Flexible and Extensible Display of Archetyped Data: The openEHR Presentation Challenge

Lisa M Thurston

Ocean Informatics, Adelaide, Australia

ABSTRACT

EHR systems following the openEHR approach are based on archetypes, which allow data models to be infinitely flexible within the constraints of the openEHR reference model. This unpredictability of the data model for a health record is the central challenge to displaying openEHR EHRs.

This paper describes an approach to displaying archetyped EHR data that which takes into account pre-existing needs for health record display technology as well as the trend towards life-long EHRs and culture-neutral EHR systems.

Display logic models correspond to the eleven categories of openEHR archetypes are created. Using this minimum set of display logic, any openEHR EHR is displayable regardless of the archetypes constraining and defining the data. Per-archetype and/or per-context display logic may be added to the minimum set to gain more control over the rendering of openEHR EHRs.

This approach is implemented in a case study which defines the display models in XSLT excerpts corresponding to archetype categories and specific archetypes. The implementation relies on consuming the EHR data in XML instances of openEHR compositions.

Keywords:

archetypes, electronic health records, data display, openEHR

INTRODUCTION

The openEHR approach to health record management is a novel approach and gaining attention in the health informatics community. A complete implementation of the approach must be able to take into account that data modelling occurs at two levels. The openEHR reference model merely defines constraints for possible health data models, whereas an infinite number of openEHR archetypes* represent actual data models. The challenge faced by health informaticians in implementing the openEHR approach is to find a way to display openEHR health records in such a way that the presentation will be reliable regardless of the archetypes used to define the health data.

ARCHETYPED EHR DATA PRESENTATION REQUIREMENTS

The openEHR approach uses archetypes to maximise data model reuse while allowing infinite data model extensibility. The aim of separating the definition of data models for data types and data structures from those of actual clinical/health concepts is twofold. On one hand the health data will be future-proof because changes to health recording software and practices cannot render the data

* An archetype is a re-usable, formal model of a domain concept.[1] Its application in this paper is to concepts in the health informatics domain, but archetypes are certainly not limited to this domain.

unreadable and unmigratable (nor the data models unusable). On the other hand, the health data will be valid and interpretable regardless of the technical, organisational and cultural context because it does not dependent on closed standards, proprietary data bases, a single terminology, language, or workflow.[2][3]

Ensuring openEHR health data can be consistently manipulated and viewed in any context is assisted by strict governance and sharing of global archetypes, but in some cases, archetypes may be produced for private use. This eventuality must be considered in designing a presentation solution for openEHR health data, since the display technology must work for all potential instances of openEHR EHR data, not only known examples.

The EHR display challenge

The commonly-held expectation with regard to EHRs is that the digitisation of health records will reduce human error, especially in clinical practice, and thus lead to better healthcare outcomes.[4] Moreover, there is preliminary evidence that the potential offered by EHRs for more advanced methods to access and view clinical data could enhance the clinician's information uptake and decision making in clinical settings.[5][6] These hopes are only going to be realised, however, if EHR systems make accessing – that is viewing – the health record faster, easier and less prone to misinterpretation. EHR display technology bears a considerable portion of the responsibility to support these healthcare aims.

Health records have an especially rigid medico-legal requirement to have integrity and function in a clear and predictable manner. While this is largely the responsibility of the non-presentation layers of an EHR system, the presentation layer and display technology must interpret all instances of archetyped data in a consistent way before it is rendered by display components. This is not the greatest challenge facing EHR display technology, however. More importantly, the archetyped data must be able to be displayed in a flexible and extensible way.

Given that the digitisation of health records is relatively recent and that very few complete EHR systems have been implemented around the world, institutions will face many challenges in moving to electronic health records. In particular, users unused to the general digitisation of health records face a steep learning curve to use an EHR system. Other users will be familiar with older health record keeping software which uses older standards and approaches. In addition, many clinical institutions have well-established culture which defines clinical recording practices and workflow.[6] In all cases, a new EHR system will be a new system to learn and the user interface should be carefully to make the transition as simple as possible.

This need for clinical professionals to transition smoothly to EHRs, must be seriously considered in the design of any EHR display technology. To ensure this, and sustained usefulness of the display technology, the presentation of EHR data:

1. must be flexible to support a variety of access needs (different languages, visual layouts, accessibility settings);
2. should be extensible to support the presentation/viewing needs of various clinical and administrative institutions.

Flexible archetyped data display technology will *adapt* with minimal additions and configuration to future display needs. The EHR display technology should also be able to be *extended* where highly-specific views are required. For example, to increase the user acceptance and usability in well-established institutional cultures, customised views and customised look-and-feel should be easy to achieve.

PRESENTATION OF ARCHETYPED EHR DATA

A single archetyped openEHR data instance can be displayed by creating a display logic model applying to that archetype. An archetypable data instance in an openEHR EHR is either a *composition*†, a *section* (which must be contained in a *composition* or *section*), an *entry* (which must be contained in a *composition* or *section*) or an *item structure* (which must be contained within an *entry* or *item structure*). According to the openEHR reference model[7] (release 1.0), there are five sorts of *entries* and four types of *item structures*, meaning there are eleven categories of archetypes in openEHR EHRs. This means that all openEHR heath record data can be displayed, providing we have a minimum set of eleven generic display logic models corresponding to these eleven categories of archetypes.

For example, let us consider we need to display a Histology Report composition. Our sample composition contains:

1. meta data about the report itself, as modelled by the *COMPOSITION.report*[8] archetype;
2. two sections for clinical findings and a clinical summary, as modelled by the *SECTION.findings*[9] and *SECTION.summary*[10] archetypes;
3. a histology observation entry in the findings section, as modelled by the *OBSERVATION.histology*[11] archetype; and
4. a clinical evaluation entry for the findings in the summary section, as modelled by the *EVALUATION.clinical_synopsis*[12] archetype.

In total five archetypes are used to create the data instance in this case, but only four categories of archetypes are used. Thus, four display logic models (the defaults for these four categories) would be sufficient to display this composition in a very generic manner. In other compositions, a completely different selection of archetypes may be used to define the data. Given that five sorts of entry are possible possible and four sorts of item structure are possible, eleven display logic models suffice to provide display logic for any given composition.

The eleven categories of openEHR archetypes

Compositions (*openEHR-EHR-COMPOSITION.x*)
Sections (*openEHR-EHR-SECTION.x*)
Entries:
- Observations (*openEHR-EHR-OBSERVATION.x*)
- Evaluations (*openEHR-EHR-EVALUATION.x*)
- Instructions (*openEHR-EHR-INSTRUCTION.x*)
- Actions (*openEHR-EHR-ACTION.x*)
- Administrative entries (*openEHR-EHR-ADMIN.x*)

Item structures:
- Single items (*openEHR-EHR-ITEM_SINGLE.x*)
- Item lists (*openEHR-EHR-ITEM_LIST.x*)
- Item tables (*openEHR-EHR-ITEM_TABLE.x*)
- Item trees (*openEHR-EHR-ITEM_TREE.x*)

Presentation flexibility and extensibility

The flexibility of this approach lies in the fact that any composition is displayable, regardless of the actual data models used in the EHR and regardless of the nature of the openEHR EHR system containing it. The openEHR reference model[7] enforces the limits on the sorts of archetypes that can be used in an EHR and changes to the reference model are expected to be rare, and minor when they do occur. This safeguards the display technology from becoming obsolete too quickly and suggests it will not require significant changes even when the reference model does change.

Using this approach offers enormous potential to provide custom views per sub-category of

† A *composition* is the smallest possible committable unit of an EHR. From a clinical perspective, the committal of a composition typically corresponds to the data recorded in a single clinical consultation. *Sections, entries* and *item structures* are constituents of a composition. In clinical setting, an entry typically corresponds to the data recorded for a single clinically-meaningful event within a consultation.[7]

archetype or per custom viewing need. Extra display models which target specific archetypes, language-specific or culture-specific viewing needs may be created as needed by both EHR software builders and system deployment administrators. In fact, users may create their own custom display scripts by creating extra display models which target specific archetypes or provide context-specific look and feel to a category of archetypes.

The display technology becomes extensible with an EHR system which knows how to apply the *preferred* display logic model to present compositions, based on institutional, role-based, or personal user viewing configuration settings. For instance, using the earlier example of a Histology Report, an organisation specialising in histological analysis may create custom display logic corresponding to the key archetypes used by Histology Reports. There are two way to do this:

1. A display logic model may be created specifically for the *OBSERVATION.histology* entry archetype (and further specialisations of this archetype). This display logic may then be shared with other institutions interested in more targeted or domain-specific presentation of histology observations.
2. A display logic model may be created in-house to override the existing display logic used to present the data defined by archetypes in the observation entry category. The configuration of the local deployment of the EHR system would need to be altered to direct the system to look at an alternate set of in-house display logic models first instead of the inbuilt defaults.

These two methods may be combined to produce highly-specific display logic models which correspond to both specific archetypes and specific deployment sites, user roles within an institution or individual users. Such context-specific display logic would function by configuring the EHR system to use an alternate set of display logic in preference to the default display logic bundled with the EHR system deployment.

Cascading display for archetyped EHR data

The nature of the display logic approach described in this paper is somewhat analogous to the use of CSS in conventional website design. CSS (cascading style sheets) are used to separate the look and feel of a web page from the page content data. A single rule in CSS contains display logic for a specific element or category of element in HTML. The cascading nature of CSS ensures that while default display logic is specified for a all HTML elements, a style sheet designed by the page author may be used to override the default display logic for specific elements (eg. paragraph elements). More than one style sheet may be applied to a single web page and ranked in order of preference, offering designers the opportunity to create “layers” of specificity for the look and feel of a page.

Display logic models applied to openEHR health data can be designed to cascade in precisely this manner. The EHR system must override only that default display logic for which preferred alternatives exist, as defined by the configuration of display preferences. This enables the authors of alternate display logic to create partial sets of customised display logic, applying only to a certain archetype or a certain openEHR datatype used within a set of archetypes. Thus, the display logic cascades from the more fine-grained levels within openEHR data than at the level of archetypes or archetype categories. For example, the display logic models could target the base text datatype in openEHR (*DV_TEXT*). If we assume this directive was placed in the display logic model for the Observation category of archetypes, then all observation entries would display *DV_TEXT* objects according to the custom directive (unless overridden by separate archetype-specific display logic models). This capability would dramatically simplify the authoring of customised of display logic because authors can define very small snippets of display logic in isolation to the rest of the display logic to a very powerful effect.

Case study: Implementing display models as XSLT

Consider an openEHR EHR system (see *Appendix A*) where the data is retrieved from storage as XML or where the data can be serialised into XML. XSLT‡ is an optimal technology to implement the display logic models for openEHR EHRs, because the language and parser to apply the display logic to the data format already exists. At the time of writing, several XSLT engines exist, supporting all major programming languages/platforms. Moreover, XSL transforms are already used widely as the display logic between XML data and HTML presentation. Hence, XSLT is familiar to developers of web applications and other XML-based applications. A large pool of tools and expertise already exist to develop XSL transforms.

In an XML-based openEHR system, discrete display logic transforms (XSLT excerpts, typically made of one or two XSL templates) are combined by the EHR display application at presentation time to produce a 'grand' XSL transform containing a complete set of display logic for the composition in question. Pre-combined XSLT containing a complete set of default display logic is bundled with an EHR system deployment and simply loaded at presentation time to save processing. This pre-combined XSLT forms the base transform to be used for any composition and extra XSLT excerpts simply override parts of this base transform. Each XSLT excerpt is targeted to a data object defined by a certain category of archetype. For example, an observation XSLT excerpt is targeted at a histology observation in the XML data instance.

Custom-made XSLT excerpts are stored in repositories where the EHR display application looks to retrieve relevant excerpts for the current composition, according to the configured order of preference. Where the most optimal XSLT excerpt can't be located by the EHR display application, the application and configuration settings decide which XSLT excerpt is "next best". The application decides whether it is more important to use an XSLT excerpt for a specific archetype (eg. a lipids studies laboratory observation XSLT excerpt for the *OBSERVATION.laboratory-lipids* archetype) in a repository that is not the preferred repository or a less-specific XSLT excerpt (eg. a generic laboratory observation excerpt designed with only the *OBSERVATION.laboratory* archetype in mind).

The combined XSLT is used by an existing XSLT engine in the EHR display application to produce the final HTML presentation of the EHR data. It can be seen that the sophistication of the final presentation is theoretically limited by the bounds of XSLT and XSLT engines. In practice, however, Javascript and other extensions may be called by the XSLT to extend the presentation behaviour beyond those limits.

Beyond presentation

The approach to archetyped data display described in this paper could be extended beyond viewing to 'interactive browsing' of data and even basic data entry. The shared principle here is that behaviour at the user-interface for compositions is connected to the archetypes used by the data instance. In the case of data display this can be generalised to critically depend only on the archetype category. For simple cases of data entry this will still apply.

The graphical user interface to 'compose' openEHR data (or enter data into a *composition* or other openEHR data structure) could be derived per archetype-category in some cases. More specific composition (composing) views could be created to cater for per-archetype and custom composition views in the same way that per-archetype display logic may be created to enable more control over the actual display of data instances produced by very specific archetypes.

‡Extensible Stylesheet Language Transformations (XSLT) is an XML-based technology commonly used to convert XML data into displayable HTML or alternatively formatted XML.

Watch this space as the archetyping of data models filters into mainstream software design!

CONCLUSION

The challenge of displaying openEHR EHRs is to provide a highly flexible and extensible presentation solution that takes advantage of openEHR's lack of reliance of specific data models. Using a minimum set of display models that match the possible categories of archetypes used in openEHR data instances ensures that compositions defined by any data models can be presented, provided that these data models (expressed in archetypes) conform to the openEHR reference model. It has been shown that this approach may be extended, using custom display models to match specific archetype categories or specific archetypes where customised presentation is required.

Because the display models are cascadable one need only create a partial set of customised display logic to override specific default display logic. Configuration can be used to control the order of preference in cascading display logic. A possibility to enhance the cascading behaviour and simplify the customisation of display logic would involve the presentation engine permitting the overriding of display logic at a more fine-grained level than per archetyped data instance.

This approach to displaying openEHR EHRs is implemented using XSLT excerpts to describe the display logic models. These are applied directly to XML instances of openEHR data by the EHR application to produce the desired rendering of the data in HTML. The EHR application provides a mechanism to alter the order of display logic preference to use alternate/custom XSLT scripts, per archetype or archetype category, which override their corresponding defaults.

While it has been demonstrated that this approach is a flexible and extensible way to display archetyped data, we may see the emergence of similar approaches to provide user interfaces for archetyped data entry and manipulation.

ACKNOWLEDGMENTS

The author thanks Dr. Sam Heard and Heath Frankel for sharing their expertise in clinical practice and health informatics to assist the preparation of this paper.

REFERENCES

[1] Beale T. Archetypes: Constraint-based Domain Models for Futureproof Information Systems. 2001: online accessed 12 March 2006 http://www.deepthought.com.au/it/archetypes/archetypes.pdf.

[2] Heard S and Beale T, eds. Archetype definitions and principles. openEHR Foundation, 2005: online accessed 12 March 2006 http://svn.openehr.org/specification/TRUNK/publishing/architecture/am/archetype_principles.pdf.

[3] Beale T (ed), Goodchild A, Tun Z, Austin T, Kalra D, Lea N , Lloyd D. Archetype Object Model. In: openEHR Specification Release 1.0. openEHR Foundation, Feb 2006: online accessed 12 March 2006 http://svn.openehr.org/specification/TRUNK/publishing/architecture/am/aom.pdf.

[4] Grimson J, Stephens G, Jung B, Grimson W, Berry D, Pardon S. Sharing Health-Care

Records over the Internet. IEEE Internet Computing, May/June 2001.

[5] Dayhoff RE, Kuzmak PM, Kirin G, Frank S. Providing a Complete Online Multimedia Patient Record. Department of Veterans Affairs, VistA Imaging Project, 1999.

[6] Sprague L. Electronic Health Records: How Close? How Far to Go? In: NHPF Issue Brief No. 800 / September 29, 2004. National Health Policy Forum, 2004.

[7] Beale T, Heard S, Kalra D and Lloyd D, eds. Reference Model: The openEHR EHR Information Model. In: openEHR Specification Release 1.0. openEHR Foundation, Feb 2006: online accessed 12 March 2006 http://svn.openehr.org/specification/TRUNK/publishing/architecture/rm/ehr_im.pdf.

openEHR archetypes referenced in order of first appearance in the text:

[8] Composition: Report (http://oceaninformatics.biz/archetypes/openEHR-EHR-COMPOSITION.report.v1.html)

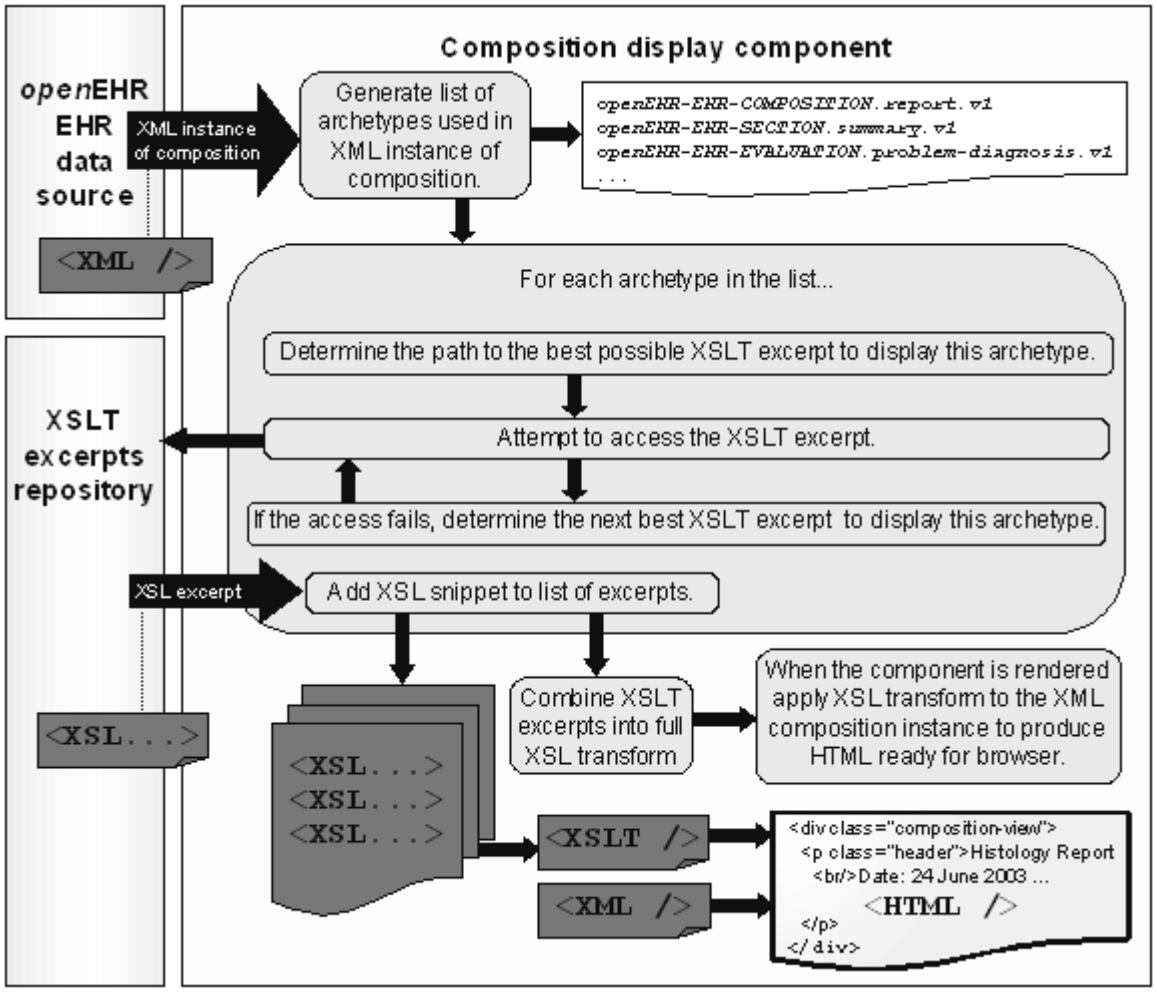
[9] Section: Clinical findings (http://oceaninformatics.biz/archetypes/openEHR-EHR-SECTION.findings.v1.html)

[10]Section: Summary (http://oceaninformatics.biz/archetypes/openEHR-EHR-SECTION.summary.v1.html)

[11]Entry: Histology (observation) (http://oceaninformatics.biz/archetypes/openEHR-EHR-OBSERVATION.histology.v1.html)

[12]Entry: Clinical synopsis (evaluation) (http://oceaninformatics.biz/archetypes/openEHR-EHR-EVALUATION.clinical_synopsis.v1.html)

APPENDIX A – Display logic diagram



Address for correspondence:
Ocean Informatics, Ground floor, 64 Hindmarsh Square, Adelaide SA 5000
Email: Lisa.Thurston@oceaninformatics.biz

Published August 2006