REFERENCE MODEL

The *open*EHR EHR_EXTRACT Information Model

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

Revision: 1.3.5

Pages: 29

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

The *open*EHR foundation

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

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

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

Copyright Notice

Amendment Record

Issue	Details	Who	Completed
1.3.5	CR-000118. Make package names lower case.	T Beale	10 Dec 2004
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

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.

Table of Contents

1 Introduction

1.1 Purpose

This document describes the architecture of the *open*EHR EHR Extract Information Model. This model is equivalent in scope to the CEN ENV 13606:2000 part 4 standard.

The intended audience includes:

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

1.2 Related Documents

Prerequisite documents for reading this document include:

- The *open*EHR Modelling Guide
- The *open*EHR 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/repositories/spec-dev/publishing/architecture/reference_model/ehr_extract/REV_HIST.html. New versions are announced on openehr-announce@openehr.org.

1.4 Peer review

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

To Be Continued: more work required

Reviewers are encouraged to comment on and/or advise on these paragraphs as well as the main content. 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.

2 Background

This section describes the inputs to the modelling process which created the *open*EHR Reference Model.

2.1 Requirements

To Be Continued:

2.2 Design Principles

To Be Continued:

3 Requirements

3.1 Basic Scenarios

Health information is sent to EHR systems for a number of reasons, and in a variety of forms. The scenarios for moving primary data (i.e. data already in existence as opposed to generated reports) in and out of an EHR system are as follows:

1. Extracts are received from other EHR systems, where the extract contains some part of, or in some cases all of some version (usually the latest) of an EHR;
2. Messages such as HL7 messages are received from non-EHR systems, and are converted to EHR extracts;
3. Messages such as HL7 messages are received from non-EHR systems and are converted directly into EHR content by the receiving system;
4. The EHR system emits messages or documents for use by non-EHR systems.
5. A whole EHR is *moved* from one EHR system to another corresponding to a change of custodianship (see Transfer of Entire EHR on page 26);

These scenarios are illustrated in FIGURE 1.

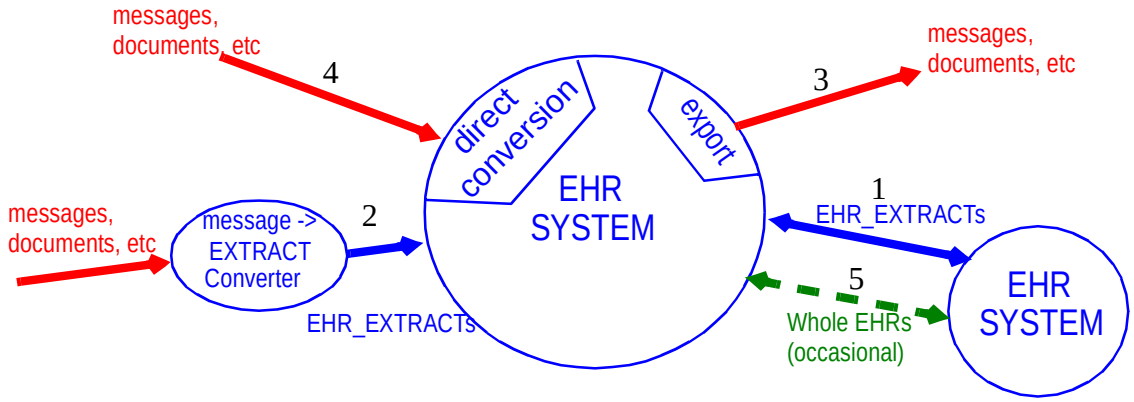

FIGURE 1 EHR Communication Scenarios

The following sections deal with the semantics of the information transmitted in scenarios 1. and 2., i.e. EHR extracts, which are the primary mechanism for transmitting selected parts of EHRs between EHR systems. The semantics of scenarios 3 and 4 are dealt with in the document describing the *open*EHR EHR service model. Scenario 5 is described separately below.

4 The Message package

4.1 Requirements

In the first two EHR extract scenarios described in Requirements on page 11, 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.

4.2 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_CONTENT`. The Message Package is illustrated in FIGURE 2.

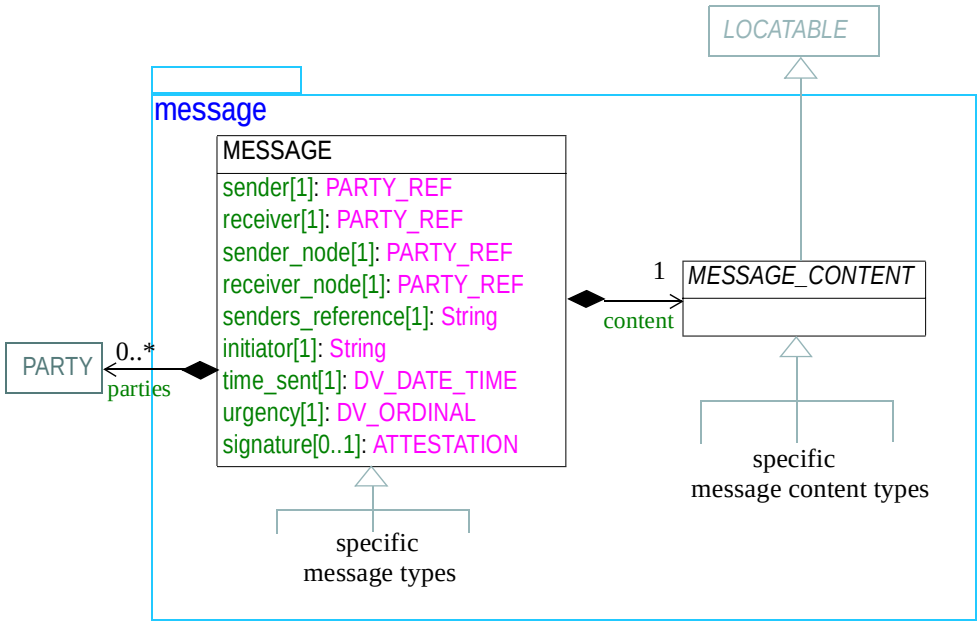

FIGURE 2 rm.message Package

A new message is required for each transmission, even if the payload was created once and is retransmitted multiple times. All demographic entities are included by value in the *parties* attribute.

To Be Continued: more investigation of CEN 13606 part 3, 4 required here.

4.3 Class Descriptions

4.3.1 MESSAGE Class

CLASS	MESSAGE	
Purpose	The message envelope for an extract, indicating the sender and receiver details, time and any other details required.	
CEN	MESSAGE	
HL7	various messages classes	
Attributes	**Signature**	**Meaning**
	time_sent: DV_DATE_TIME	Date/time the message was sent.
	sender: PARTY_REF	Party sending the extract.
	receiver: PARTY_REF	Party the extract is sent to.
	sender_node: PARTY_REF	EHR node from which the message is sent.
	receiver_node: PARTY_REF	EHR node receiving the message.
	senders_reference: String	Identification of message at sender's end.
	initiator: String	Indicates which party - sender or receiver caused the message to be created and sent. If the receiver (initiator = "R"), there was an EHR_REQUEST. If the sender (initiator = "S"), there is no request, and the extract is being sent unsolicited.
	urgency: DV_ORDINAL	Urgency with which receiver should deal with message
	signature: ATTESTATION	Signature of message content.
	parties: Set<PARTY>	Parties referred to by all PARTY_REF and ATTESTATION instances in this message instance.
	content: MESSAGE_CONTENT	The content of the message.

CLASS	MESSAGE
Invariants	***Time_sent_exists***: time_sent /= Void ***Sender_exists***: sender /= Void ***Receiver_exists***: receiver /= Void ***Senders_reference_exists***: servers_reference /= Void ***and then not*** senders_reference.is_empty ***Sender_node_exists***: sender_node /= Void ***Receiver_node_exists***: receiver_node /= Void ***Urgency_exists***: urgency /= Void ***Initiator_valid***: initiator /= Void ***and then*** (initiator = "S" ***or else*** initiator = "R") ***Parties_valid***: Parties /= Void ***implies not*** parties.is_empty ***Content_exists***: content /= Void

4.3.2 MESSAGE_CONTENT Class

CLASS	*MESSAGE_CONTENT (abstract)*	
Purpose	Abstract supertype for any message content.	
CEN	???	
HL7		
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
Invariants	***Is_archetype_root***: is_archetype_root	

5 Ehr Extract Package

5.1 Requirements

The requirements for EHR Extracts are driven by the need for systems to communicate certain versions of Compositions, arranged under a Folder structure. For Compositions (the main content of interest in the extract), the needs are:

1. an extract contains *requested versions* of Compositions, e.g.:
   - the *latest versions* of *nominated* Compositions (expected to be the scenario in the vast majority of cases for communication for clinical care)
   - the *latest version of all Compositions* in the record (i.e. a current snapshot of the record)
   - *some or all historical versions* of one or more nominated Compositions
   - *all historical versions of all Compositions* (i.e. the entire record including all historical states). The use case for the transfer of a whole EHR corresponding to the transfer of management responsibilty/legal custody is described in Transfer of Entire EHR on page 26.
2. a potential requirement in EHR systems to be able to determine what part of a Composition changed in a given update or correction
3. a potential requirement in EHR systems to be able to determine what audit information is associated with a particular part of a Composition

These latter two requirements drive the need for sending historical versions of a given logical Composition.

For Folders, the needs appear to include the following possibilities:

1. no Folders at all
2. a copy of some part of a Folder tree from an EHR, e.g. containing a set of Compositions of interest.
3. the entire folder tree from the source EHR, if the entire EHR contents are requested
4. a newly created Folder tree for the purpose of organising Compositions specifically for the Extract
5. a mixture of newly created and copied Folder structures.

Accordingly, this specification assumes that no meaningful versioning information for Folders can be included in Extracts, since in general, only pieces of the original Folder structure, if any, will be included, along with non-versioned new Folder structures created specifically for the Extract.

The remaining requirements are consequences of needing to supply enough information referenced by Compositions in order to ensure the integrity of the information at the receiver's end. To this end, the following further things are needed in an Extract:

1. sufficient *demographic* data for each demographic entity referred to by a Composition
2. *attestations* associated with each version of a Composition
3. potentially additional *terminological* data (such as a terminology extract) to ensure receivers can process all terms in an extract
4. relevant *access control* information

5.2 Design

The `ehr_extract` package is designed to fulfill these requirements. It is illustrated in FIGURE 3.

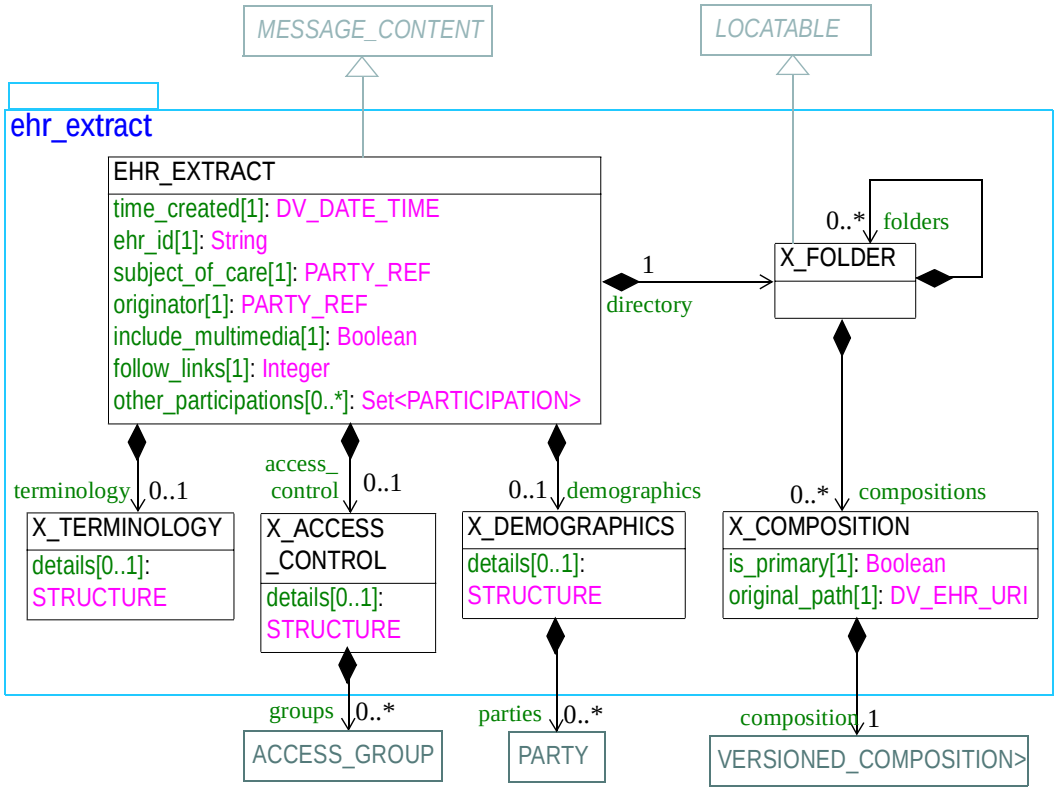

FIGURE 3 rm.ehr_extract Package

The content of an EHR extract consists of the following:

- copies of a selection of Compositions, optionally within a folder structure;
- a folder structure which may include folder sub-trees from the source EHR, and/or folder trees created during the extraction process (e.g. corresponding to a discharge summary structure or similar). These folder structures may potentially be archetyped;
- copies of all entities from other services referenced from the EHR, including demographic entities (Parties) and access control entities (Access_groups);
- any extra information required for the receiver to understand the extract, potentially including terminology extracts (e.g. in the form of <key, rubric> tables).

None of the other services (particularly demographics, access control etc) in the sender's environment is assumed to be available in the receiver's environment; consequently, in general, referenced entities must be included. However, an extract can be constructed in such a way as to leave out such entities, for the case when both the receiver and sender nodes exist within the same environment and have access to the same services. In any case, it is crucial to understand that the receiver's view of the extract must include *exactly the versions of all referenced entities* as existed when the extracted information was first created. In a shared environment, the use of versioned ids (subtypes of `RM.COMMON.IDENTIFICATION.OBJECT_ID`) guarantees this.

In the case of unsolicited extracts, the structure of Folders and Compositions may be *ad hoc*, but would preferably follow archetyped models for "discharge summary", "discharge referral", "transfer of care" and other well-known documents in the health system. The structure of requested extracts is

more likely to be *ad hoc*, since requests will usually be in the form of a query, such as "all compositions between date_1 and date_2", or a list of persistent compositions, such as "'current medications', 'care plan', 'therapeutic precautions'". However, it may also be structured, if archetypes are developed for repeated requests; in this case, the request will simply identify the archetype model of the extract.

Extracts are characterised by the time of creation, the parties authorising, sending and receiving, and the "initiator", i.e. who caused it - the receiver (requested) or the sender (unsolicited).

To Be Continued: describe how PARTY_IDs, other EXTERNAL_IDs work inside an EHR_EXTRACT.

5.3 Class Descriptions

5.3.1 EHR_EXTRACT Class

CLASS	EHR_EXTRACT	
Purpose	The outer package for information extracted from an EHR by another EHR system.	
CEN	EHCR_EXTRACT	
Synapses	TBD	
GEHR	G1_EHR_EXTRACT	
Inherit	MESSAGE_CONTENT	
Attributes	**Signature**	**Meaning**
	time_created: DV_DATE_TIME	Date/time the extract was created
	ehr_id: String	Id of the EHR from which the extract was created.
	subject_of_care: PARTY_REF	Id of the subject of care to whom the extract relates.
	originator: PARTY_REF	Health care agent authorising the extract to be created and sent.
	other_participations: Set<PARTICIPATION>	Any other participations in the extract creation process
	include_multimedia: Boolean	Indicates whether multimedia content referred to by instances of DV_MULTIMEDIA is included or not.

CLASS	EHR_EXTRACT	
	follow_links: `Integer`	Degree of link following `DV_LINKs` emanating from Compositions originally determined to be required for the extract. All Compositions at the target end of followed links are also included in the extract. - 0 = don't follow; - 1 = follow first degree links; - 2 = follow 2nd degree links; - .... - n = follow nth degree links
	directory: `X_FOLDER`	Directory structure of this extract, in which Compositions are contained.
	terminology: `X_TERMINOLOGY`	List of full terminology identification details for all terms used in Compositions appearing in this extract.
	demographics: `X_DEMOGRAPHICS`	List of snapshots of all demographic entities referenced from the Compositions in this extract.
	access_control: `X_ACCESS_CONTROL`	List of all access_control settings relevant to the Compositions in this extract.
Invariants	***Originator_valid***: originator /= Void ***Other_participations_valid***: other_participations /= Void ***implies not*** other_participations.empty ***Directory_valid***: directory /= Void ***and*** directory.is_archetype_root ***Time_created_exists***: time_created /= Void ***Ehr_id_exists***: ehr_id /= Void ***and then not*** ehr_id.empty ***Subject_of_care_valid***: subject_of_care /= Void ***Directory_valid***: directory /= Void	

5.3.2 X_COMPOSITION Class

CLASS	X_COMPOSITION	
Purpose	Container for Composition in extract. Indicates whether it was part of the primary set and what it's original path was.	
Attributes	**Signature**	**Meaning**
	is_primary: `Boolean`	True if the Composition in this container was part of the primary set for the Extract , i.e. not added due to link-following.

CLASS	X_COMPOSITION	
	original_path: DV_EHR_URI	The original path of the Composition in the source EHR, used for matching compositions in the receiver's EHR.
	composition: VERSIONED_COMPOSITION	The COMPOSITION content.
Invariants	***Original_path_exists***: original_path /= Void ***Composition_exists***: composition /= Void	

5.3.3 X_FOLDER Class

CLASS	X_FOLDER	
Purpose	Folder in an extract.	
Inherit	LOCATABLE	
Attributes	**Signature**	**Meaning**
	folders: List<X_FOLDER>	sub-folders of this folder, including distinct Folder trees, which may be separately archetyped.
	compositions: List <X_COMPOSITION>	X_COMPOSITIONs in this folder.
Invariants	***folders_validity***: folders /= Void ***implies not*** folders.empty ***compositions_validity***: compositions /= Void ***implies not*** compositions.empty	

5.3.4 X_TERMINOLOGY Class

CLASS	X_TERMINOLOGY	
Purpose	Container class for all terminology data required in an EHR Extract.	
Attributes	**Signature**	**Meaning**
	details: STRUCTURE	Terminological details.
Invariants		

5.3.5 X_DEMOGRAPHICS Class

CLASS	X_DEMOGRAPHICS	
Purpose	Container class for all demograhic data required in an EHR Extract. The list of Parties must be supplied except in the case when an EHR extract is sent within the one environment, and the receiver system has access to the same demographic server as the sender.	
Attributes	**Signature**	**Meaning**
	parties: `Table <PARTY, OBJECT_ID>`	Parties referred to in this extract.
	details: `STRUCTURE`	Other demographic details.
Functions	**Signature**	**Meaning**
	party(key: `OBJECT_ID`): `PARTY`	Obtain the party for the given key.
Invariants	***Parties_validity***: parties /= Void ***implies not*** parties.empty	

5.3.6 X_ACCESS_CONTROL Class

CLASS	X_ACCESS_CONTROL	
Purpose	Container class for all access control data required in an EHR Extract. The list of Access groups must be supplied except in the case when an EHR extract is sent within the one environment, and the receiver system has access to the same access control server as the sender.	
Attributes	**Signature**	**Meaning**
	groups: `Table <ACCESS_GROUP,OBJECT_ID>`	Access groups referred to in this extract.
	details: `STRUCTURE`	Other access control details.
Functions	**Signature**	**Meaning**
	group(key: `OBJECT_ID`): `ACCESS_GROUP`	Obtain the access group for the given key.
Invariants	***Groups_validity***: groups /= Void ***implies not*** groups.empty	

6 Semantics of EHR extracts

6.1 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 17. 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 4. 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.

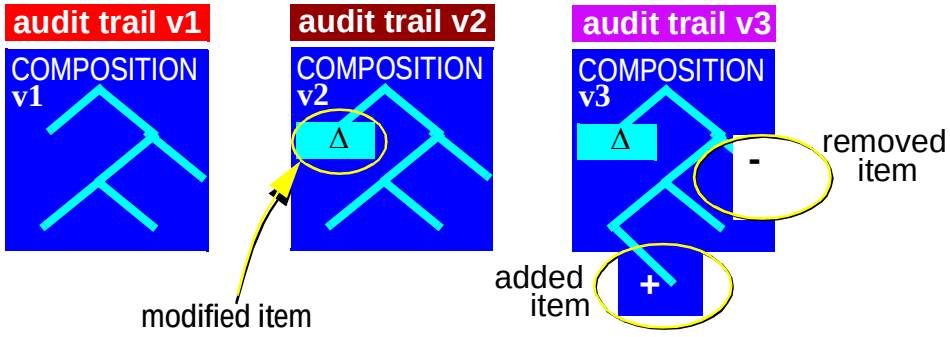

FIGURE 4 Successive Composition versions in a logical

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.

The approach here is a departure from the CEN ENV 13606-4:2000 EHR Extract prestandard (although the future revision underway may change this), 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 CEN pre-standard. The approach was not chosen for a number of reasons:

- 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 CEN specification appears to be in error - the sending of deleted information does not appear to be handled
- the sending of deleted information 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 5 illustrates the structure generated by the accumulation of only three changes shown in the successive versions in FIGURE 4. The large numbers of changes likely in persistent Compositions will generate far more complex structures.

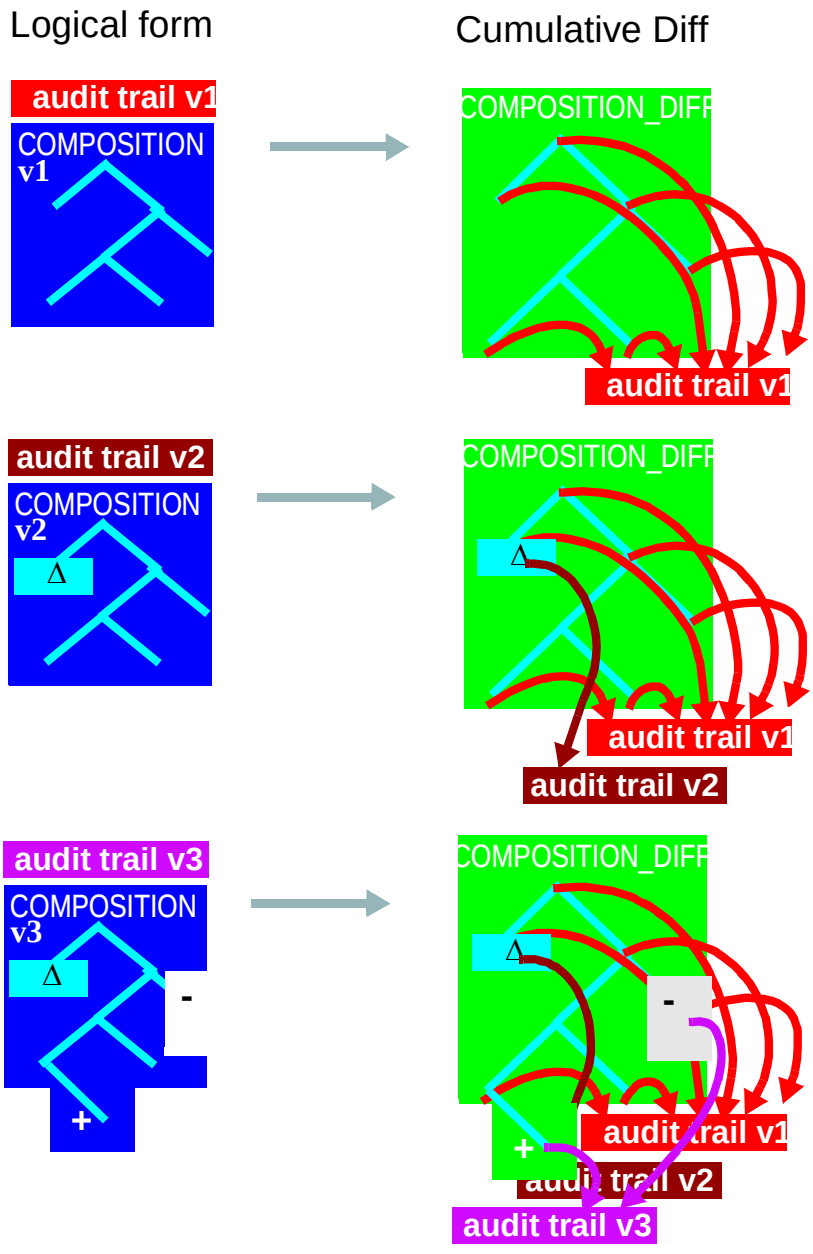

FIGURE 5 Generation of Cumulative Difference Form

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

6.2 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 new `X_DEMOGRAPHICS` instance and write the demographic snapshots for each party mentioned in the `EHR_EXTRACT` itself into the *parties* list;
- For each Composition in the original set, do:
  - create an `X_COMPOSITION`, and set *is_primary*, and write the target Composition *original_path* in;
  - 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 container, i.e. *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"

`To Be Continued: except when no parties included due to local xfer`

  - 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_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: do something about Access_control objects;

7 Communication Scenarios

7.1 Single Hop

To Be Continued:

7.2 Multiple Hop

To Be Continued:

7.3 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.

7.4 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.

Apart from the first one, these conditions are exactly the same as for the normal communication scanrios, 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;

- 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.

8 Receiving and Processing Extracts

The general process for received extracts depends on what they are intended for. Some extracts can be automatically processed, but most extracts containing persistent compositions must be manually reviewed and merged (or not) by a clinician.

For extracts which are intended to be used to automatically update the patient record at the receiver's end, the process is described by the algorithm below. The input of the algorithm is:

- the received extract;

The algorithm is:

- Match the subject of the EHR;
- authenticate etc etc
- For each non-persistent Composition in the primary Composition list of the extract, do:
  - by using paths, if an equivalent composition exists in the local EHR:
    - determine the temporal relationship of the two versions. If the Composition in the extract was derived from the composition in the local record (as it would normally be if the received Composition is a correction to the local version);
  - else:
    - create a new `VERSIONED_COMPOSITION` and write the Composition from the extract in as the first version, with the appropriate `ACQUIRED_TA` audit trail object;

To Be Continued:

8.1 Security and Non-repudiation

8.1.1 Digital Signing

To Be Continued: Digital signatures have to be developed for each implementation technology that extracts are created in.

END OF DOCUMENT