose	Type of request designed for synchronisation of Contributions between *open*EHR servers.	
Inherit	MESSAGE_CONTENT	
Attributes	**Signature**	**Meaning**
1..1	**specification**: SYNC_EXTRACT_SPEC	Details of specification of synchronisation request.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	SYNC_EXTRACT_REQUEST
Invariants	***Specification_valid***: specification /= Void

7.2.2 SYNC_EXTRACT Class

CLASS	SYNC_EXTRACT	
Purpose	Synchronisation Extract.	
Inherit	`MESSAGE_CONTENT`	
Attributes	Signature	Meaning
1..1	**specification**: `SYNC_EXTRACT_SPEC`	Details of specification of this Extract.
1..1	**items**: `List<X_CONTRIBUTION>`	Content, in the form of a serialised Contributions.
Invariants	***Specification_valid***: specification /= Void	

7.2.3 SYNC_EXTRACT_SPEC Class

CLASS	SYNC_EXTRACT_SPEC	
Purpose	Details of specification of Extract, used in a request to specify an Extract, or in a response, to describe what is actually in the Extract.	
Attributes	Signature	Meaning
1..1	**includes_versions**: `Boolean`	True if the Versions from the Contribution are included; False if just the Contribution and its Audit are included.
0..1	**contribution_list**: `List<HIER_OJBECT_ID>`	List of Contributions to include / that are included in the Extract.
0..1	**contributions_since**: `DV_DATE_TIME`	Specify Contributions included in Extract by threshold date.
1..1	**all_contributions**: `Boolean`	True if all Contributions in the record are included.
Invariants		

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

7.2.4 X_CONTRIBUTION Class

CLASS	X_CONTRIBUTION	
Purpose	Serialised form of Contribution for an Extract.	
Attributes	**Signature**	**Meaning**
1..1	**uid**: `HIER_OBJECT_ID`	Uid of Contribution in source system.
1..1	**audit**: `AUDIT_DETAILS`	Audit of Contribution in source system.
0..1	**versions**: `List<VERSION<T>>`	Serialised Versions from Contribution in source system.
Invariants	***Uid_valid***: uid /= Void ***Audit_valid***: audit /= Void ***Versions_valid***: versions /= Void **and then not** versions.is_empty	

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

8 The Message package

TBD_2: this section still to be reviewed

8.1 Requirements

In the first two EHR extract scenarios described in Requirements on page 9, extracts may be received in response to a request, or they may be unsolicited. Most transfers of care (e.g. discharge summaries and referrals) and pathology test results will generate unsolicited extracts, whereas solicited requests will usually occur due to the patient presenting him or herself in another part of the health system without an explicit transfer of care.

8.2 Design

The `message` package provides the basic abstractions for the sending and receiving of any point to point message containing a payload, of abstract type `MESSAGE`. The Message Package is illustrated in FIGURE 14.

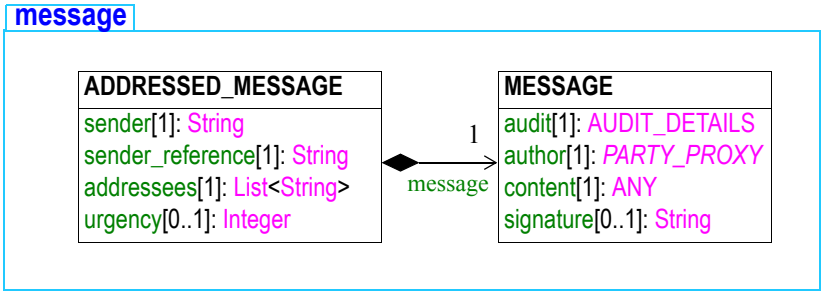


FIGURE 14 rm.message Package

A new message is required for each transmission, even if the payload was created once and is retransmitted multiple times.

Integrity and Security

The `MESSAGE` object may include a digital hash (i.e. digest or digital finderprint) of the serialised content, made for example using the SHA-1 or MD5 algorithms. The purpose of the digest is to provide an integrity check on the data. This protects against non-malicious changes to the data (e.g. due to software bugs, incorrect transaction management). Often this will be acceptable within secure, closed environments, such as a private hospital or community health network.

Protection against malicious modification can be provided by encryption.

To Be Continued: normalised serialised expression

8.3 Class Descriptions

8.3.1 ADDRESSED_MESSAGE Class

CLASS	ADDRESSED_MESSAGE
Purpose	The concept of a message addressed to nominated recipients.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	ADDRESSED_MESSAGE	
CEN	MESSAGE	
HL7	various message classes	
Attributes	**Signature**	**Meaning**
1..1	**sender**: String	Party sending the message.
1..1	**sender_reference**: String	Identification of message used by sender. This will be the same no matter how many times this message is sent to these recipients.
1..1	**addressees**: List<String>	Intended recipients, in the form of internet addresses.
0..1	**urgency**: Integer	Urgency with which destination should deal with message: -1 - low 0 - normal 1 - high
1..1	**message**: MESSAGE	The content of the message.
Invariants	***Sender_valid***: sender /= Void **and then not** sender.is_empty ***Addressees_valid***: addressees /= Void **and then not** addressees.is_empty ***Sender_reference_exists***: sender_reference /= Void **and then not** sender_reference.is_empty ***Message_exists***: message /= Void	

8.3.2 MESSAGE Class

CLASS	MESSAGE	
Purpose	A "message" is an authored, possibly signed, piece of content intended for one or more recipients. Since the recipient may or may not be known directly, recipients are specified in the ADDRESSED_MESSAGE class.	
Attributes	**Signature**	**Meaning**
1..1	**audit**: AUDIT_DETAILS	Details of who actually created the message- and when. This is the person who entered the data or otherwise caused the message to be created, or might be a piece of softare.
1..1	**author**: PARTY_PROXY	Party responsible for the message content, who may or may not be technically responsible for its creation (e.g. by data entry etc).

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

CLASS	MESSAGE	
1..1	**content**: `ANY`	Content of the message.
0..1	**signature**: `String`	Optional signature by the *author* of message content in openPGP format. The signature is created as a Hash and optional signing of the serialisation of this message object with this signature field Void.
Invariants	***Author_valid***: author /= Void ***Audit_valid***: audit /= Void ***Content_valid***: content /= Void	

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

9 Use Case Examples

9.1 Single Patient Discharge Summary

The generation of a discharge summary, referral or other typical extract of information for a patient from an EMR environment can be handled with the use of the generic Extract type. The content would be specified by archetypes for the following:

- the Extract / Chapter / Folder structure;
- the clinical content for the patient, typically based on standard archetypes for diagnosis, procedure, history, lab etc;
- the demographic entities mentioned in the clinical content.

Slots can be used within the Extract structure archetype, and t

9.2 Medico-legal Investigations

It is currently believed that access to prior versions will only take place for reasons of medico-legal investigations, and that this will normally occur *in situ*, i.e. at the relevant HCF, and require special legal intervention. The practical consequence of this is that only latest versions of `VERSIONED_COMPOSITIONs` are normally sent in `EHR_EXTRACTs`. However, in the case of a medico-legal investigation, earlier `VERSION<COMPOSITION>s` may be sent in an extract. `VERSIONED_COMPOSITIONs` are never sent in `EHR_EXTRACTs`, but might be sent in a situation where the entire EHR is changing custodianship, e.g. if the patient moves to another GP, or another country.

9.3 Transfer of Entire EHR

There are two possible ideas of transferring an "entire EHR". The first is to satisfy a request for "the latest cut" of a patient's entire EHR (or even perhaps the entire snapshot for some earlier moment in time). This scenario is simply a special case of the normal request/extract scenario in which the following conditions are true.

- The set of Compositions requested just happens to be all of the existing ones.
- There is no guarantee that all the requested compositions will be incorporated into the receiver’s EHR for the patient in question - some may be discarded as irrelevant, or out of date.
- Both the sending and receiving EHR systems will continue to create and modifiy their EHRs according to independent processes.
- In general the sending and receiving systems’ versions of any given EHR will diverge in time due to these processes - there is no a priori assumption that the two EHRs must remain synchronised.

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

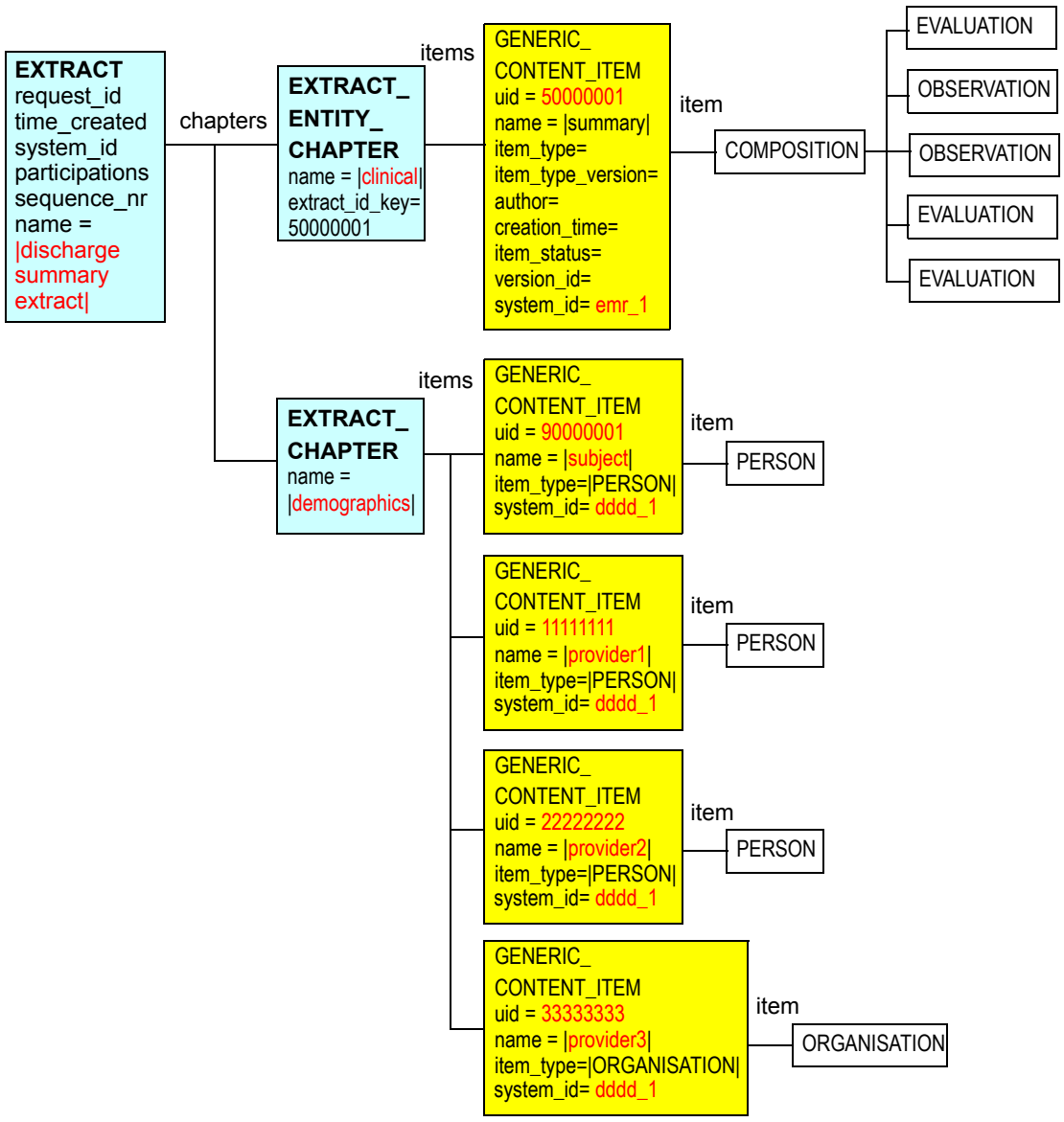


FIGURE 15 Single patient discharge summary

Apart from the first one, these conditions are exactly the same as for the normal communication scenarios, and are dealt with by these scenarios.

The second scenario corresponds to a *change of custodianship* of an EHR, in which case the following is true:

- the whole EHR including all its versions, contributions, entire folder structure, all relevant demographic, terminological, access control and presumably all referenced patient-specific information (such as images, executing guidelines etc) stored on departmental systems is transferred to a new place;
- the old EHR is then decommissioned, archived, and possibly removed from the online system;

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

- the EHR in its new location (and the patient) become the responsibility of a new health care facility and/or information custodians, and is subject to what ever information governance is in place in the new location;
- all medico-legal responsibility passes to the new custodian, requiring that all previous versions in time be retained.

As far as is known, there is no solid experience showing what the generally accepted requirements for transfer-of-custodianship are. It is currently thought that the exact definition of what needs to be transferred in the second scenario could be complex and dependent upon local or regional particularities.

Further, it is thought that at the technical level, transfer of "entire EHRs" might well be accomplished by a variety of means in any particular circumstance, such as:

- physical movement of computer system;
- physical movement of binary or database files of some kind;
- low-level dump of EHR at origin and restore of EHR at receiver, assuming same/compatible database systems;
- use of binary transfer protocols e.g. CORBA, .Net etc.

At this point, it is hard to show that the operation to re-establish an EHR in its entirety in another place can be described by a clear and generally accepted set of requirements which could be formally modelled. Consequently, the specifications provided here do not claim to satisfy any particular scenario of this kind, although it is conceivable that they could be used to enact it in some particular situations, depending on the needs. Further work is required to determine what additional features might be needed in the proposed models to satisfy the EHR transfer scenarios in different countries, jurisdictions etc.

To Be Continued:

9.4 Single Hop

To Be Continued:

9.5 Multiple Hop

To Be Continued:

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

10 Semantics of *open*EHR and ISO 13606 extracts

10.1 Versioning Semantics

Although for most clinical situations, it is the latest versions of Compositions which are sent to a receiver, there are requirements for various amounts of version-related information to be included, as described in Requirements on page 9. At a minimum, Compositions always include the audit trail corresponding to the particular version which the Composition represents. In some cases, historical versions of a logical Composition are needed for some medico-legal reason. It may even be required that the receiver system wants to reconstruct a complete facsimile of the versioned object, logically identical to its form at the source (but most likely stored in a different versioning implementation).

The *open*EHR extract specification defines the simplest means of satisfying these needs, namely to include all Compositions in their whole form, including in the case where they are successive versions of a single logical Composition such as "family history", as illustrated in FIGURE 16. The main justification for this is that no assumptions should made on sender or receiver systems to do with their ability to represent or efficiently process versions. Whole Compositions can always be processed by even the simplest systems.

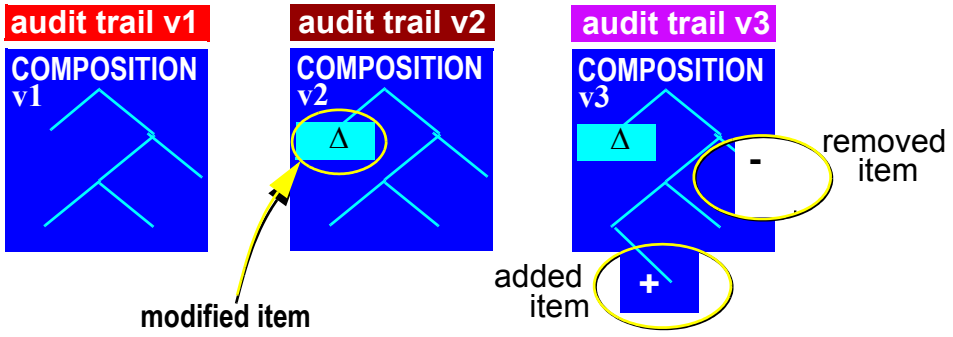


FIGURE 16 Successive Composition versions in a logical Transaction

It is assumed that any system that wants to be able to determine things such as who was responsible for changing a certain fragment of a Composition, when some part of a Composition came into being, or the differences between two particular versions of a Composition, must have version control capability locally. This usually means having some implementation of a version control model such as the one described in the *open*EHR Common Reference Model, which can do efficient versioning, differencing and so on. Supplying Compositions in their full form ensures that no assumption is made on what such an implementation might be.

This approach is a departure from the ISO 13606-1:2008 EHR Extract standard, which defines Compositions so as to include revision history information on every node of the structure. Although it is not stated in the 13606 specification whether the 'Composition' is in fact supposed to be understood as a copy of a Composition from an EHR, or as a 'cumulative diff' of Composition versions in an EHR, analysis shows that only the latter can make sense because the Composition (Composition) is the unit of creation and modification, and there is logically only one audit trail for each version. Even the 100th version has associated with it only one audit trail.

This raises the question of whether a 'diff' form of Compositions should be used in the *open*EHR Extract, conforming to the ISO standard. The approach was not chosen for a number of reasons:

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

- it implies that senders can generate 'diff' information structures and that receivers can process them, i.e. it makes more assumptions than necessary about the sophistication of systems;
- the ISO specification appears to be in error with respect to deletions - the sending of logical deletions does not appear to be handled properly;
- the sending of deletions is not normally desired, and may be illegal (e.g. in Europe there are EC directives preventing the sending of statements corrected by clinicians or patients).

It is worth contemplating just how complex cumulative difference information would be. FIGURE 17 illustrates the structure generated by the accumulation of only three changes shown in the successive versions in FIGURE 16. The large numbers of changes likely in persistent Compositions will generate far more complex structures.

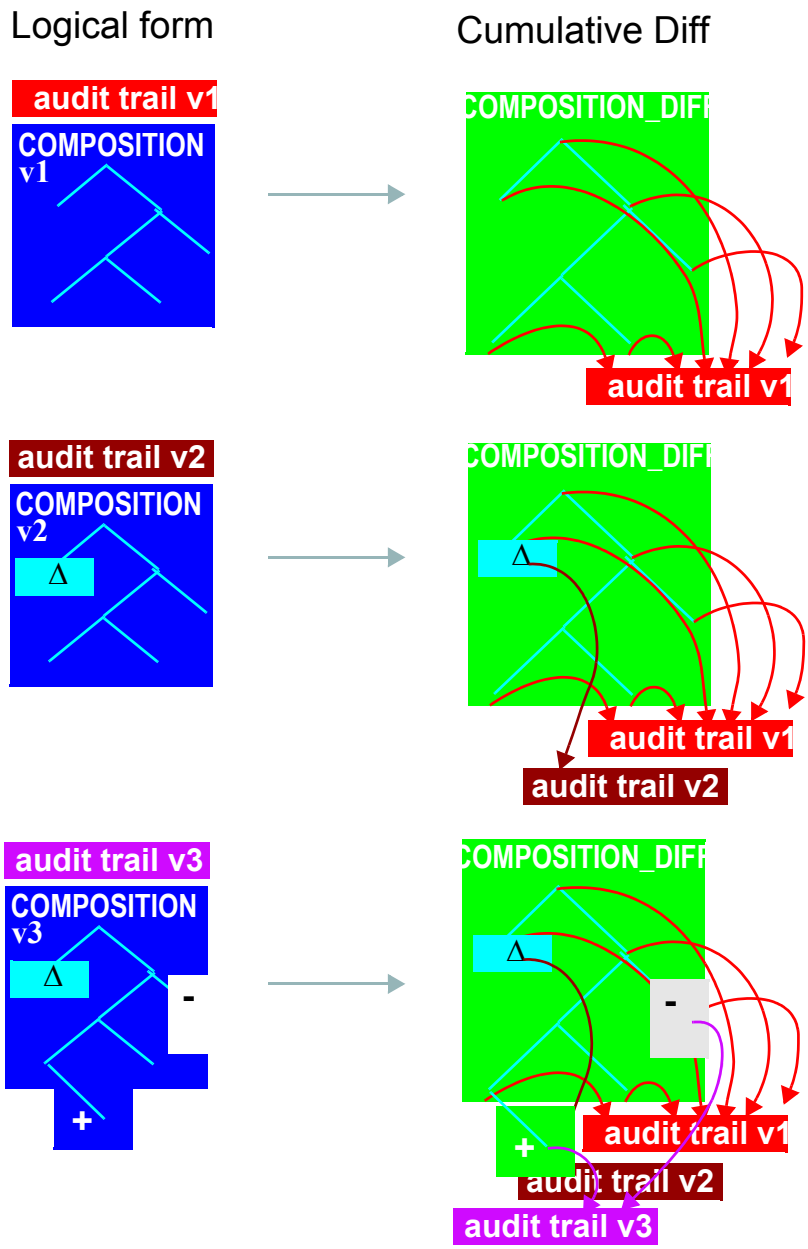


FIGURE 17 Generation of Cumulative Difference Form

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

In conclusion, while sending a difference form of Compositions is not out of the question in a future when EHR systems are routinely capable of sophisticated version handling, it is considered too complex currently, and the controls over sending deleted information have not been sufficiently well described.

10.2 Creation Semantics

The following describes an algorithm which guarantees the correct contents of an EHR extract. The input to the algorithm is:

- the list of EHR Compositions required in the extract (the "primary" Composition set);
- optionally a folder structure in which the Compositions are to be structured in the extract;
- the *include_multimedia* flag indicating whether DV_MULTIMEDIA content is to be included inline or not;
- the *follow_links* attribute indicating to what depth DV_LINK references emanating from Compositions should be followed and the Compositions containing the link targets also included in the extract.

The algorithm is as follows.

- Create a new EHR_EXTRACT including the folder structure;
- Create a demographics EXTRACT_CHAPTER and write the PARTYs in;
- For each Composition in the original set, do:
  - create an X_VERSIONED_COMPOSITION, and set *is_primary*;
  - for each instance of OBJECT_REF encountered (e.g. PARTY_REF), obtain the target of the reference from the relevant service, and copy it to the appropriate chapter, e.g. *demographics*, *access_groups* tables with the key = the OBJECT_REF.*id*;
  - copy/serialise the Composition into the appropriate place in the folder structure rewriting its OBJECT_REFs so that namespace = "local"
  - for each instance of DV_MULTIMEDIA encountered, include or exclude the content referred to by the *uri* or *data* attributes, according to the *include_multimedia* flag;
  - according to the value of *follow_links*, for each instance of DV_LINK encountered (only from/to Archetyped entities):
    * follow the links recursively. For each link: create an X_VERSIONED_COMPOSITION; set *is_primary* = False, write the path and write the target Compositions in the extract if not already there;
    * create the DV_LINK objects so that their paths refer correctly to the Compositions in the Extract;

TBD_3: do something about Access_control objects;

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org

END OF DOCUMENT

© 2003-2007 The openEHR Foundation.
email: info@openEHR.org web: http://www.openEHR.org