MEDINFO 2004
M. Fieschi et al. (Eds)
Amsterdam: IOS Press
© 2004 IMIA. All rights reserved

# Archetypes: the PropeR way

**Helma van der Linden[a], Jane Grimson[a,b], Huibert Tange[a], Jan Talmon[a], Arie Hasman[a]**

[a] *Department of Medical Informatics, University Maastricht, The Netherlands*
[b] *Centre for Health Informatics, Trinity College, Dublin, Ireland*

**Abstract**

*The PropeR project studies the effect of Decision Support in an Electronic Health Record system (EHR) on the quality of care. One of the applications supports a multidisciplinary primary care team rehabilitating stroke patients in their home environment. This project required an EHR system that could handle information of multiple disciplines and multiple, distributed data sources. It should also be flexible enough to handle an entirely different domain with only minor modifications. The resulting EHR system is a distributed system based on international standards and reusable components. It is generic in nature since all references to the domain are stored in separate XML documents: the clinical data are defined through a modified version of the OpenEHR archetypes, described in XML and views on the data, either for review or for data entry are also described in XML documents. Currently, the system is being evaluated by a care team using laptops with a wireless internet connection.*
*The use of standards greatly improves the quality and reusability of the resulting software system, but they do not solve the issues that surface during implementation, such as context and screen representation.*

***Keywords***

Medical record systems, archetypes, XML, OMG HDTF, COAS, PIDS, stroke.

## Introduction

Decades of software development have produced an enormous variety of EHR systems[1], from small research systems focusing on one topic to large systems used in many hospitals.

Almost all systems currently in use in regular practice have suffered from the fact that software development could not keep up with the rate of change in the medical domain. It was also sometimes difficult for IT professionals to understand the medical concepts they had to translate into software.

The Synapses [1, 2] and the Good European Health Record (GEHR) [3] projects among others tried to overcome these problems by using predefined generic, abstract building blocks. Instances of these blocks could easily be stored in a generic database. The blocks also serve as the basis for construction of medical concepts. In the Good Electronic Health Record project (the second GEHR project) [4] these constructions became known as archetypes. The latter GEHR project is now succeeded by the OpenEHR project [5].

1. In this paper we will refer to any system holding health related information on patients as an EHR, although we are well aware that others give different meanings to the various synonyms.

### Archetypes

Archetypes (1) are electronic descriptions of medical concepts, such as blood pressure, lab test and diagnosis, created from building blocks by medical experts. The building blocks are defined in the Reference Model (2), which is further constrained by the Archetype Language/Model (3). A database/Information Model (4) that is able to handle the elements of the Reference Model is therefore able to handle archetypes[6]. Archetypes are typically stored separated from the Information Model in an archetype repository.

These four models (1 – 4) delegate the work to the respective domains: the IT specialists can build both the software that can handle archetypes and a database defined to handle the building blocks of the Reference Model and the Archetype Language/ Model, while medical specialists define the necessary medical concepts using the constrained building blocks available through the Archetype Language/Model.

This not only separates responsibilities, but also allows for work to proceed in parallel since software development does not have to wait until all data definition problems are solved.

### PropeR

The PropeR project started in 2000 to study the effect of decision support software (DSS) in an EHR on the quality of care in two settings. This article focuses on the setting where stroke patients are rehabilitated in their home environment [7]. This required a multidisciplinary EHR system that could be extended with a DSS component. The EHR system had to be distributed, because it has to integrate information from various systems. In addition, the practitioners needed to access the system from various locations. At the time, there was no system available that could meet these requirements so the decision was made to develop a new system. Extra requirements were defined to achieve an EHR system that would not only meet the usual requirements for any EHR system (privacy, security, reliability etc.) as well as the project specific requirements, but would result in an EHR system that would be stable and flexible enough to be useful beyond the scope of the current project. It should be able to handle both changing user requirements as well as different (medical) do-

mains. These requirements could be enforced using standards and available reusable components.

In summary, we set out to create a generic software system that could handle data retrieval and data entry without hardcoded knowledge of the data definitions it processes, with minimal impact on the software when adding and modifying data definitions, or even when replacing the entire domain model.

The architecture of the PropeR system will only be briefly discussed in this paper, for details see [8, 9]. In this paper the focus will be on the implementation of the archetypes concept in a system used by different disciplines and the issues that evolved in the development process.

### Proper Architecture

The PropeR EHR system (see Figure 1) is a loosely coupled client-server application based on the OMG HDTF specifications [10]. These specifications are platform and programming language independent specifications and solve generic problems. The two specifications currently implemented in the PropeR EHR system are the Person Identifications Specification (PIDS) and the Clinical Observation Access Specification (COAS). The PIDS defines a generic interface to a master patient index and handles the omnipresent problem of correctly identifying a person. The COAS defines a generic interface to medical data in a distributed environment.

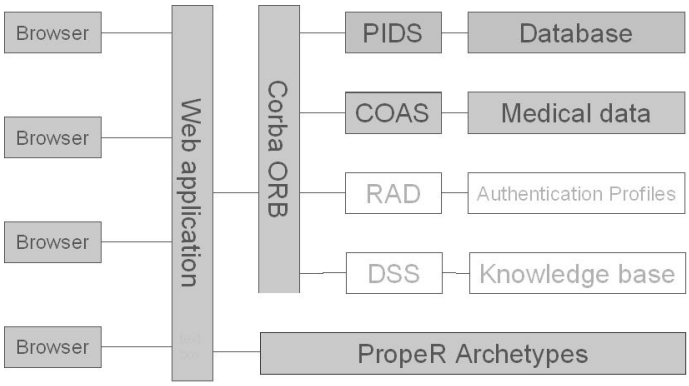


*Figure 1 - Proper EHR diagram.*

These specifications are implemented as open source components by the OpenEMed team in Los Alamos National Laboratory [11]. The PropeR EHR system uses the OpenEMed PIDS and COAS servers with their respective databases as database systems.

A third specification, the Resource Access Definition Framework (RAD) is a generic interface to an authorization server and handles role-based access profiles. The RAD server will be implemented in the near future. The DSS will also be implemented in the near future. In Figure 1 they are marked in a light color.

The COAS specification describes data structure, not data content. It is primarily defined for data retrieval, but the OpenEMed team has added storage functionality to their implementation. The archetype concept can be considered as a complement to COAS, since it describes data content based on predefined data structures. Thus the COAS specification provides a generic structure for data storage and retrieval, like the database/Information Model, mentioned before.

### PropeR Archetypes

Initial study of the OpenEHR archetype documentation showed that there was no archetype builder available. Time prevented the development of such a builder during the PropeR project. We also wanted to use the COAS ObsDataStruct specification as the Information Model, for several reasons:

- it is a generic structure, recursive in nature, which can hold any data structure;
- it is part of an international standard;
- a useable reference implementation is available.

This led to the definition of the PropeR archetypes.

The PropeR project defines an archetype as a medical concept that can be represented in the user interface as one element. In a traditional system it would be a field with a label. A view in this context is a set of related elements (i.e. archetypes) that are presented together on the user interface.

PropeR archetypes had to meet the following requirements:

- Easy development: since there are no dedicated archetype builder tools available, the archetypes have to be built by hand. This process should be as simple as possible.
- Based on COAS specifications: Archetypes are expressed in structures that can be converted to COAS specifications and/or use COAS data types.

The Proper archetypes are to be used in the following environment:

- Generic software: the application handling the data should have as minimal as possible hardcoded knowledge of the archetypes and of the data content. It should be easy to replace one domain with a completely different one, with at most minor modifications to the software.
- Generic display: there should be no hard coded information to link archetypes to their screen representation. This would reduce the generic nature of the software.
- Audit trail: the system should provide an audit trail. All versions of a specific instance of an archetypes (i.e. archetypes with the data filled in), are stored.

## Issues in modeling archetypes

Two issues arose in the process of defining the correct archetypes: the context and the screen representation.

In the following part we will explain these issues and the solution we developed.

### Context

The PropeR EHR system will, at first, work as a stand alone system, but ultimately be used as a distributed system connecting data from different sources and being able to communicate with other systems.

Proper communication requires knowledge of context of the information transferred.Studying the available literature showed there are several ways of EHR data modeling for the purpose of

communication. One approach defines every concept (i.e. "archetype") as an end node of a tree, with unique paths leading to the required concept. The tree is modeled up front, based on common principles (e.g. BloodPressure is stored as part of "Vital Signs", which is stored as part of "Cardiac System") [2], or based on the typical paths of data entry (e.g. BloodPressure is stored as part of "Physical Examination", which is stored as part of "Patient Encounter") [12].

Another approach is modeling each concept as a unit in itself. We call this a loosely coupled approach. In either case the information is extended with appropriate context information such as timestamps and other relevant information.

The tree approach provides a unique path for each concept, which provides additional context information. It is straightforward to communicate the entire tree or part of the tree to another system. However, this approach requires knowledge of the tree in the software, in order to build or parse the tree. Also, both systems need to use the same tree definition.

The loosely coupled approach requires no knowledge of the way the concept is stored, but lacks the context information the tree approach provides. Systems communicating these items need only identical definitions of the structure.

We rejected mapping of stored data to a tree definition, since it would require a lot of work and would close the system to new concepts, i.e. when a new item is added, the tree and the mapping have to be updated and the software and all the systems it communicates with have to be updated to use the newest definitions.

The loosely coupled approach does not require any mapping, since communication is done on the level of items, i.e. archetype instances. If necessary, the archetype definition is also communicated. This requires items to be defined on the lowest possible level, the level of medical concepts.

In the PropeR project exchange of information between systems will be done using the COAS specification, which is also used for data storage and retrieval. Audit trail is implemented by adding new versions of updated items to the database[1].

To preserve the context, the Form concept was introduced in the PropeR EHR system. A Form could be equivalent to a paper-based form or to a view that has no paper-based equivalent. It defines a set of archetypes. Instances of a Form contain references to a specific instance of these archetypes. In functionality it is roughly equivalent to the OpenEHR Organiser [13].

### Screen representation

Screen representations of data must have knowledge of the data to properly display the information. This means that screen representation definitions also depend on the capabilities of the devices used to display them and on the way the information is entered (e.g. handheld vs. regular PC vs. touch screen). An even more important aspect is the fact that screen representations should result in views that match ergonomic principles, enhance interpretation and improve the quality of the data entered.

As stated before we define generic software as software that has no hard coded knowledge of the information it handles, the business model, or the screen representations of this information. This leads to the conclusion that screen representations should be defined and stored separately from the software that uses them.

There are two options for storing screen representations:

- storing them as part of the archetype
- storing them separate of the archetypes

The OpenEHR statements show that archetypes can be defined by domain specialists, while software-related matters are handled by software specialists. Storing screen representations in archetypes violates this separation, since it is our opinion that the definition of screen representations should be a joined effort of software developers and users (who are not necessarily the same persons as the domain specialists defining the archetypes). Ideally, user interface design is done by specialist human interface experts.

We decided to implement the second option, since making major changes in archetype structures will occur less frequently than adding new views.

## Implementation

### *PropeR archetypes*

We decided to implement the PropeR archetypes as XML descriptions [14], combined in one XML document. We called these descriptions Elements. An Element has a code, which is the "name" used in querying the COAS server and qualifiers that add context information, such as the ID of the patient, start and end time of the observation etc., and of course, the value of the element. An example of an element is shown below.

```
<element code="RecentTravels">
        <qualifier name="DNS:proper.mi.unimaas.nl/QualifierCode/
ProperObjectID">
<!--  Id generated by the software for the sole purpose of identifying
instances of an archetype -->
                <textType format="QualifiedCodeStr"/>
        </qualifier>
<… more qualifiers …>
        <element code="DNS:proper.mi.unimaas.nl/TravelStart-
Time">
                <dateType format="HL7"/>
        </element>
        <element code="DNS:proper.mi.unimaas.nl/TravelEndTime">
                <dateType format="HL7"/>
        </element>
        <element code="DNS:proper.mi.unimaas.nl/TravelDestina-
tion">
                <textType format="CodedConcept">
                        <value code="DNS:proper.mi.unimaas.nl/
                        None"/>
                        <value code="DNS:proper.mi.unimaas.nl/
                        TheNetherlands" />
                        <value code="DNS:proper.mi.unimaas.nl/
                        Europe"/>
                        <value code="DNS:proper.mi.unimaas.nl/
                        RestOfTheWorld"/>
                        </textType>
        </element>
</element>
```

1. Data storage is only allowed in the local database. All other data sources are considered read-only.

The software parses this document into a list of DataElements. A DataElement is an in-memory representation of an archetype and is used as a template for the instances, i.e. data is retrieved from the COAS server and a copy of the DataElement is created with actual data filled in.

***PropeR screen Forms***

A PropeR Form is an in-memory representation of a set of DataElements in a particular order. A Form started out as an electronic representation of a paper-based form, but in essence it is merely a view on a set of DataElements. A Form is also defined in an XML document, one document for each Form. It describes which DataElements should be shown on screen and how. Since the PropeR software uses HTML pages as user interface, the descriptions are geared to HTML to simplify the conversion.

There are descriptions for display forms (i.e. for review) and for input forms (i.e. for data entry). Each element in a form description contains a "label" attribute which holds the display name of the label. Elements of type "Date" also have a format attribute, which holds the format string for displaying the date. These attributes make it easy to internationalize the user interface. An abbreviated example is shown below:

```
<form code="TravelForm" label="Recent travels of patient">
        <qualifier name="DNS:omg.org/DSObservationAccess/
HL72.3/OBR/ObservationDate_Time">
                <display type="display" label="Start of Observa-
tion">
                        <type name="date" format="yyyy-MM-dd
HH:mm:ss">
                </display>
                <display type="input" label="Start TimeStamp of
Observation">
                        <type name="date" format="yyyy-MM-dd
HH:mm:ss" verify="true">
                </display>
        </qualifier>
        <…more qualifiers …>
        <element code="DNS:proper.mi.unimaas.nl/TravelStart-
Time">
                <display type="display" label="Start of Travel">
                        <type name="date" format="yyyy-MM-dd
HH:mm:ss">
                </display>
                <display type="input" label="StartTime of Travel">
                        <type name="date" format="yyyy-MM-dd
HH:mm:ss" verify="true"/>
                </display>
        </element>
        <… more elements …>
</form>
```

There is however a slight performance penalty due to the in-memory manipulation of the XML document and the subsequent transformation using XSL. Since all processing is done on the server, upgrading the hardware will increase the performance. Experiments showed that there is no noticable difference in performance when the system is accessed through a dial-up connection with an average 56.6K modem versus access through a Local Area Network.

Although the XML and XSL documents were coded by hand, using a specialized XML/XSL editor, it was a relatively easy task. Adding a new form would take roughly 4 hours from scratch to final version for any person familiar with XML.

We could not avoid to use some hard coded references to the data, although it is very limited: the software handles classes of Persons (i.e. information stored in PIDS servers), Forms and DataElements (information stored in COAS servers).

HTML pages, especially when generated from a template, are not the most prominent examples of ergonomic design. This is also true for the PropeR EHR system. Improving the screen design will require either a complete rewrite of the XML/XSL description of the forms or a switch from HTML pages to a dedicated client application.

The system is able to extract information from different sources as well as store/retrieve information from a local database. A different project in our department will be using the PropeR EHR system in a clinical department with a local database as well as a connection to the hospital information system.

## Evaluation

An experiment to test the EHR system is running. Four primary care practitioners (a speech therapist, a physiotherapist and two care coordinators) are equipped with a laptop and a GPRS card. They make a wireless internet connection to the PropeR EHR system at the patient's home. The main issue to evaluate in this experiment is the software itself (userfriendliness, possibilities of bugs, performance).

We hope to finish this experiment quickly and start the larger experiment where all 17 therapists are equipped with similar hardware. This experiment focuses on the evaluation of the system in terms of user satisfaction, quality and quantity of patient data.

### Acknowledgments

The authors wish to thank the OpenEMed team for their continuous help during the development of the PropeR software, all the primary care practitioners for their patience and enthusiasm to participate in the project and Teri Pittman for her comments on this paper. Finally, we thank Vodafone Netherlands Foundation for providing the GPRS cards and the operational costs.

## References

[1] anonymous. The Synapses project. http://www.chime.ucl.ac.uk/work-areas/ehrs/SYNAPSES/; Last accessed: 2003/09/08.

[2] Grimson W, Berry D, Grimson J, Stephens G, Felton E, Given P, et al. Federated healthcare record server - the Synapses paradigm. *Int J Med Inf* 1998;52:3-27.

[3] anonymous. The Good European Health Record. http://www.chime.ucl.ac.uk/work-areas/ehrs/GEHR/; Last accessed: 2003/09/08.

[4] Beale T, al. e. Good Electronic Health Record Australia. http://www.gehr.org; Last accessed: 2003/09/08.

[5] anonymous. OpenEHR. http://www.openehr.org OpenEHR foundation; Last accessed: 2003/09/08.

[6] Beale T, Goodchild A, Heard S. Design Principles for the EHR. http://www.openehr.org/downloads/design_principles_2_4.pdf; Last accessed: 2003/09/09.

[7] anonymous. Transmuraal Zorgmodel CVA Regio Heuvelland (in Dutch). Heerlen: Synchron; 1996 april 1996.

[8] van der Linden H, Boers G, Tange H, Talmon J, Hasman A. PropeR: a multidisciplinary EPR system. *Int J Med Inform* 2003;70:149-160.

[9] van der Linden H, Tange H, Talmon JL, Hasman A. PropeR revisited. In: Baud R, Fieschi M, Le Beux P, Ruch P, editors. *Proc Medical Informatics Europe*; 2003; St. Malo, France: IOS Press; 2003. p. 346-351.

[10] anonymous. OMG Healthcare DTF. http://www.omg.org/healthcare; Last accessed: 2003/09/08.

[11] anonymous. OpenEMed. http://www.openemed.org http://www.openemed.org; Last accessed: December 2002.

[12] van Ginneken AM, Verkoijen MJ. A Multi-Disciplinary Approach to a User Interface for Structured Data Entry. In: IMIA, editor. *MEDINFO*; 2001; London: IOS Press; 2001. p. 693-697.

[13] Beale T, Heard S, Kalra D, Lloyd D. The openEHR EHR Reference Model. http://www.openehr.org/downloads/ehr_rm/ehr_rm_4_3_2.pdf OpenEHR; Last accessed: 2003/09/09.

[14] anonymous. Extensible Markup Language (XML). http://www.w3.org/XML/ World Wide Web Consortium; Last accessed: 2003/09/08.

**Address for correspondence**

Helma van der Linden
University Maastricht
Medical Informatics
POBox 616
6200 MD Maastricht
The Netherlands
h.vanderlinden@mi.unimaas.nl