Journal of Medical Systems manuscript No.
(will be inserted by the editor)

A Generative tool for building health applications driven by ISO 13606 archetypes

Marcos Menárguez-Tortosa · Catalina Martínez-Costa · Jesualdo Tomás Fernández-Breis

Received: date / Accepted: date

Abstract The use of Electronic Healthcare Records (EHR) standards in the development of healthcare applications is crucial for achieving the semantic interoperability of clinical information. Advanced EHR standards make use of the dual model architecture, which provides a solution for clinical interoperability based on the separation of the information and knowledge. However, the impact of such standards is biased by the limited availability of tools that facilitate their usage and practical implementation. In this paper, we present an approach for the automatic generation of clinical applications for the ISO 13606 EHR standard, which is based on the dual model architecture. This generator has been generically designed, so it can be easily adapted to other dual model standards and can generate applications for multiple technological platforms. Such good properties are based on the combination of standards for the representation of generic user interfaces and model-driven engineering techniques.

Keywords Archetypes · ISO 13606 · Medical Systems · Automatic generation of applications

Marcos Menárguez-Tortosa
Facultad de Informática, Universidad de Murcia, CP 30071, Spain
Tel.: +34-868884234
Fax: +34-868884151
E-mail: marcos@um.es

Catalina Martínez-Costa
Facultad de Informática, Universidad de Murcia, CP 30071, Spain
Tel.: +34- 868888787
Fax: +34-868884151
E-mail: cmartinezcosta@um.es

Jesualdo Tomás Fernández-Breis
Facultad de Informática, Universidad de Murcia, CP 30071, Spain
Tel.: +134-868884613
Fax: +34-868884151
E-mail: jfernand@um.es

1 Introduction

According to the news published in [El 70% de las primeras consultas en Primaria es susceptible de resolverse con las TIC(L
70% of first consultations in primary care could be reduced by increasing the use of information technologies. For instance, the development of applications related to mental health or chronic diseases could make patients self-sufficient in many situations, reducing waiting lists and costs. This has been one of the reasons of the increase in the investments in this sector in the last 20 years. During this period, the application of the information technologies to the health sector has gained relevance.

The semantic interoperability of clinical information is defined as the ability of information systems to exchange and understand clinical information regardless of the system in which it was created. This implies that the receiving system has to be able not only to understand that information but also to show it in a way clinicians can understand it. Since the late 80s, the achievement of clinical information interoperability has been the objective of many projects carried out in different countries [Chen et al(2009)Chen, Klein, Sundval, Karlsson, and Ahlfeldt,epSOS European eHealth project(Last accessed: Septe
Farzandipour et al(2011)Farzandipour, Ahmadi, Sadoughi, and Karimi irajk,Grams(2010),
Jian et al(2007)Jian, Hsu, Hao, Wen, Hsu, Lee, Li, and Chang,Lin et al(2010)Lin, Lin, Roan, and Yeh,
Palacio et al(2010)Palacio, Harrison, and Garets], and the digitalization of clinical records has been carried out in many hospitals and primary care centers.

As a result, several standards for representing and communicating the electronic healthcare record (EHR) such as ISO 13606 [ISO 13606 standard(Last accessed: September 2011)],
openEHR [OpenEHR specification(Last accessed: September 2011)] or HL7 [Health Level Seven(Last accessed: September 2
have been developed, and the use of such standards is considered essential in order to achieve the full interoperability of clinical information [Stroetmann et al(2009)Stroetmann, Kalra, Lewalle, Rector, Rodrig
Most of such standards involve a change in how clinical information systems are developed, since they follow the dual model architecture. The dual model architecture [Beale(2002)] is based on the definition of information and knowledge separately by defining two conceptual levels by using the reference and archetype models. They provide a way of specifying this clinical information by means of archetypes. Archetypes represent the minimal information unit that clinical information systems can exchange, thus being the basic semantic interoperability unit [Kalra and Archana(2008)]. Moreover, archetypes also include the context for interpreting clinical information. The importance of using EHR standards in order to achieve semantic interoperability has become evident in recent years. Dual model-based standards such as ISO 13606 or openEHR seem to be promising solutions for that purpose but, although many archetypes have been developed, there are limited tools that allow working with them. In this work we are interested in those tools that might help clinicians to use EHR standards in their daily working life.

However, most of current EHR systems do not use any EHR standard and, consequently, clinical information interoperability has not been achieved yet and patient clinical information remains unaccesible in clinical centers. A reason for that might be that the use of EHR standards for developing applications requires a deep understanding of such standards. Hence, one of the aims of the present work is to facilitate the use of EHR standards in the development of healthcare applications by generating software in an automatic manner.

In this paper, we present ArchForms, which is a generator of applications from archetypes. Archetypes are supposed to be developed by clinicians, so this generator would take as input the result of the work done by the experts and would provide them with a functional application. The functionality of the generated applications would not only be data input, but also data validation and generation of data extracts compliant with a particular dual model-based EHR standard. This means that the development process used in ArchForms can be applied for a series of standards, although the current implementation has been done for the European standard ISO 13606.

Hence, ArchForms is capable of generating web applications from ISO 13606 archetypes. The resulting web applications generate ISO 13606 compliant EHR extracts in XML format to exchange data with other systems and to facilitate the insertion of data in the already existing databases. The approach used for developing this web application generator does not only allow generating applications for a particular technology, but it can also be easily adapted to different software platforms, user interfaces and devices. This is possible because the process for generating the application transforms first the archetype into a generic representation of the graphical interface, and in further stages uses both the archetype and that graphical interface to generate a full application.

The structure of this paper is described next. In Section 2, some background information about ISO 13606 and related work on the generation of GUIs and applications from archetypes is provided. An overview of our approach is provided in Section 3. This approach is based on a series of phases, which are explained in Sections 4 and 5. The evaluation of the approach is presented in Section 6. Finally, some discussion and conclusions are presented in Section 7.

2 Background

In this section, we will introduce the basic concepts that will permit the understanding of our research work. First, we will describe the ISO 13606 EHR standard, for which the applications will be automatically generated. Then, we will describe the related work about the development of graphical interfaces and applications from archetypes.

2.1 ISO 13606

The ISO 13606 specification was proposed by the CEN/TC 251 Health Informatics Technical Committee. This standard intends to support interoperability between EHR systems and to provide components for enabling the interaction with EHR services. Its modelling approach is based on the dual model architecture.

That architecture was first introduced in the GEHR project [Good European Health Record (GEHR) project(Last access in the late 80s. It is based on the definition of two different conceptual levels, namely, information and knowledge. Each level uses a different data model. Thus, the information level uses the reference model and the knowledge level uses the archetype one. The reference model defines the set of entities that form the building blocks of the electronic healthcare record. The archetype model is used to define clinical concepts in the form of structured and constrained combinations of the entities contained in the reference model. These clinical concepts are named archetypes and are usually defined by using the Archetype Definition Language

(ADL) [OpenEHR Foundation. ADL language(Last accessed: September 2011)]. Archetypes can be used to define clinical concepts such as heart rate, a laboratory test, or a blood pressure measurement. An archetype can be defined as a specialization of an already existing archetype, can include another archetype fragment in it and can be put into a particular group of archetypes to be used for a specific purpose, often corresponding to a screen form, also known as *templates*. Archetypes constitute a standardized way of capturing clinical data according to the archetype model, which provides the context for interpreting clinical information.

The ISO 13606 reference model consists of a set of data structures and data types that are constrained by means of archetypes in order to define which clinical information has to be captured and how. Data structures organize the different parts of a clinical document. In this way, the patient's clinical information is usually defined by using a *COMPOSITION*, which is similar to a clinical form and several *COMPOSITION* entities are grouped into a *FOLDER*. *SECTIONs* represent the headings that make up a clinical form and inside each *SECTION*, clinical concepts are defined by means of *ENTRY* entities. Finally, these last entities are organized in terms of table, list, or tree structures by means of *CLUSTER* and *ELEMENT* entities. Each *ELEMENT* uses a data type for defining the meaning of a data value. There are six main categories of data types: primitive, constructed, textual, coded, quantity and temporal.

Figure 1 depicts an excerpt of an ISO 13606 archetype used to define an antenatal checkup. The root node of this archetype is a *COMPOSITION* (*COMPOSITION[at00010.1]*), which comprises two *SECTIONs* that represent two tests, General and Abdominal. In that figure, the Abdominal test *SECTION* (*SECTION[at0101]*) is shown. It defines two *ENTRY* entities that correspond to two screen forms. We will focus on the heart rate *ENTRY* concept (*ENTRY[at0028]*). This form is in charge of capturing the fetus heart rate measurement (*ELEMENT[at0029]*), the Instrument used to measure it (*ELEMENT[at0033]*) and its description (*ELEMENT[at0035]*), that is, whether it is Regular or Irregular.

2.2 Related work

The feasibility of generating user interfaces from archetypes has been already asserted in recent years. Next, some of the existing approaches that have proposed the use of archetypes for generating GUIs are described.

In [Schuler et al(2006)Schuler, Garde, Heard, and Beale], a study investigated the use of openEHR archetypes to automatically generate GUIs. The proposal was based on the use of the Mozilla XML User Interface Language (XUL) [XML User Interface Language(Last accessed: Septemb
and the results showed that the automatic generation of GUIs from openEHR archetypes was feasible. This approach generates screen forms for data input and validation, and uses web services in order to make the data persistent.

Commercial tools have implemented similar approaches for openEHR templates: eZ-HIS [Nardon et al(2008)Nardon, França, and Naves], developed by the Brazilian company Zilics, and the Template Designer tool included in the Ocean Knowledge Studio [OpenEHR Template Designer, Ocean(Last accessed: September 2011)]. The first one is a health information system implemented according to the openEHR standard. It proposes to generate forms conforming to the tree structure of the archetype and it allows to store the data extracts in relational databases or in

```
definition
  COMPOSITION [at00010.1] occurrences matches {1..1} matches { --Antenatal checkup
    content cardinality matches {2..2; ordered; unique} matches {
      ...
      SECTION [at0101] occurrences matches {1..1} matches { --Abdominal test
        members cardinality matches {1..3; ordered; unique} matches {
          ENTRY [at0028] occurrences matches {1..1} matches { -- Heart rate
            items cardinality matches {1..3; ordered; unique} matches {
              ELEMENT [at0029] occurrences matches {0..1} matches { --Fetus heart rate
                value matches {
                  PQ [at0030] occurrences matches {1..1} matches {
                    value matches {|120,180|}
                    units matches {
                      CS [at0031]  matches {--Heart rate units
                        codeValue matches {"{H.B.}/min"}
                        codingSchemeName matches {"UCUM"}}}
                    property matches {
                      CD [at0032]  matches {
                        codeValue matches {"249043002"} --Fetal heart rate
                        codingSchemeName matches {"SNOMED-CT"}}}
              }}}
              ELEMENT [at0033] occurrences matches {0..1} matches { --Instrument
                value matches {
                  CV [at0034] occurrences matches {1..1} matches {
                    displayName matches {/.*/}
                    codingSchemeName matches {SNOMED-CT}
                    codeValue matches {/.*/}
              }}}
              ELEMENT [at0035] occurrences matches {1..1} matches { --Heart rate description
                value matches {
                  CODED_TEXT [at0036] occurrences matches {1..1} matches {
                    codedValue matches {
                      CD [at0037] occurrences matches {0..1} matches { --Regular
                        displayName matches {Regular}
                        codingSchemeName matches {SNOMED-CT}
                        codeValue matches {289443009}}
                      CD [at0038] occurrences matches {0..1} matches { --Irregular
                        displayName matches {Irregular}
                        codingSchemeName matches {SNOMED-CT}
                        codeValue matches {289444003}}
              }}}}
  }}}}}}
```

Fig. 1 Excerpt of the ISO 13606 antenatal checkup archetype

XML. The second approach, the Template Designer, allows for defining the presentation layout from openEHR templates and generates the implementation as .NET forms.

In the open-source community, Opereffa [Opereffa(Last accessed: September 2011)] and EHRFlex [Brass et al(2010)Brass, Moner, Hildebrand, and Robles] feature web forms generation from archetypes based on Java technology. The former is specific to the openEHR standard and the latter supports the ISO 13606 one. Nevertheless, EHRFlex can be adapted to different dual-model standards by using an abstract reference model.

On the other hand, the generation of GUIs can also be approached as a preview function in archetype editors, such as the LIU archetype editor [LiU archetype editor(Last accessed: September 2011)] from the Swedish Linköping University. In that approach, GUIs are automatically obtained from openEHR archetypes and rendered using Java Swing.

In [der Linden et al(2009)der Linden, Austin, and Talmon], a two-model approach for specifying and generating screen representation for archetype-based information inspired by the two-model approach of archetypes is presented. The authors propose to distinguish two models: GUI and Content. The first model is platform-independent and is converted into a specific technology. The second one describes the content-related presentation knowledge. This proposal aims to model GUIs in a generic way and independently of an specific implementation platform. However, the approach is focused on EHR extract visualization.

In summary, most of them are only focused on the GUI generation and the data entry validation but they do not consider the generation of functional applications, which is one of the main objectives of our work. In those cases in which applications are generated, the systems seem to have been designed for a particular EHR standard.

3 Overview of the ArchForms architecture

The approach we present in this work makes possible to obtain applications from archetypes. Archetypes describe what and how clinical information should be captured. They are similar to clinical forms and, therefore, we propose to use them as input for building our clinical applications. As mentioned before in this paper, they are usually defined in ADL. This language is a concrete syntax of the archetype object model (AOM). In AOM, archetypes are defined independently of a reference model and this makes their processing harder for our purpose.

In [Fernández-Breis et al(2006)Fernández-Breis, Menárguez-Tortosa, Moner, Valencia-García, Maldonado, Vivancos-Vic we proposed a semantic representation for ISO 13606 and openEHR archetypes by using the Web Ontology Language (OWL) [OWL Web Ontology Language Reference(Last accessed: September 2011)] and in [Martínez-Costa et al(2009)Martínez-Costa, Menárguez-Tortosa, Fernández-Breis, and Maldonado] we provided a methodology for transforming automatically ADL archetypes into OWL. Using ontologies for representing archetypes provide us with several benefits, among which we must point out that the processing of such archetypes has become easier. In the AOM representation, concepts are represented by means of generic entities that refer to reference model concepts are related each other by means of string properties. An archetype expressed according to these ontologies will combine archetype and reference model concepts linked by means of ontology relationships. In this way, information will be structured in a more comprehensible way and its processing will be easier. In this work, such semantically enriched archetypes are used as input of the generation process.

The ArchForms architecture is then able to develop ISO 13606 compliant applications taking as input one or more semantic archetypes. It has been designed in a generic way, allowing the development of applications based on different technologies. Moreover, developers do not need to have a deep understanding of the ISO 13606 standard by means of the separation established between the standard interpretation process and the application code generation for a particular technology. The generative architecture of ArchForms, which is depicted in Figure 2, has two main phases: (1) generation of the generic GUI models; and (2) generation of the code for the target platform, which will be described in the next subsections.

On the technical side, model-driven engineering techniques play a fundamental role in both phases. They will provide us with a series of tools that will allow us to define a modular and easy to maintain method. Also, they will enable to obtain the technical artifacts that make the generation and control of the applications. In particular, archetypes and generic GUIs are represented as models and we have used in both phases the model to text transformation language MOFScript [MOFScript(Last accessed: September 2011)]. It makes possible defining a set of rules in which information from the representation of the archetype as a model and static text are combined. Therefore, our approach is based on two abstraction levels as suggested by the MDA Guide [Object Management Group(Last accessed: September 2011)] and as it has been performed in other domains such as [Beydoun et al(2009)Beydoun, Low, Mouraditis, and Henderson-Seller

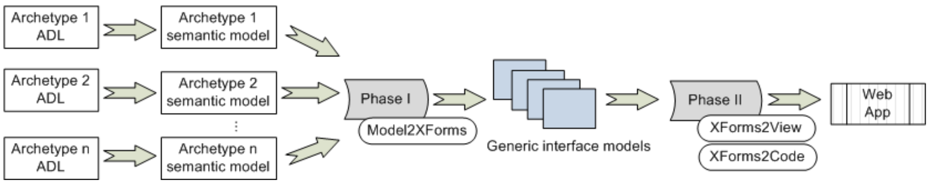


Fig. 2 Application development phases.

3.1 Overview of the generation of the GUI models

In order to generate the application forms, archetypes are represented as generic GUI models, independently of a particular interface technology. This requires the interpretation of the ISO 13606 standard.

For this purpose, XForms [XForms(Last accessed: September 2011)] has been used. This is an XML-based W3C recommendation for the generic representation of forms, independently of a specific device or user interface technology. It allows the separation of data logics (e.g. data validation) from presentation. XForms provides a set of generic controls for defining forms, similar to XHTML counterparts, but that omits specific device presentation features.

Therefore, the input of this process will be one or more semantic archetypes and the result will be their corresponding generic forms. The generation of the generic forms will be done by using the Model2XForms transformer, which defines a set of transformation rules that provides the mappings for the structural components, constraints and data types of the input archetype and their XForms representation. As a result, one or more XForms model are obtained.

The application of this technological approach for generating the GUI models will be explained in Section 4.

3.2 Overview of the code generation for a target platform

Once the generic GUI models are available, the source code of the application for a particular platform can be generated. Two transformers, namely, XForms2View and XForms2Code, will generate the specific application source code for a particular platform (see Fig. 3). Both of them are reusable and platform-independent since they define hooks that are completed by platform-specific modules. The first one, XForms2View, generates one web form for each generic GUI model obtained in phase one. The second transformer, XForms2Code, generates the classes that support the validation of data and the persistence of EHR extracts. This phase is better explained in Section 5.

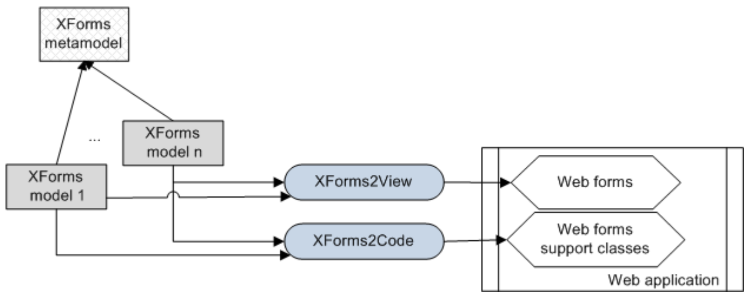


Fig. 3 Generation of the Web Applications

4 From ISO 13606 to Generic GUI Models

In this phase, a set of generic GUI models are obtained in XForms from one or more ADL archetypes. The latter will be represented as models according to the semantic representation mentioned in Section 3. First, we will describe how archetypes can be represented in XForms. Second, we will see how the transformation process has been implemented.

4.1 Generic GUI models

The generic GUI models aim at defining the different user interfaces obtained from archetypes in a generic way. In this work, an XForms form will be defined for each *ENTRY* entity of the archetype. This form will consist of two main parts: the model, in which the model elements are defined; and the definition of a set of generic controls.

We have defined a mapping between each data structure and type of the ISO 13606 reference model and XForms controls. Then, these controls are customized according to the archetype definition. Figure 4 depicts an excerpt of the definition of the control for the *ELEMENT[at0033]* of the antenal checkup archetype shown in Figure 1.

```
<group id="COMP_antenatal_at0033" bind="COMP_antenatal_at0033_bind">
  <hint id="COMP_antenatal_at0033_hint">
    <output value="Instrument for measuring the fetus heart rate"/>
  </hint>
  <label><output value="instrument"/></label>
  <group id="COMP_antenatal_at0034" bind="COMP_antenatal_at0034_bind" >
    <input id="COMP_antenatal_at0034_displayName" bind="COMP_antenatal_at0034_displayName_bind">
      <label><output value="Name"/></label>
    </input>
    <label><output value="Code Schema"/></label> <label><output value="SNOMED-CT" /></label>
    <input id="COMP_antenatal_at0034_codeValue" bind="COMP_antenatal_at0034_codeValue_bind">
      <label><output value="Coded Value"/></label>
    </input>
  </group>
</group>
```

Fig. 4 Excerpt of the XForms control definition for the Instrument concept

The fetus heart rate is measured by an instrument which is represented by using a Coded Value type. It is defined by means of two *GROUP* controls: one that represents the *ELEMENT* entity; and one that groups its value definition given by the attributes *displayName*, *codingSchemeName* and *codeValue*. The attributes *displayName* and *codeValue* are represented by means of *INPUT* controls and they correspond to a human-readable description of a concept, and to a code from the coding system indicated in the attribute *codingSchemeName*, respectively. Note that the attribute *codingSchemeName* is shown as a label, since the archetype constrains its value to "SNOMED-CT", that is, the attribute is not editable.

If we look their definition in depth, each *INPUT* defines the properties *id*, *bind* and *label*. The *id* identifies uniquely the control, the *label* contains the *OUTPUT* control that defines a text to be displayed in the form, and *bind* defines the relationship between the control and a data model element. This last property is defined in all XForms controls and will allow us to link them to the data model.

In this way, a set of *BIND* tags is defined in the model part. These tags provide the corresponding validity information for each control defined. Figure 5 shows an example of a bind that constrains the *codeValue* attribute and forces it to be a string conforming a specific pattern (pattern=.*). Besides, the control is mandatory (required="true") and the *nodeset* attribute identifies the model data node that is constrained.

```
<model>
...
  <bind id="COMP_antenatal_at0034_codeValue_bind" nodeset="/COMP_antenatal_at0034_codeValue"
    required="true" type="xs:string"  constraint="pattern=.*"/>
...
</model>
```

Fig. 5 Excerpt of the relationship established between the data model elements and the XForms controls for the Instrument concept

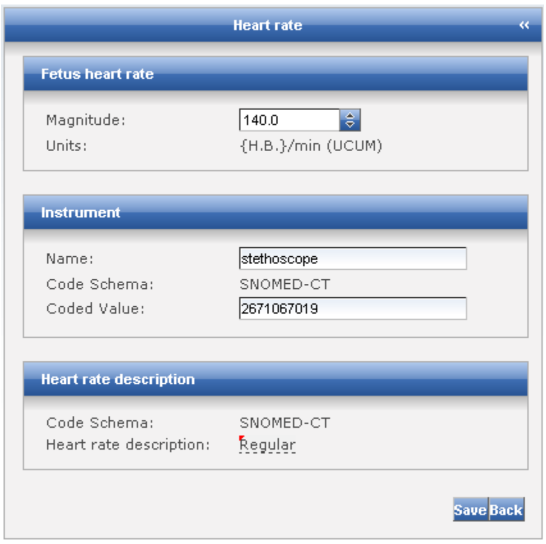
The other ISO 13606 data structures and types will be defined in a similar way. We have already mentioned that an XForm file will be defined for each *ENTRY* entity of the archetype. The other upper-level structures, namely, *FOLDER*, *COMPOSITION* and *SECTION*, will be used to organize the clinical forms of the application.

4.2 Example

Figure 6 depicts a graphical representation of the application generated for the "heart rate" section of the antenatal checkup archetype partly shown before. It includes three *ELEMENTs*: "Fetal heart rate", "Instrument" and "Heart rate description".

In the figure, the "Fetal heart rate" is represented by using a quantity type, and the "Heart rate description" and the "Instrument" are represented by using a coded data type. The "Instrument" has already been explained. The "Fetal heart rate" is represented by the Physical Quantity type and it is defined by the attributes value, unit, property and precision. The first three ones represent, respectively, the magnitude of the quantity measured, the measurement unit, and the name of the property measured according to a particular coding system. The last attribute, namely, precision, stands for the precision associated with the measurement and it is not represented graphically. Consequently, its associated constraints are applied to the value field. In the example, the heart rate magnitude is editable and restricted so that it has a value within an interval ([120-180]). The unit is defined as (H.B./min., heart beats per minute) and the name of the property measured takes the SNOMED-CT value corresponding to (fetal heart rate). These last two properties are represented as codes in particular coding systems and are fixed by the archetype. The unit is only shown for improving the usability of the interface.

Finally, the "Heart rate description" is represented by using a Coded Text type. In the archetype it is defined as two codes from the SNOMED-CT coding system (Regular, Irregular). Thus, the form includes a selection list.



Fetus heart rate

Magnitude:	140.0
Units:	{H.B.}/min (UCUM)

Instrument

Name:	stethoscope
Code Schema:	SNOMED-CT
Coded Value:	2671067019

Heart rate description

Code Schema:	SNOMED-CT
Heart rate description:	Regular

Fig. 6 Partial view of an application for antenatal checkup

5 Generation of the Web Applications

In this phase, the source code of the web application is generated for a particular implementation technology. The generic GUI models from the previous step are transformed into web forms according to a specific user interface technology and the application source code is developed by using the controls and restrictions from the XForms models. That source code is in charge of validating and managing data for a particular implementation technology and also for managing some application resources such as some configuration files.

The web applications generated by ArchForms allow for the creation, edition and visualization of ISO 13606 compliant extracts. The architecture of such applications is based on the model-view-controller pattern (see Fig. 7) that isolates the business logic from the input and presentation layers. It consists mainly of the following components:

– Controllers: They implement the business logics of the application. There is a main controller and secondary ones. The main controller represents the root concept of the archetype and secondary controllers are created for each ENTRY restricted by the archetype. Among others, they are responsible for implementing the validation of the constraints defined for the archetype data structures and types, and for the generation of EHR extracts.
– Views: They are the GUIs that will be exploited by the users and are built from the XForms models.
– Model: It contains the ISO 13606 Java classes used for the EHR extract generation.

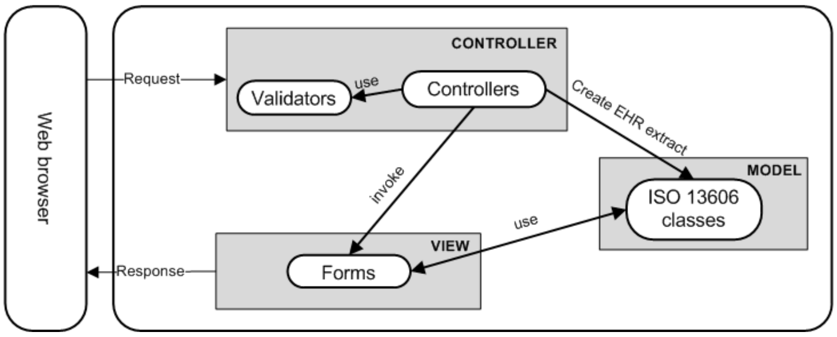
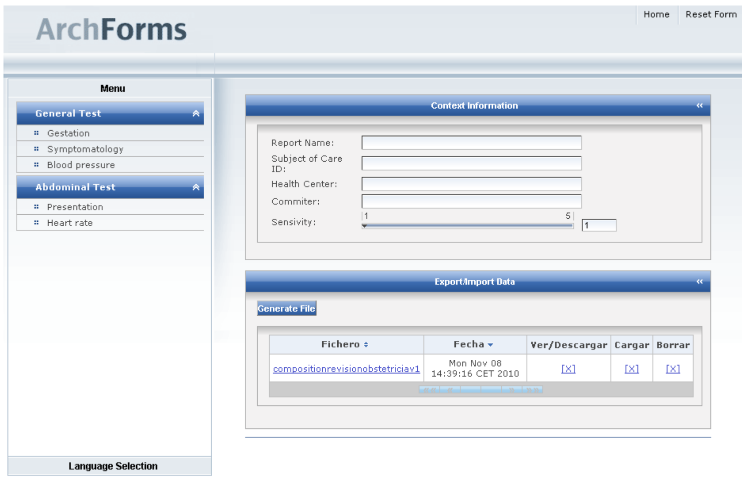


Fig. 7 Architecture of the ArchForms applications

The user interface of the applications has three main functional components:

– Forms: The web forms that are used for editing the clinical information as constrained by the archetypes.
– Navigation menu: Hierarchical visualization of the structure of the archetypes, providing access to the forms.
– EHR Extract Manager: Friendly visualization and load of EHR extracts.

The first component can be observed on Figure 6. The Navigation menu and the EHR Extract Manager is shown in Figure 8.



Report Name:
Subject of Care ID:
Health Center:
Commiter:
Sensivity: 1 5 1

Fichero	Fecha	Ver/Descargar	Cargar	Borrar
compositionrevisionobstetriciav1	Mon Nov 08 14:39:16 CET 2010	[X]	[X]	[X]

Fig. 8 View of the extract manager of the neonatal checkup application

5.1 Implementation

By using the XForms models previously generated, two MOFScript transformers have been developed (see Fig. 3): XForms2View and XForms2Code. The first one generates the implementation of the views for a specific framework. It transforms the generic controls in XForms into the corresponding ones in the target implementation framework. The second transformer, XForms2Code, generates from the XForms model the classes that support the validation of the data input by the users and the persistence of the extracts. These classes contain all the elements needed for representing the data structures and constraints of the input archetypes. In order to define these transformers as platform independent modules, they define hooks that are implemented for specific technologies.

Figure 9 depicts how both transformers can be used for generating the source code for three different implementation technologies: Seam, PrimeFaces and TouchFaces. For each one of them, the three following modules are respectively defined: Codegen_Seam, CodeGen_Prime and CodeGen_Touch. All the implementations generate web applications based on Java Server Faces (JSF) and Java technology but differ in the framework or user interface technology used:

– ArchFormsSeam: It uses RichFaces [RichFaces site(Last accessed: September 2011)] for the user interface and the Seam framework [The Seam Framework(Last accessed: September 2011)] for the implementation.
– ArchFormsPrime: It uses Primefaces library [PrimeFaces site(Last accessed: September 2011)].
– ArchTouch: It uses TouchFaces, a mobile user interface toolkit to create mobile web applications for handheld devices with WebKit [The WebKit open-source project(Last accessed: September 2011)] based browsers such as iPhone, Android, etc.

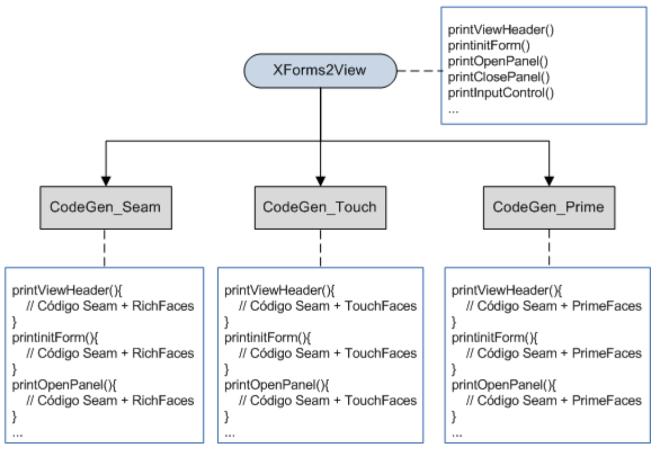


Fig. 9 Generation logics of the XForms2View transformer for the three implementation technologies

6 Evaluation

The ArchForms generator has been applied for generating a series of archetype-based web applications, which has been useful for the technical validation of the approach. The applications can be generated from a series of input archetypes and they are capable of importing and exporting XML extracts. At the moment, ArchForms applications can be generated for three Java-based frameworks: Seam, PrimeFaces and TouchFaces. Examples of the resulting applications can be found and used online [Examples of applications generated with ArchForms(Last accessed: September 2011)].

A testbed application has also been developed which includes all applicable restrictions to the classes of the reference model. These applications have also been evaluated in terms of usability and compatibility with web browsers. Some of the usability aspects considered have been: the interface clarity and legibility, the use of messages, the easy navegability of the forms, the application internationalization and the visualization of the EHR extracts. We have tested the following web browsers: Mozilla Firefox 3.X and 4.X, Safari 4.X and 5.X, Internet Explorer and Google Chrome. In all of them, the generated applications were completely functional and their visualization was correct. During the ArchForms implementation, validation tests have also been performed with the W3C Markup Validation Service [W3C Markup Validation Service(Last accessed: September 2011)] in order to check the correctness of the generated web documents.

Special attention has been paid to the usability and compatibility of the web applications based on TouchFaces. They can be used in mobile devices with tactile screen that use WebKit based browsers. These applications have been tested for Android and iPhone. Figure 10 shows the corresponding application example for iPhone for the archetype presented in Figure 6.

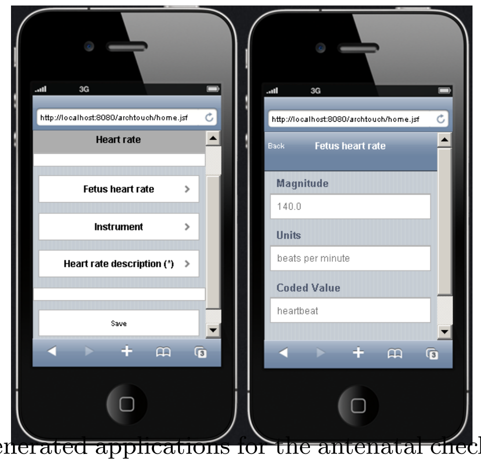
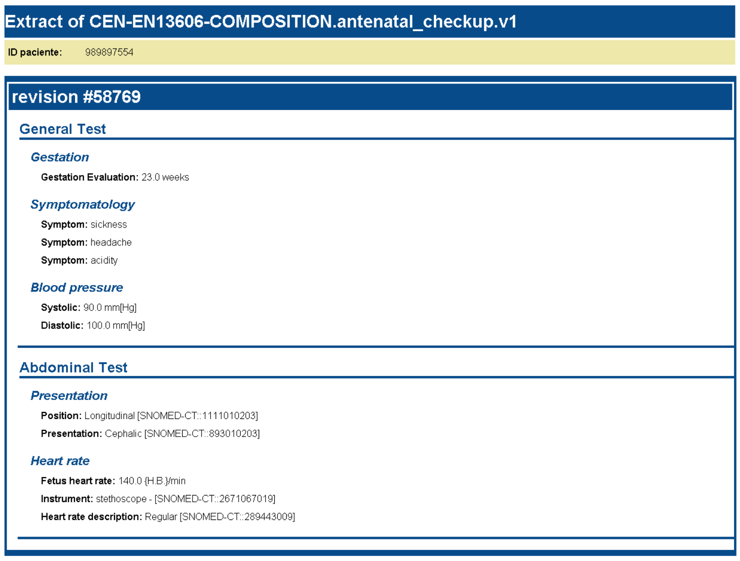


Fig. 10 Neonatal checkup iPhone application.

Note that the generated applications for the antenatal checkup archetype share the functionality but they differ in how the user interacts with the particular device. These applications make possible the generation of XML extracts from the input data. An example of such XML extract generated for a simulated revision of a patient is shown in Figure 11. In order to make its visualization easier, an XSL stylesheet is used.


Extract of CEN-EN13606-COMPOSITION.antenatal_checkup.v1

ID paciente: 989897554

revision #58769

General Test

Gestation

Gestation Evaluation: 23.0 weeks

Symptomatology

Symptom: sickness
Symptom: headache
Symptom: acidity

Blood pressure

Systolic: 90.0 mm[Hg]
Diastolic: 100.0 mm[Hg]

Abdominal Test

Presentation

Position: Longitudinal [SNOMED-CT::1111010203]
Presentation: Cephalic [SNOMED-CT::893010203]

Heart rate

Fetus heart rate: 140.0 {H.B.}/min
Instrument: stethoscope - [SNOMED-CT::2671067019]
Heart rate description: Regular [SNOMED-CT::289443009]

Fig. 11 Graphical view of the data extract for the neonatal checkup application.

7 Discussion and Conclusions

The use of EHR standards in the development of healthcare applications is crucial for achieving the semantic interoperability of clinical information. The EHR standards that follow the dual model architecture, such as ISO 13606 or openEHR, provide a solution for clinical interoperability and here we propose to use them for generating clinical applications.

Archetypes constitute a standardized way of capturing clinical information and it is possible to use them for generating graphical user interfaces. Our approach, ArchForms, provides an architecture that enables the generation of ISO 13606 applications for different implementation technologies. Nowadays, there are no official organizations in charge of verifying the ISO 13606 compliance and there is no official XML Schema. So far, there is only one publicly available XML Schema for the ISO 13606, which can be found at the website of the ISO 13606 association [EN 13606 Association(Last accessed: September 2011)]. Therefore, the extracts of the ArchForms applications are compliant with such schema.

The generation of an application consists of two phases: (1) generation of the generic GUI models; and (2) generation of the code for the target platform. The generation process has a previous transformation step, through which the ADL archetype is transformed into a semantic archetype by applying methods developed by our group that have been reported in previous work.

In the first generation phase, a set of XForms models are generated from each ENTRY concept of the archetype, independently from the specific user interface implementation technology. They represent each web form of the application. In the second phase, the source code for an specific implementation platform is generated. Thanks to the generic design of this process, developers do not need to have a deep understanding of the ISO 13606 standard by means of the separation established between the standard interpretation process and the application code generation for a particular technology.

As it has been described in section 2.2, there are other related works. Table 1 compares some of their approaches according to the following features: (1) the ability of generating GUIs from archetypes, (2) the implementation of data entry validation, (3) the generic GUI modeling, (4) the ability of generating functional applications, (5) the ability of being adapted to be used with other EHR standards and (6) the ability of generating EHR extracts.

Table 1 Table showing some of the existing approaches for GUI generation from archetypes: C1(GUI Generation); C2(Data Validation); C3(Generic GUI Modeling); C4(Full Application Generation); C5(Several EHR standards); and C6(Extracts Generation)

	C1	C2	C3	C4	C5	C6
Schuler et. al [Schuler et al(2006)Schuler, Garde, Heard, and Beale]	x	x				
Van der Linden et. al [der Linden et al(2009)der Linden, Austin, and Talmon]	x	x	x		x	
Ocean Informatics [OpenEHR Template Designer, Ocean(Last accessed: September 2011)]	x	x				
Opereffa [Opereffa(Last accessed: September 2011)]	x	x		x		
EHRFlex [Brass et al(2010)Brass, Moner, Hildebrand, and Robles]	x	x		x	x	
LIU EHR GUI [LiU archetype editor(Last accessed: September 2011)]	x					
eZ-HIS [Nardon et al(2008)Nardon, França, and Naves]	x	x				x
ArchForms	x	x	x	x	x	x

As can be inferred from the table, most approaches are only focused on the GUI generation and the data entry validation but they do not consider the generation of clinical applications [Schuler et al(2006)Schuler, Garde, Heard, and Beale, Nardon et al(2008)Nardon, França, and Naves, OpenEHR Template Designer, Ocean(Last accessed: September 2011), LiU archetype editor(Last accessed: September 2011), der Linden et al(2009)der Linden, Austin, and Talmon]. Nearly all of them generate GUIs from archetypes for a specific implementation technology, but the approach presented in [der Linden et al(2009)der Linden, Austin, and Talmon] proposes to generate them in a generic way as we do here. Moreover, most of them are focused on a specific EHR standard and can not be easily adapted to other standards. Only ArchForms and the approaches presented in [Brass et al(2010)Brass, Moner, Hildebrand, and Robles, der Linden et al(2009)der Linden, Austin, and Talmon] offer this feature. There are two proposals that allow also the generation of full clinical applications from archetypes [Opereffa(Last accessed: September 2011),Brass et al(2010)Brass, Moner, Hildebrand, and Robles], but they do not have been designed in a generic way for making possible implementations for different platforms.

We have generated web applications for three Java-based frameworks: Seam, Primefaces and Touchfaces. Therefore, we have shown that the generative architecture proposed has been designed in a generic enough way to be adaptable to different implementation technologies. Next, we analyze how ArchForms has to be extended to generate applications for different implementation technologies.

The Model2XForms transformer generates a set of generic GUI models. This transformer can be reused for creating new versions of the generator, since the generated XForms models are generic. Reusing the Model2XForms transformer allows the developer to generate the code based on generic components such as group panel, check box, data entry field, etc. rather than on ISO 13606 elements.

The XForms2View transformer generates the code for the views for a particular UI technology from the XForms models. Its design is based on the use of generic methods for the components to be generated. The concrete implementation of such methods is provided by a module. This transformer is highly reusable as long as the target implementation framework allows the creation of views based on components and controls, and it provides a clear separation between the views and the rest of layers.

The XForms2Code transformer is responsible for generating the structure of the classes that provide support to the web application forms. That structure will depend on the structure of the archetype used for the generation. This transformer is less generic than the previous ones and it would probably need an adaptation in case a very different implementation technology is used. In this case, we are not handling a generic metamodel for the structure of the classes of the web application, but we take advantage of the complete definition of XForms to generate a set of classes symmetric to the structure of the view components. In this way, that set of classes can be used as support classes for the forms. The capacity of the new framework for associating the structure of the classes to the views without needing additional classes can be defined as the limitation for reusing this transformer.

ArchForms applications use XML files as a persistence mechanism. This permits the generation of applications that can be adapted to the persistence mechanism used in a particular healthcare institution. On the other hand, the mappings between the archetype and the XForms controls have been generically defined. In some cases, it might be necessary to do some customizations, such as expanding some text field or hiding some specific attributes. This could be achieved by using a configuration file.

Moreover, templates processing will be added to the system to configure the different components of archetypes for which the application has to be generated. On the other hand, connecting the generated applications to terminology servers is also a future research objective. The current version of the system generates a web application based on Java technologies. By using the ArchForms architecture, implementations for other technologies could also be developed.

ArchForms applications are not intrusive with health information systems already available in healthcare institutions, since only a Java application server is needed to execute them. The integration of ArchForms applications and such existing systems is through the data extracts.

In summary, we have presented a solution for the automatic generation of web applications using ISO 13606 archetypes. The adaptation of ArchForms in order to generate web applications for other dual-model based standards such as openEHR could be easily performed by defining the corresponding representation of its clinical data types and structures in XForms.

Acknowledgements This work has been possible thanks to the Spanish Ministry for Science and Education through grant TIN2010-21388-C02-02

Conflict of Interest

The authors declare that they have no conflict of interest.

References

Beale(2002). Beale T (2002) Archetypes: Constraint-based Domain Models for Future-proof Information Systems. In: Edited by Kenneth Baclawski and Haim Kilov Northeastern University, Boston (ed) Eleventh OOPSLA Workshop on Behavioral Semantics: Serving the Customer, pp 16–32

Beydoun et al(2009)Beydoun, Low, Mouraditis, and Henderson-Sellers. Beydoun G, Low G, Mouraditis H, Henderson-Sellers B (2009) A security-aware metamodel for multi-agent systems. Information and Software Technology 51(5):832–845

Brass et al(2010)Brass, Moner, Hildebrand, and Robles. Brass A, Moner D, Hildebrand C, Robles M (2010) Standardized and Flexible Health Data Management with an Archetype Driven EHR System (EHRflex). In: Seamless Care-Safe Care. The Challenges of Interoperability and Patient Safety in Health Care. Proceedings of the EFMI Special Topic Conference, IOS Press BV, Amsterdam, pp 212–218

Chen et al(2009)Chen, Klein, Sundval, Karlsson, and Ahlfeldt. Chen R, Klein G, Sundval E, Karlsson D, Ahlfeldt H (2009) Archetype-based conversion of ehr content models: pilot experience with a regional ehr system. BMC Medical Informatics and Decision Making 9:33

El 70% de las primeras consultas en Primaria es susceptible de resolverse con las TIC(Last accessed: September 2011). El 70% de las primeras consultas en Primaria es susceptible de resolverse con las TIC (Last accessed: September 2011) URL `http://directivos.publicacionmedica.com/noticia/el-70-de-las-primeras-consultas-en-primaria-es-susceptible-de-resolverse-con-las-tic`

EN 13606 Association(Last accessed: September 2011). EN 13606 Association (Last accessed: September 2011) URL `http://www.en13606.eu`

epSOS European eHealth project(Last accessed: September 2011). epSOS European eHealth project (Last accessed: September 2011) URL `http://www.epsos.eu/`

Examples of applications generated with ArchForms(Last accessed: September 2011). Examples of applications generated with ArchForms (Last accessed: September 2011) URL `http://miuras.inf.um.es/~researchehr/archforms.html`

Farzandipour et al(2011)Farzandipour, Ahmadi, Sadoughi, and Karimi irajk. Farzandipour M, Ahmadi M, Sadoughi F, Karimi irajk I (2011) Adopting confidentiality principles for electronic health records in iran: A delphi study. Journal of Medical Systems 35:333–343, URL `http://dx.doi.org/10.1007/s10916-009-9370-x`, 10.1007/s10916-009-9370-x

Fernández-Breis et al(2006)Fernández-Breis, Menárguez-Tortosa, Moner, Valencia-García, Maldonado, Vivancos-Vicente, Miranda-Mena, and Fernández-Breis JT, Menárguez-Tortosa M, Moner D, Valencia-García R, Maldonado JA, Vivancos-Vicente PJ, Miranda-Mena T, Martínez-Béjar R (2006) An Ontological

Infrastructure for the Semantic Integration of Clinical Archetypes. Lecture Notes in Computer Science 4303:156–167

Good European Health Record (GEHR) project(Last accessed: September 2011). Good European Health Record (GEHR) project (Last accessed: September 2011) URL `http://www.chime.ucl.ac.uk/work-areas/ehrs/GEHR/index.htm`

Grams(2010). Grams R (2010) The Obama EHR Experiment. Journal of Medical Systems pp 1–6, URL `http://dx.doi.org/10.1007/s10916-010-9559-z`, 10.1007/s10916-010-9559-z

Health Level Seven(Last accessed: September 2011). Health Level Seven (Last accessed: September 2011) URL `http://www.hl7.org`

ISO 13606 standard(Last accessed: September 2011). ISO 13606 standard (Last accessed: September 2011) URL `http://www.iso.org/iso/home.htm`

Jian et al(2007)Jian, Hsu, Hao, Wen, Hsu, Lee, Li, and Chang. Jian W, Hsu C, Hao T, Wen H, Hsu M, Lee Y, Li Y, Chang P (2007) Building a portable data and information interoperability infrastructure - framework for a standard Taiwan Electronic Medical Record Template. Computer Methods and Programs in Biomedicine 88(2):102–111

Kalra and Archana(2008). Kalra D, Archana T (2008) The EHR and Clinical Archetypes: time for clinical engagement! In: eHealth Planning and Management Symposium, Copenhagen, Denmark

Lin et al(2010)Lin, Lin, Roan, and Yeh. Lin CH, Lin IC, Roan JS, Yeh JS (2010) Critical factors influencing hospitals adoption of hl7 version 2 standards: An empirical investigation. Journal of Medical Systems pp 1–10, URL `http://dx.doi.org/10.1007/s10916-010-9580-2`, 10.1007/s10916-010-9580-2

der Linden et al(2009)der Linden, Austin, and Talmon. der Linden HV, Austin T, Talmon J (2009) Generic screen representations for future-proof systems, is it possible?: There is more to a GUI than meets the eye. Computer Methods and Programs in Biomedicine 95(3):213 – 226

LiU archetype editor(Last accessed: September 2011). LiU archetype editor (Last accessed: September 2011) URL `http://www.imt.liu.se/mi/ehr/tools/`

Martínez-Costa et al(2009)Martínez-Costa, Menárguez-Tortosa, Fernández-Breis, and Maldonado. Martínez-Costa C, Menárguez-Tortosa M, Fernández-Breis JT, Maldonado JA (2009) A model-driven approach for representing clinical archetypes for Semantic Web environments. Journal of Biomedical Informatics 42(1):150 – 164

MOFScript(Last accessed: September 2011). MOFScript (Last accessed: September 2011) `http://www.eclipse.org/gmt/mofscript/`

Nardon et al(2008)Nardon, França, and Naves. Nardon F, França T, Naves H (2008) Construção de Aplicações em Saúde Baseadas em Arquétipos. In: XI CBIS

Object Management Group(Last accessed: September 2011). Object Management Group (Last accessed: September 2011) Model driven architecture guide. URL `http://www.omg.org/cgi-bin/doc?omg/03-06-01`

OpenEHR Foundation. ADL language(Last accessed: September 2011). OpenEHR Foundation ADL language (Last accessed: September 2011) URL `http://www.openehr.org/releases/1.0.1/architecture/am/adl2.pdf`

OpenEHR specification(Last accessed: September 2011). OpenEHR specification (Last accessed: September 2011) URL `http://www.openehr.org/home.html`

OpenEHR Template Designer, Ocean(Last accessed: September 2011). OpenEHR Template Designer, Ocean (Last accessed: September 2011) URL `http://wiki.oceaninformatics.com/confluence/display/TTL/Template+Designer`

Opereffa(Last accessed: September 2011). Opereffa (Last accessed: September 2011) URL `http://opereffa.chime.ucl.ac.uk/introduction.jsf`

OWL Web Ontology Language Reference(Last accessed: September 2011). OWL Web Ontology Language Reference (Last accessed: September 2011) URL `http://www.w3.org/tr/owl-ref/`

Palacio et al(2010)Palacio, Harrison, and Garets. Palacio C, Harrison J, Garets D (2010) Benchmarking electronic medical records initiatives in the us: a conceptual model. Journal of Medical Systems 34:273–279, URL `http://dx.doi.org/10.1007/s10916-008-9238-5`, 10.1007/s10916-008-9238-5

PrimeFaces site(Last accessed: September 2011). PrimeFaces site (Last accessed: September 2011) URL `http://www.primefaces.org/`

RichFaces site(Last accessed: September 2011). RichFaces site (Last accessed: September 2011) URL `http://www.jboss.org/richfaces`

Schuler et al(2006)Schuler, Garde, Heard, and Beale. Schuler T, Garde S, Heard S, Beale T (2006) Towards automatically generating graphical user interfaces from openEHR archetypes. Stud Health Technol Inform 124:221–226, URL http://view.ncbi.nlm.nih.gov/pubmed/17108529

Stroetmann et al(2009)Stroetmann, Kalra, Lewalle, Rector, Rodrigues, Stroetmann, Surjan, Ustun, Virtanen, and Zanstra. Stroetmann VN, Kalra D, Lewalle P, Rector A, Rodrigues JM, Stroetmann KA, Surjan G, Ustun B, Virtanen M, Zanstra PE (2009) Semantic Interoperability for Better Health and Safer Healthcare. Deployment and Research Roadmap for Europe. ISBN-13 : 978-92-79-11139-6

The Seam Framework(Last accessed: September 2011). The Seam Framework (Last accessed: September 2011) URL http://seamframework.org/

The WebKit open-source project(Last accessed: September 2011). The WebKit open-source project (Last accessed: September 2011) URL http://www.webkit.org/

W3C Markup Validation Service(Last accessed: September 2011). W3C Markup Validation Service (Last accessed: September 2011) URL http://validator.w3.org/

XForms(Last accessed: September 2011). XForms (Last accessed: September 2011) URL http://www.w3.org/TR/xforms11/

XML User Interface Language(Last accessed: September 2011). XML User Interface Language (Last accessed: September 2011) URL https://developer.mozilla.org/en/xul