Development of an Ubiquitous Decision Support System for Clinical Guidelines using MDA *

Ivan Porres[1], Eladio Domínguez[2], Beatriz Pérez[2], Áurea Rodríguez[2] and María A. Zapata[2]

[1] Department of Computer Science, Åbo Akademi University,
Joukahaisenkatu 3-5 B, FIN-20520 Turku, Finland
iporres@abo.fi

[2] Dpt. of Computer Science and Systems Engineering, University of Zaragoza,
Pedro Cerbuna 12, E-50009 Zaragoza, Spain
{noesis,beaperez,arv852,mazapata}@unizar.es

1 Introduction

The health sector uses clinical guidelines and protocols (we will refer to both terms as "guidelines") as help instruments for decision making. These establish the interventions that must be carried out with regard to the current circumstances of a patient. Guidelines can be represented using many different approaches and notations [1], of which the most common one is natural language. In some cases, informal notations can be lead to ambiguities, inconsistencies or difficulties for understanding by health professionals. In order to avoid these problems, we consider that the guidelines must be specified by means of easily understandable formal notations which can be analyzed and implemented. We consider that the implementation of guidelines in an information system can be of great help to the medical staff in their decision making process [1], [2].

In this paper, we discuss the development of an ubiquitous decision support system (UDSS) for guidelines in a MDA context. Such system should take into account the indications of the guideline, the current status of the patient and should provide the necessary information to help physicians in their decisions. Considering the characteristics of the hospital environment, we think that such a system must be developed in an ubiquitous setting [1]. Additionally, the UDSS will record the information related with the application of the guideline to the patient [1], [2].

The manual implementation of a UDSS for guidelines would be a long and costly endeavor due to the vast amount of guidelines, their possibility of change over time and their dependence on the hospital which uses them. So, we have considered more appropriate to construct such system using a model-based development approach. We have decided, as is shown in [1], [2], to represent guidelines using UML Statecharts [9]. Based on this model representation, we developed a code generation tool to obtain the UDSS for every guideline in an automatic

* This work has been partially funded by the Spanish Ministry of Education and Science through the project TIN2005-05534 and the FPU grant AP2003-2713.

way. Using our approach, the implementation of new guidelines only requires the design of a new statechart modeling it. Based on this model, the UDSS for the guideline is generated automatically. In this way, the cost of generating the UDSS is lower than the required for a manual implementation.

It is worth noting that, to our knowledge, this is the first work to use the MDA approach for the development of a UDSS for guidelines. For this reason our approach would open a new field for the application of the MDA approach.

2 Using UML Statecharts to Represent Clinical Guidelines

We have decided to use UML 2.0 Statecharts [9] for representing the dynamics of guidelines [2] for several reasons. They allow the user to properly model concurrent behaviors, as well as to represent state hierarchy, increasing both scalability and legibility. In addition, UML Statecharts are considered to be easy to communicate and an understandable visual formalism [3]. Another advantage of using UML is that it has been supported by many modeling tools. In particular, many of such UML tools comply with the MDA approach, giving facilities for code generation. Moreover, the UML subset defining statecharts has been studied and formalized by many researchers, who have also developed analysis tools [6].

Furthermore, the implementation of the UML Statecharts allows the UDSS to show the information related to the guideline, such as patient's states, the possible events that can take place, the clinical conditions to be fulfilled when each event is triggered, the clinical actions to be carried out, as well as the corresponding new patient's state. In addition, this implementation helps to record a trace of the application of the guideline, such as physician's decisions and patient's states. This information will be stored to be part of the patient's clinical history and to be used for obtaining a summary concerning the application of the guideline to the patient, for physician's future decisions or legal support, etc.

As for the use of UML Statecharts for representing guidelines, we want to highlight two contributions of our approach. 1) We have established several *representation patterns* to assist in the modelization process because there is no single criteria to follow to create a statechart representing a guideline. We do not describe these patterns in this paper due to space reasons. 2) We have had to define our own *action language* for representing guidelines, due to UML does not have one single and precise action semantics [5]. Moreover, UML proposes a wide variety of specialized actions, which we want to avoid, because of our intention of representing guidelines in an easily way.

3 Development of a UDSS

In this section we discuss the process we propose to develop an ubiquitous decision support system for a given guideline, based on a MDA approach [8].

Once we have modeled a guideline as a UML statechart, the next step consists of using a specific MDA tool for code generation. There are a large amount of

MDA based tools, but most of them provide support for code generation based on their own particular transformation approach. In our case, we require a specific approach of model to text transformation that takes into account the specific semantics of the statecharts which represent the guidelines. Because of that, we have chosen a tool with support for customizable model to text transformation. The idea is to define only one set of transformations for all guidelines.

In order to finally obtain the UDSS for the guideline, we propose the use of the generated code from the statechart model, combined with a platform library. The aim of the platform library is to provide the standard services that are common to all guidelines. The generated code has invocations to the methods of this library in order to make use of its services.

The definition of both, the transformations and the platform library, is independent of the used guideline. So, following our approach, if the definition of the guideline is changed, it will be only necessary to modify the statechart which represents the guideline, making it easy to obtain the UDSS for that guideline, without the necessity of modifying the code manually.

Customization of the Model to Text Transformations. Among the large amount of MDA based tools, we have chosen the MOFScript Eclipse plug-in [7], which has support for customizable model to text transformation. As input models, MOFScript can use any model in any language based on the EMF [4] metamodel. From these input models, the tool can then generate any arbitrary text by using a defined set of MOFScript transformations. Each of these transformations contains rules which define the behavior of the transformation.

In our particular case, we use the UML 2.0 metamodel and the statechart which represents the guideline as the model. The code generator is defined as an only set of transformations in the MOFScript language. There are two kinds of transformation rules: (1) those which traverse the model (taking into account the different elements in a UML statechart and their specific semantics) and collect the information in it and (2) those who generate actual code.

As we have commented previously, we have defined our own action language as part of our representation patterns for guidelines. Taking into account the possibility of adding new elements to that language in the future, we have defined a method which allows us to generate code from the elements of that action language in an easily way. We do not comment that method due to space reasons.

Development of the Platform Library. The aim of using the platform library is to make the UDSS more modular. We achieve this by defining the programming interfaces of the library general enough for being used independently of the platform chosen. This library has different methods, related with the implementation of the UDSS, such as the user interface, the strategy to store the data, etc. The implementation of those methods will depend on the platform we want to use. The generated code has invocations to those methods so that, in order to change the platform, it will be necessary to modify neither the defined transformations nor the generated code. That is, we only have to modify the implementation of those methods.

In addition, we want to develop the decision support system so that it can be accessed in an ubiquitous way, through various kinds of interaction devices. So our intention is to implement the library in such a way that it can provide support for the necessary ubiquitous services, in order to obtain the UDSS with the characteristics that we have explained previously.

4 Conclusions and Future Work

In this paper we have shown a model-based development approach to construct a UDSS for guidelines. Our approach makes it possible to automatically obtain the UDSS for a guideline represented by a statechart.

At present, we have developed a first prototype using a simple user interface, so as future work we have to modify the implementation of the methods defined in the platform library in order to use another more suitable way of interaction with the user. In addition, the development of a way for recording the information related to the application of the guideline to the patient and the determination of evolution mechanisms to manage changes in the definition of the guideline, are a goal for further development.

References

1. E. Domínguez, A. De Miguel, B. Pérez, A. Rodríguez, and M. A. Zapata. Sistemas de información hospitalaria en el contexto de la computación ubicua. In *Actas del II Congreso Iberoamericano sobre Computación Ubicua [CICU 06]*, pages 147–156, Alcalá de Henares, Madrid, Junio 2006.
2. E. Domínguez, B. Pérez, A. Rodríguez, and M. A. Zapata. Medical protocols for taking decisions, in an ubiquitous computation context. *Novática*, 177:38–41, Sept-Oct 2005.
3. S. Efroni, D. Harel, and I. R. Cohen. Toward rigorous comprehension of biological complexity: modeling, execution, and visualisation of thymic T-cell maturation. *Genome Research*, 13(11):2485–2497, 2003.
4. EMF Development team. The Eclipse Modeling Framework website, `http://www.eclipse.org/emf`.
5. R. B. France, S. Ghosh, T. Dinh-Trong, and A. Solberg. Model-driven development using uml 2.0: Promises and pitfalls. *IEEE Computer*, 39(2):59, 2006.
6. J. Lilius and I. Porres. vUML: A Tool for Verifying UML Models. In *The 14th IEEE International Conference on Automated Software Engineering*, Cocoa Beach, Florida, Oct 1999. IEEE Computer Society.
7. MOFScript Eclipse plug in. MOFScript website, `http://www.eclipse.org/gmt/mofscript`.
8. OMG. OMG Model Driven Architecture, June 2003. Document omg/2003-06-01. Available at `http://www.omg.org/`.
9. OMG. UML 2.0 Superstructure Specification, August 2005. Document formal/05-07-04. Available at `http://www.omg.org/`.