MEDINFO 2007
K. Kuhn et al. (Eds)
IOS Press, 2007

Generic Screen Representations for Future Proof Systems – Is It Possible? Two-model approach to a generic GUI

Helma van der Linden[a], Thilo Schuler[b], Rong Chen[c], Jan Talmon[a]

[a] *Medical Informatics, University Maastricht, The Netherlands*
[b] *Department of Medical Informatics, University of Freiburg, Germany*
[c] *Department of Biomedical Engineering, Linköping University, Sweden*

Abstract

Semantic interoperability should not only cover system interpretation of incoming information, but should be extended to include screen representation. This article describes a two-model approach to generate a screen representation for archetype-based information, which is inspired by the two-model approach used by openEHR for their archetypes. It provides a separation between software-related display knowledge and domain-related display knowledge and is designed with reuse of components in mind. This approach leads to a flexible GUI that can adapt not only to information structures that are not predefined within the receiving system and display them in a meaningful way, but also to novel ways of displaying the information.

We are working on a proof of concept implementation to validate the approach.

Keywords:
computerized medical record, electronic health record, user-computer interface, openEHR, archetypes, HL7

Introduction

Patient mobility is increasing over the last decade varying from "shopping around for care" in different health care organizations to prolonged stays abroad with the increasing need of care. This results in fragmented patient-related health information distributed across different systems.

A current approach for future health information systems is to create a virtual health record that integrates the fragmented information by connecting distributed systems and presenting them as one. The underlying principle is information exchange based on standardized messages. The two major approaches in this respect are HL7 v3[1] [1] and CEN/TC251 13606 [2, 3] combined with openEHR archetypes[2] [4].

1 In this article HL7 will refer to the new v3 standard in its latest form.

2 In this article we will refer to the CEN 13606/openEHR archetypes as openEHR or archetypes for brevity.

With ongoing efforts towards harmonization of the best of both frameworks it will be possible to see virtual health records come into reality within some decades.

In this article we assume the existence of such an environment where virtual health records exist and information exchange is not limited to the systems of a single organization, but expanded to incorporate other organizations on a regional or maybe even global scale. In this environment it is possible to retrieve or receive patient information of which the structure has not been known before. Indeed, as advocated by openEHR, true future-proof electronic health record systems will be able to accommodate new medical concepts without the need for redevelopment. The key idea behind archetypes is to express new information structures as a combination of predefined classes.

The openEHR Foundation has currently the only architecture that allows handling of unknown information structures. We therefore focus on openEHR archetypes in this article. This does not imply that archetypes are the only means of exchanging information.

Scenario

A GP suspects that the patient suffers from a hereditary disease and refers the patient to the genetics clinic where tests will be done to confirm or reject his suspicion. Unfortunately, his suspicion is confirmed and the GP receives a discharge letter that contains a summary, the lab results and a family tree.

When we assume that the discharge letter is a structured message containing the data structures with the relevant data rather than a formatted display document, the question arises how the information of the discharge letter should be displayed on the GP's screen and in particular the information that is normally not part of the GP's system (i.e. the family tree).

The article discusses an approach to display new information structures on a user's screen using as much display knowledge as available.

Background

ISO 18308 defines semantic interoperability as the ability for information shared by systems to be understood at the level of formally defined domain concepts so that the information is computer processable by the receiving system [5].

Both HL7 and openEHR support semantic interoperability at two levels: at the data structure level and at the domain level.

At the data structure level, medical concepts are described using predefined data structures. This ensures that the information exchanged is complete (i.e. it contains all relevant data and metadata) and can be parsed, stored and subsequently retrieved. At the domain level metadata such as the code and coding scheme are used to avoid ambiguity in understanding.

The PropeR project has revealed that semantic interoperability is not simply a matter of interoperability between systems, but also between user and system. To refer to the scenario, even if the GP's system is capable of storing and subsequently retrieving the fully structured family tree, if there is no suitable screen representation, it is very difficult for the GP to correctly interpret the information.

Van der Meijden [6] and van Ginneken [7] have already discussed the difference between data entry and data retrieval with respect to the screen representation. Since it is logical to assume that data entry is only done in the local system, the issue of undefined data structures does not occur during data entry as we may assume that the local system is designed to support the specific user tasks in the application domain. The approach we present here will primarily be focused on data consultation and not on data entry.

Methods

In the context of the PropeR project [8, 9] we built a web based EHR system based on a simplified version of archetypes. We focused on the implementation of a domain-agnostic system and the strict separation between archetypes and screen representations.

We followed a similar approach by researching the feasibility of generating a GUI based on openEHR archetypes [10].

The lessons learned in both projects were combined to develop a more generic approach that can handle the situation we discussed before.

Results

Presentation level interoperability

In our view displaying information in an unambiguous way that supports the user's work processes, requires three types of knowledge:

- Knowledge of the information to display;
- Knowledge of the way a user is accustomed to view information;
- Knowledge of the device that is used to display the information.

Information-related presentation knowledge

At the lowest level this refers to the display of the data types that are used to construct the information structure: numbers are displayed differently than text. This is however, not sufficient. Even the example of a simple blood pressure shows that a higher level of knowledge is necessary to correctly display a blood pressure; that is in the common form of two numbers separated by a slash. A graphic tree form would best represent a family tree.

Localized presentation knowledge

Displaying information can be subject to local customs, varying from the local language and the local date format to preferred units (e.g. mg/dl vs. µmol/l) and coding schemes. There are also personal differences in what the best way of information presentation is with different reasons such as learned behavior or different cognition strengths (e.g. visual, textual).

Device-related presentation knowledge

The current trend towards ubiquitous computing has produced a large range of devices capable of sending and retrieving information ranging from desktop computer and laptops to tablet pcs, pda's and smartphones. While each modern model contains a webbrowser, and thus an abstraction from the underlying device, the supported functionality and the screen size places extra constrains on the presentation.

These different types of knowledge are often hard coded into the GUI of the client application. This makes it very hard to display incoming information from a different domain.

A two-model approach to generic GUI generation

Given the premise that future-proof systems are also capable of displaying information from other domains, it is necessary that these systems contain domain-agnostic screen representational functionality.

From the PropeRWeb application we learned that screen representation knowledge, however low-level, should not be incorporated in the archetype definition [11]. Not only does it introduce two different kinds of knowledge (medical domain knowledge and presentation knowledge) in a single model, but it is also common knowledge that a single data type, especially numerical, can be displayed in different ways, for example as a single number or as a table or graph.

In our approach we distinguish two models: a display oriented model (the GUI model) that defines widgets as screen presentation units and a domain oriented model (the content model) that defines content units which create meaningful presentations using widgets. The first model is the realm of the GUI designer, while domain experts use the second model.

Localized presentation knowledge is defined in profiles and views. This results in four sets of presentation units, which are shown in figure 1: widgets, content units, views and profiles.

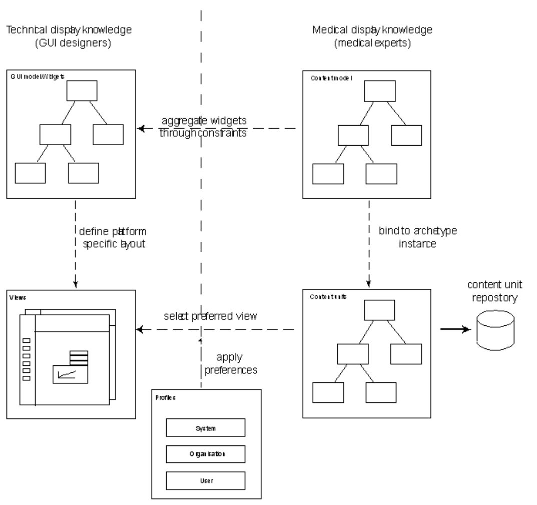

Figure 1 - Two-model approach to a generic GUI

All presentation units follow object-oriented design in which a specific unit inherits characteristics from a more generic unit. This improves consistency and flexibility. It is also valid to define multiple units for the corresponding information unit.

GUI model

The building blocks of a GUI are widgets. A widget is a display unit that contains presentation knowledge for a single data type. These widgets can be mapped to classes in the Reference Model of openEHR. Two types of widgets exist: data-oriented widgets such as “text”, “image” and “number” and layout-oriented widgets such as “list” and “table”.

The GUI model defines generic widgets that are converted to specific versions in the underlying system by using views.

Content model

Content units are defined using a content unit definition language. They are a semantic aggregation of widgets. They can be regarded as the display counterpart of archetypes. Content units, like archetypes can include other content units. Content units specify the binding to the information in the archetype instance as well as an established layout, such as the X/Y format of a blood pressure. Note that this layout only specifies relative positions of the included widgets and/or content units.

At the top level a content unit matches a COMPOSITION. These high-level content units are called *documents*. Different documents can be designed to reflect the differences in users’ roles.

Content units can also include calculations, e.g. the total score of a test. They can also include normal ranges for semantic interpretation of the value. For example: the value of a body mass index can be color-coded based on the semantic interpretation (e.g. “normal” is green, “obese” is red).

Like archetypes content units are stored in a content unit repository. Since they are a semantic, platform independent representation of an archetype, they can be shared in the same way archetypes are sharable.

Views

Views can be regarded as implementations of content units customized for the device or application that will be displaying the information. Views can also include other GUI artifacts such as navigation bars.

A view is focused on presentation of the content and therefore part of the GUI designers’ realm.

Profiles

The focus of profiles is the conversion of the information to match the user’s expectations and thus avoid interpretation errors.

A profile contains preferences at various levels that modify the presentation of the information. There are three levels:

- *System level*. This level contains generic preferences that should always be applied, e.g. language, date format, metric vs. imperial system etc.
- *Local level*. This level contains generic preferences that are organization or location specific and are more domain-related. These preferences include preferred units and preferred terminologies.
- This level can also include role-based preferences that refer to role-based documents.
- *User level*. This level contains specific user related preferences that can modify the preferred view for a certain type of COMPOSITION e.g. if the user prefers graphs to tables.

Presentation generation

The process of generating the presentation is based on the pipeline concept. A pipeline can be compared to an assembly line where material arrives in a certain form, which is then processed by various stages along the line and finally delivered as a complete product. Adding or removing stages delivers a different product without affecting the other stages. A successful implementation of the pipeline concept can be found in Apache Cocoon [12].

A pipeline offers a component-based approach to the transformation of information. By adding or removing transformation components, the end result can change without affecting the other components. Different pipelines can implement different functionalities while sharing components.

In our approach an openEHR composition enters the pipeline. This composition can be the result of a query for information or the result of a notification of new information. In both cases the composition can contain instances of unknown archetypes.

Transformations handle device selection, presentation units selection, profile application and the final rendering of the selected view.

Figure 2 shows the generation process.

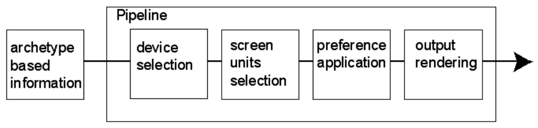

Figure 2 - Generation of view

There are several advantages to this solution:

- First and foremost this approach offers the flexibility of defining specific, optimized screen representations for known information structures, while providing the means to generate useable screen representations of unknown information structures.
- By separating the various types of presentation knowledge into distinct models, it is possible to separate pure GUI knowledge from medical display knowledge, thus honoring the two-model approach and promoting reuse.
- This approach is flexible enough to build a role-based GUI. Nurses and physicians can see the same information but optimally presented for their specific needs, while the only difference in development might be a document definition.
- The evolution of medical knowledge will always create new data types and new archetypes, which would lead to new display representations. Our approach ensures that as much of the available display knowledge can be reused.
- New and novel ways to view EHR information are a topic of ongoing research. By adhering to the proposed approach these views can benefit from the available display knowledge that is already expressed in content units. [13]
- Both HL7 and openEHR archetypes are using a limited set of predefined data types with an ongoing effort to harmonize the sets between the two parties. By describing one or more widgets for each data type it should be possible to provide a meaningful display of the information, without incorporating presentation knowledge in the information structure. This means the current archetype or message specifications need not be extended and the number of widgets is not very large.
- The information can be converted to match local and user preferences such as preferred coding scheme, language, units and more.
- A fallback mechanism is used to select a more generic representation in the absence of a specific one.
- Standardized content units could be shared between systems; the same way archetypes can be shared. This increments the intelligence and usability of the system.
- The pipeline approach not only allows reuse of components, but also offers flexibility in adding functionality by a simple addition of pipelines.

This approach complements the openEHR architecture where templates are used to create a higher-level composition by constraining and ordering archetypes. In contrast, the openEHR templates are used to *create* an information structure, while the approach proposed in this paper is used to *display* the information.

HL7 focuses on message exchange only and therefore considers this problem to be part of the receiving system's domain. However, given the similarities in structure between archetypes and messages we believe this approach is equally useful in that realm.

There are also disadvantages:

- Higher-level, specific views can only exist for predefined information. New information or information from different domains will fall back to a more basic representation.
- A repository, equal to that for archetypes, is necessary for the various presentation units.
- A mechanism for retrieving an appropriate screen representation for the current archetype is necessary, since the most appropriate selection is based on multiple parameters, described earlier, that cannot be stored in the archetype instance.

The advantages of having flexible GUI interfaces outweigh the disadvantages. By incrementally defining screen representations that can be built on top of each other, there is less duplication of work in building a GUI. A higher-level screen representation allows the user to better interpret the presented information thus leading to more efficient and more reliable information exchange. Screen representations for new information structures can be added to the system without major redevelopment of the application.

Currently we are working on a proof of concept using the Apache Cocoon web application framework [12] to build a web application that can display instances of various openEHR archetypes based on the approach described here.

The Apache Cocoon web application framework is a generic open source framework that is heavily based on the concept of separation of concerns to define strict distinctions between model and view. It implements the pipeline concept and also provides a set of generic widgets and an XML-based language to define what we call views. Since Cocoon excels in processing XML it is a good candidate to build a generic generated web based GUI using the approach that we have presented before. A first version will be presented in the openEHR workshop of Medinfo 2007.

Related work

A similar approach is developed by Ocean Informatics and implemented in their EhrView [14]. The EhrView application modifies the information through a series of XSLT stylesheets. These stylesheets are selected by matching the archetypes names in the composition. The matching process selects the most specific stylesheet available in a repository.

The EhrView application does not separate content related modeling from software related modeling and it is only defined for one type of device: a regular screen of a desktop or laptop pc. It also offers limited options to adjust to local or user preferences.

Fiala et al. [15] have described a component-based approach for adaptive web documents that influenced our approach. They too make a distinction between content-related and display related presentation knowledge and they also used the pipeline concept to define web document generation. However, their focus is on adapting information presentation to user preferences and devices. The information is known and defined in advance and there is no method to handle unknown information structures or describe conversions to preferred units.

Conclusion

Semantic interoperability does not stop when information from one system can be successfully understood and/or incorporated in another system. It is also necessary to provide a screen representation that gives the user of the receiving system a clear understanding of the new information.

In this article we described an approach that extends the two-model approach that is currently used by openEHR by a similar approach for the GUI.

We argued that this approach leads to a flexible GUI that can adapt to information structures that are not predefined and still display them in a meaningful, higher-level way. We are currently working on a proof of concept implementation.

Acknowledgments

We would like to thank the team from Ocean Informatics who kindly provided the information we needed. We would also like to thank Patrick Ahles for his valuable insights.

References

[1] Health Level 7 Inc. [last visited 21 Mar 2007]; Available from: http://www.hl7.org

[2] Health Informatics – Electronic health record communication - Part 1: Reference Model: CEN/TC251; 2005 March 2005. Report No.: prEN13606-1:2005:E.

[3] Health Informatics – Electronic health record communication - Part 2: Archetypes: CEN/TC251; 2005 December 2005. Report No.: prEN13606-2:2005:E.

[4] openEHR. [last visited 21 Mar 2007]; Available from: http://www.openehr.org

[5] ANSI. ISO/TS 18308 Health Informatics - Requirements for an Electronic Health Record Architecture: ISO; 2003 2003-04-23.

[6] van der Meijden MJ, Tange HJ, Boiten J, Troost J, Hasman A. An experimental electronic patient record for stroke patients. Part 2: system description. Int J Med Inform. 2000 Sep;58-59:127-40.

[7] van Ginneken AM. Considerations for the representation of meta-data for the support of structured data entry. Methods Inf Med. 2003;42(3):226-35.

[8] van der Linden H, Boers G, Tange H, Talmon J, Hasman A. PropeR: a multi disciplinary EPR system. Int J Med Inform. 2003 Jul;70(2-3):149-60.

[9] van der Linden H, Talmon J, Tange H, Grimson J, Hasman A. PropeR revisited. Int J Med Inform. 2005 Mar;74(2-4):235-44.

[10] Schuler T, Garde S, Heard S, Beale T. Towards Automatically Generating Graphical User Interfaces from openEHR Archetypes. Stud Health Technol Inform. 2006;124:221-6.

[11] van der Linden H, Grimson J, Tange H, Talmon J, Hasman A. Archetypes: the PropeR way. Medinfo. 2004;11(Pt 2):1110-4.

[12] Apache Cocoon Web Application Framework. [last visited 21 Mar 2007]; Available from: http://cocoon.apache.org

[13] Sundvall E, Nyström M, Forss M, Chen R, Petersson H, Åhlfeldt H. Graphical Overview and Navigation of Electronic Health Records in a prototyping environment using Google Earth and openEHR Archetypes. MedInfo 2007; 2007; Brisbane: accepted for publication; 2007.

[14] Ocean Informatics Pty L. EhrView. [last visited 21 Mar 2007]; Available from: http://www.oceaninformatics.biz/products/EhrView.html

[15] Fiala Z, Hinz M, Meißner K, Wehner F. A Component-based Approach for Adaptive, Dynamic Web Documents. Journal of Web Engineering. 2003 September 2003;2 (1&2):058-73.

Address for correspondence

Helma van der Linden
Medical Informatics
University Maastricht
POBOX 616
6200 MD Maastricht
The Netherlands
hvanderlinden@mi.unimaas.nl