A Tool for Multi-scale Modeling of Software Architectures: Application to the Smart Home for Telemonitoring Elderly People at Home

Ilhem Khlif, Mohamed Hadj Kacem, Khalil Drira, and Ahmed Hadj Kacem

1 Introduction

The design of a software architecture is a complex task. On the one hand, we have to describe the system with enough details for understanding without ambiguity and implementing in conformance with architects requirements and users expectations. On the other hand, we have to master the complexity induced by the increasing model details both at the human and automated processing levels. So, there is a need for a new approach that automates the construction of the design architecture and guarantees its correctness. An iterative modeling process that helps architects to elaborate complex but yet tractable and appropriate architectural models and specifications can be implemented by successive refinements. Providing Rules for formalizing and conducting such a process is our objective, which we implemented in visual modeling notations. For this purpose, we propose to consider different architecture descriptions with different levels of modeling details called **"the scales"**. We define a step-wise iterative process starting from a coarse-grained description and leading to a fine-grained description. We propose a modeling solution to describe software architectures using a visual notation based on the UML graphic language [1]. UML is a standard modeling language defined by the OMG. These diagrams are submitted to vertical and horizontal transformations. The intermediate models provide a description with a given abstraction that

I. Khlif · M. H. Kacem (✉) · A. H. Kacem
University of Sfax, ReDCAD Research Laboratory, Sfax, Tunisia
e-mail: ilhem.khlif@redcad.org; mohamed.hadjkacem@isimsf.usf.tn; ahmed.hadjkacem@fsegs.rnu.tn

K. Drira
LAAS-CNRS, Université de Toulouse, Toulouse, France
e-mail: khalil.drira@laas.fr

L. Chaari (ed.), *Digital Health in Focus of Predictive, Preventive and Personalised Medicine*, Advances in Predictive, Preventive and Personalised Medicine 12,
https://doi.org/10.1007/978-3-030-49815-3_18

allow the validation to be conducted significantly while remaining tractable w.r.t. complexity. The validation scope can involve intrinsic properties ensuring the model correctness w.r.t. the UML description. To ensure model consistency, our approach supports model transformation and validation of UML models with OCL constraints. In order to experiment our approach, we tested it with a predictive and preventive system dedicated to the smart home application for maintaining personalized medicine at home. This system is helpful for people with loss of autonomy, exposed to risks of accidents or needing a precise daily medical follow-up. The remainder of the paper is organized as follows. In Sect. 2, we describe the multi-scale approach in Sect. 2. Section 3 presents the e-health application dedicated to the smart home for the homecare of elderly people. We conclude and outline some perspectives in Sect. 4.

2 Approach in a Nutshell

We propose a multi-scale modeling approach for software architectures [2, 5]. The proposed design approach is founded on UML notations and uses component diagrams. The diagrams are submitted to vertical and horizontal transformations for refinement; this is done to reach a fine-grain description that contains necessary details. The model transformation ensures the correctness of UML description, and the correctness of the modeled system. UML provides a formal language, the Object Constraint Language (OCL), to define constraints on model elements. Our approach supports model transformation and validation of UML models with OCL constraints [4].

The approach supports the modeling of multi-scale architectures using the semi-formal language UML. We propose a modeling solution to describe software architectures using a visual notation based on the UML graphic language. UML is a standard modeling language defined by the OMG [6]. The UML diagrams make it possible to present the structural properties as well as the behavioral properties of the multi-scale architecture. These diagrams are submitted to vertical and horizontal transformations. The intermediate models provide a description with a given abstraction that allow the validation to be conducted significantly while remaining tractable w.r.t. complexity. The validation scope can involve intrinsic properties ensuring the model correctness w.r.t. the UML description. To achieve this, we propose a set of model transformation rules. The rules manage the refinement and abstraction process (vertical and horizontal) as a model transformation from a coarse-grain description to a fine-grain description. To ensure model consistency, our approach supports model transformation and validation of UML models with OCL constraints. The choice of using OCL is motivated by its wide adoption in the MDE approach and the fact that it is a standard formal language supported by OMG. The second phase ensures validation of the model with the OCL language. Consequently, we have defined constraints on the elements of the model using OCL language. Tools are used to check and validate OCL constraints such as the

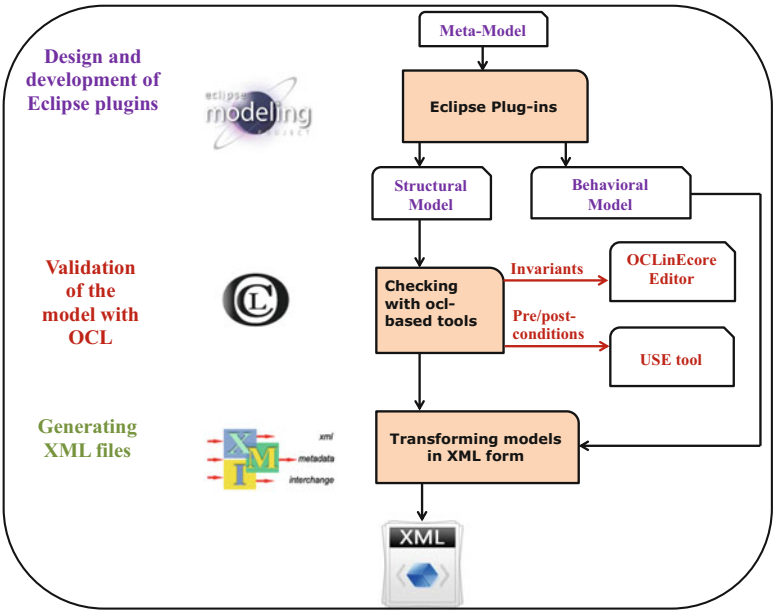

Fig. 1 Description of used tools for multi-scale modeling

OCLinEcore Editor to validate the invariants and the USE tool to validate the pre/post conditions. The third phase allows XML files to be generated using XMI, which transforms the models into XML. So we can get the XML file of the templates that we created. Figure 1 shows the different used tools during the design process.

2.1 Multi-scale Modeling

We present the multi-scale approach by a two-dimensional array describing vertical and horizontal scales [3]. The top-down scale transformation process, much like regular refinement, begins with a high level description of a system which we describe as a whole. Then, scale changes are applied to obtain a more detailed description, by describing components and connections. An iterative modeling allows to refine software systems descriptions. The vertical scales add the architecture decomposition details to obtain a more detail on the internal architecture of previously defined components. The horizontal scales describe or give details on the interconnections between components and their interfaces. We iterate on the architecture until reaching the details necessary to verify the associated architectural properties. The first scale Sv_0 begins with specifying the application requirements. It defines the whole application by its name. Two horizontal refinements called horizontal scales are associated with the first scale Sv_1. The first horizontal scale

shows all components that compose the application. The second one describes the links between those components. Four horizontal refinements are associated with the second scale Sv_2. The first scale presents subcomponents for components, and enumerates all the roles that each component can take. The second one identifies the list of communication ports for each component, and refines those roles. The third one shows the list of interfaces for communication ports. The last one is obtained by successive refinements while adding the list of connections established between components and subcomponents. This scale allows us to define the architectural style. With a graphical modeling language like UML, we are not able to express all the information required to convey the exact information of the domain in the diagram. We propose to use OCL to define constraints on models to define the well-formedness rules of our iterative design process.

3 Application to the Smart Home System

This section focuses on modeling the smart home system for the homecare monitoring of elderly and disabled persons. The smart home constituent elements are largely distributed in the house area. The main issue is to ensure efficient management of the optimized comfort, and the safety of the elderly and disabled person at home. We specify the essential information architecture and we illustrate, in (Fig. 2), the constituent elements of the smart home System. The monitoring center is

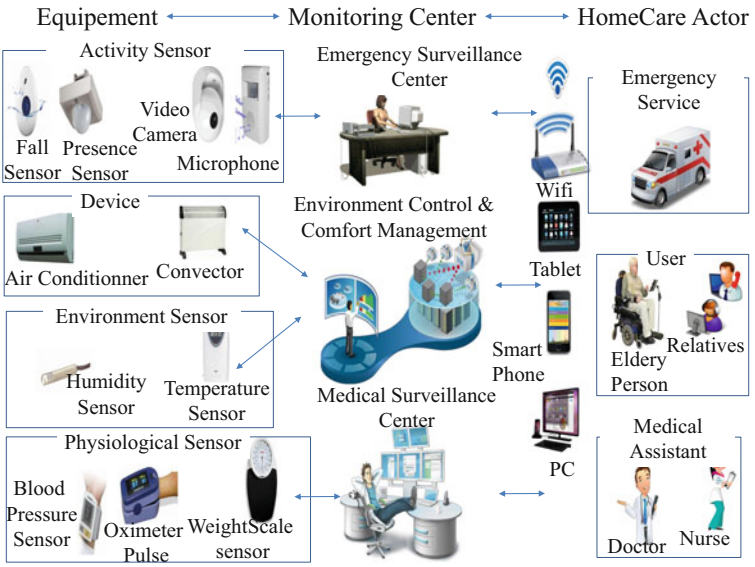

Fig. 2 Smart home system

composed of three systems:the Environment Control and Comfort Management, the Emergency Surveillance Center, and the Medical Surveillance Center. Thus, the actors selected who interact with other entities of the system are: The Home Care Actor, who interacts with the monitoring center, by setting medical or emergency conditions. The Equipment, that includes sensors and house devices. The emergency surveillance center controls critical situations using the activity sensors. In order to track the presence, activity sensors can be installed in each room. Activity sensors include fall sensors, presence sensors, video camera and microphone. The sensors send urgent signals to the center which treats immediately the received information. Once the signal is correct and the situation is critical, the center call the Emergency Medical Service to react and help the person. The medical surveillance center monitors physiological sensors. To track the medical information, physiological sensors can be installed in the bed, the chair, and on the body to detect the O2 level, the blood pressure, and the weight. The functions are achieved by the Oximeter, the Pressure Sensor, and the Weight Scale Sensor, classified as physiological sensors. While there are problems, the center requires the medical assistant intervention (the doctor, the nurse). The comfort management and the environment control system guarantees a comfort life for the users which are the elderly person and his relatives. This center enables communications between users, control the environment sensors (Humidity and Temperature Sensors), and commands the house devices (Convectors, Air conditioners). We are interested in studying the smart home system established for the home monitoring of elderly and disabled persons at home.

3.1 Structural Features

We experiment our approach by applying successive iterations to the smart home SoS. We illustrate the implemented iterative process applied to the smart home system. We obtained then the following results: In $S_{0.0}$, we define the application named "*SmartHome*". The constituent systems of the smart home are described (in $S_{1.1}$): *HomeCare-Actor*, *Equipment*, and *MonitoringCenter*. Those systems communicate with each other via the monitoring center. Those participants communicate with each other via the monitoring center. Those relationships are represented (in $S_{1.2}$) as links. UML associations. We also illustrate instances obtained in the next scale. We apply successive model transformation operations to add the following composites: *MedicalAssistant, EmergencyService, User, Physiological-Sensor, Activity-Sensor, Environment-Sensor, House-Device, MedicalSurveillance-Center, EmergencySurveillanceSystem, and EnvironementControlAndComfortManagement*. The *MonitoringCenter* plays the role of an "*EventDispatcher*". The *HomeCare-Actor* and *Equipement* play roles of "*Producer-Consumer*" in the application. $S_{2.1}$ allows to add the list of the ports for each component. We briefly describe the list of required/provided services of the *HomeCare-Actor* component. The *MedicalAssistant* receives information about the patient's situa-

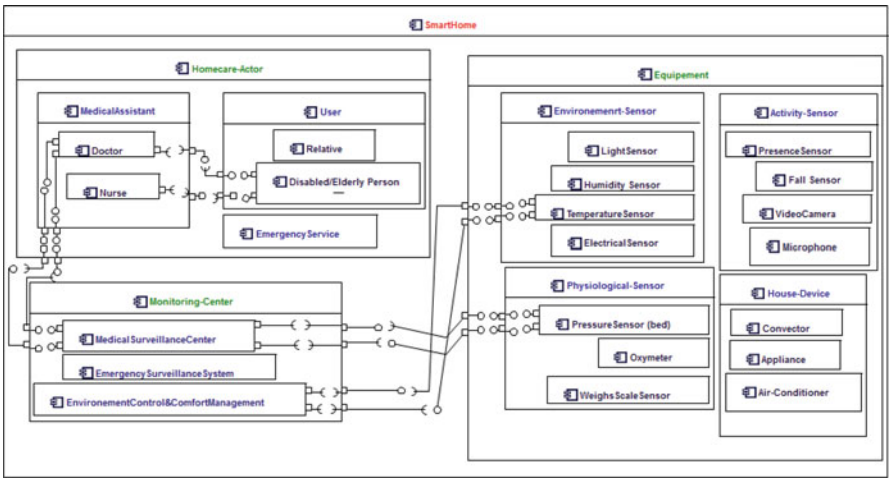

Fig. 3 Structural modeling applied to the smart home (Scale Sv_3/Sh_3)

tion from the *MedicalSurveillanceCenter*, he manages the patient's medical care (provides) and return a report after the care. The *EmergencyService* receives information about a critical situation *EmergencySurveillanceCenter*, reacts to save the patient (provides), and return a report after the intervention. The *User* receives not only emergency and medical services but also comfort services like online communication or house device command provided by the *EnvironementControl And ComfortManagement* component. $S_{2.2}$ assigns to each port an interface of the type provided or required according to the type of service. Finally, we indicate at the scale $S_{2.3}$ connections established according to the used topology and we define the "*Publish-Subscribe*" style. We implemented Eclipse plugins that allow multi-scale modeling of a software architecture based on the component diagram and the sequence diagram. We have checked and validated the OCL constraints under Eclipse (the structural part) and with the USE tool (the dynamic part). Finally, we have presented the level of instances of our case study using the Eclipse plugins (Fig. 3).

4 Conclusion

In this paper, we applied the multi-scale modeling approach on a case study for a predictive and preventive system dedicated to the smart home application for maintaining personalised medicine at home. This system is helpful for people with loss of autonomy, exposed to risks of accidents or needing a precise daily medical follow-up. We have experimented and evaluated the functional and the performance aspect of our approach as well as the functional aspect of our developed tool

supporting the multi-scale modeling approach for software architectures. In our future work, we expect to apply the multiscale modeling approach to other complex use cases for e-health applications for public health.

References

1. Khlif, I., Hadj Kacem, M., Hadj Kacem, A.: Iterative multi-scale modeling of software-intensive systems of systems architectures. In: Proceedings of the Symposium on Applied Computing, SAC 2017, Marrakech, 3–7 Apr 2017, pp. 1781–1786 (2017)
2. Khlif, I., Hadj Kacem, M., Hadj Kacem, A., Drira, K.: A multi-scale modelling perspective for SoS architectures. In: Proceedings of the 2014 European Conference on Software Architecture Workshops, ECSAW 14. Association for Computing Machinery, New York (2014)
3. Khlif, I., Hadj Kacem, M., Hadj Kacem, A., Drira, K.: A UML-based approach for multi-scale software architectures. In: 17th International Conference on Enterprise Information Systems (ICEIS), pp. 374–381 (2015)
4. Khlif, I., Hadj Kacem, M., Stolf, P., Hadj Kacem, A.: Software architectures: multi-scale refinement. In: 14th IEEE International Conference on Software Engineering Research, Management and Applications, SERA 2016, Towson, 8–10 June 2016, pp. 265–272 (2016)
5. Khlif, I., Tounsi, I., Hadj Kacem, M., Eichler, C., Hadj Kacem, A.: A refinement-based approach for specifying multi-scale software architectures: application to sos. In: Proceedings of the 33rd Annual ACM Symposium on Applied Computing, SAC '18, pp. 1660–1667. ACM, New York (2018)
6. UML, O.M.G.: Unified modelling language: Infrastructure (2011)