A Model-Driven Engineering Approach for the Well-Being of Ageing People

Koshima, Amanuel; Englebert, Vincent; Amrani, Moussa; Debieche, Abdelmounaim; Wakjira , Amanuel

Published in:
Advances in Conceptual Modeling: ER 2016 Workshops, AHA, MoBiD, MORE-BI, MReBA, QMMQ, SCME, and WM2SP, Gifu, Japan, November 14–17, 2016, Proceedings

DOI:
10.1007/978-3-319-47717-6_3

Publication date:
2016

Citation for pulished version (HARVARD):
Koshima, A, Englebert, V, Amrani, M, Debieche, A & Wakjira , A 2016, A Model-Driven Engineering Approach for the Well-Being of Ageing People. in *Advances in Conceptual Modeling: ER 2016 Workshops, AHA, MoBiD, MORE-BI, MReBA, QMMQ, SCME, and WM2SP, Gifu, Japan, November 14–17, 2016, Proceedings.* Springer, pp. 21-29. https://doi.org/10.1007/978-3-319-47717-6_3

A Model-Driven Engineering Approach for the Well-Being of Ageing People

Amanuel Alemayehu Koshima, Vincent Englebert, Moussa Amani,
Abdelmounaim Debieche, and Amanuel Wakjira

University of Namur, PReCISE Research Center, Belgium
`{Amanuel.Koshima,Vincent.Englebert,Moussa.Amrani,Abdelmounaim.Debieche}@unamur.be`
Vrije Universiteit Brussel
ETRO, Belgium
`{acwakjir}@etro.vub.ac.be`

Abstract. Ambient Assisted Living has been widely perceived as a viable solution to mitigate the astronomical increase in the cost of health care. In the context of our Geras Project, we propose a Model-Driven Engineering framework for handling high-level specifications that capture the concerns of elderly people still living at home. These concerns are related to concrete living issues, like being notified of a ringing phone for a deaf people, or receiving adequate assistance after a fall. The framework explicitly models three aspects: agent's goals, formally capturing users' concerns; abstract solutions, defining a canvas for answering the goal; and concrete solutions in terms of APIs or various combination of APIs, for their operationalisation. We illustrate the usage of our framework on two simple scenarios.

Keywords: Ambient Assisting Ageing People; Smart Home; Model-Driven Engineering; Goal Elicitation; Software Factory

1 Introduction

According to the World Health Organization report on ageing and health [12], the life expectancy of people has improved and most people can expect to live into their 60s and beyond. The increase in life expectancy augmented with low fertility rates result in an uneven demographic composition in developed countries [9, 15]. For instance, the ages of 30% of the European population are expected to be 65 years or above in 2050 [8]. Ageing people are more susceptible to frailty, chronic diseases, and increased multimorbidity, higher hospitalization rates and prevalence of health risk [9]. The demographic change and inadequate number of health care professionals will make the future costs of an ageing society unsustainable in terms of health care and social services [15]. Ambient Assisted Living (AAL) has been widely perceived as a viable solution to mitigate the astronomical increase in the cost of health care. Besides, it provides a new model of positive ageing that empowers older adults to maintain independence, functionality, well-being and higher quality of life in their residence [9].

AAL is a residential setting equipped with embedded technologies (sensors, actuators, cameras, and similar electronic devices) so as to support elderly people in their daily life by monitoring themselves and their environments [15]. This potential of smart home attracts industries and academia to provide different smart home based solutions for elderly citizens. However, the interoperability of different smart home solutions is far-fetched due to the fact that they address a limited set of requirements and problems, and they are usually developed in isolation and target different architectures and operating systems [13]. Model Driven Engineering (MDE) approach is proven to address interoperability problems among heterogeneous software systems [6].

In the context of our project Geras, we present in this paper a theoretical framework for handling high-level specifications aimed at capturing the concerns of elderly people living at home, and assisting them by deploying concrete solutions tailored to their home equipment, taking into account possibly conflicting concerns of residents.

The rest of the paper is organized as follows: Section 2 presents the Geras Framework. In Section 3, we demonstrate the running example to motivate our work. Section 4 discuses related work, and Section 5 presents conclusion and future work.

2 Geras Framework

The Geras framework aims to be a software factory to assemble and generate concrete smart home solutions from high-level abstract specifications. Geras adopts a Model Driven Engineering (MDE) approach.

MDE is a software engineering methodology that is adopted to deal with an ever increasing complexity of software solutions. MDE raises the level of abstractions of software development from technological details (i.e., source codes and underlying platforms) to the problem domain. In MDE, models are the principal artifacts that give full descriptions of software systems and are used for analysis, simulation, and source code generation of a software system [11]. Domain concepts are defined using well-suited Domain Specific Modeling Languages (DSML) at acceptable levels of abstraction [11].

MDE applies separation of concern principles that reduces complexity, improves reusability, and ensures simpler evolution of modeling languages [11]. It separates the domain knowledge from the underlying implementation details, as a result, the business domain and the underlying smart home technologies could evolve separately. These benefits of the MDE approach motivates the adoption of MDE in the Geras framework.

The Geras framework uses DSML tool-chains to generate IoT-based smart home solutions. Fig. 1 demonstrates the general architecture of the Geras framework. Although our concepts could be spread on several modeling levels (i.e. Agent and AgentType in Fig. 2), we decided to present them at the same level

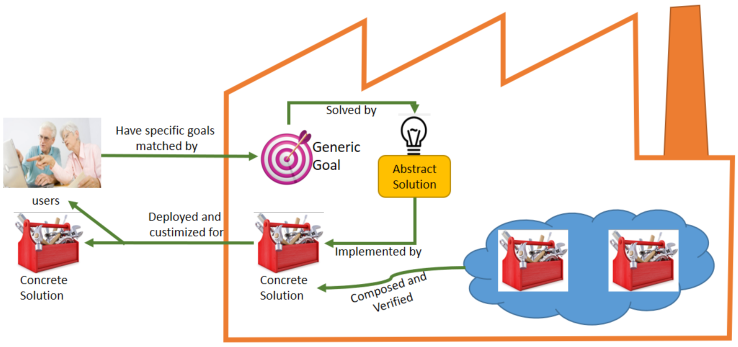

Fig. 1. General architecture of the Geras framework

(M2) for sake of simplicity. A generic goal represents a high level user concern (e.g. listening phone calls or having a quiet environment). Each goal may be associated with related abstract solutions. An abstract solution (AS) is akin to the concept of a patent, and does not provide much details about its implementation. The user profiles (e.g. environment, constraints, etc.) are used to tune the software generation process at design time. Besides, contextual information can also be used at runtime to equip the framework with context-aware capabilities.

In Geras, the mapping between a goal and an AS is performed in two steps. The AS is firstly described using a feature model that captures the variability of different product configurations (the technical solutions) [1]. Afterwards, the goal is mapped to the feature model. One goal can be achieved by one or more features of the solution. Apel *et al.* define a feature as “a characteristic or end-user-visible behavior of a software system” [1]. Of course, users do not usually live in isolation, therefore, the Geras framework should support social goal modeling as well.

One of the main challenge in social goal modeling is to detect conflicting goals automatically. In order to detect conflicts, we merge the feature selection of each user into one consolidated model. Afterwards, we use the feature configurator to validate whether this is consistent or not, for instance, two goals mapped to features that belong to a same parent and have an *alternative (XOR)* relationship denote a conflict. This case can be illustrated with an example: a husband who is deaf wants to listen a phone call, and this goal is mapped to a `high-volume` feature in the feature model. On the other hand, his wife who has a depressive character and wants to have a `quiet environment`. The `quiet environment` goal is mapped to a `low-volume` feature. In the merged model, both the `high-volume` and the `low-volume` features are selected, although they have an XOR relationship in the feature model. Hence, the configurator automatically detects this conflict.

The software factory is guided by a DSML that allows us to model both the “problem” and the “technical” spaces. Fig. 2 depicts the problem space meta-model. Meta-models are defined with an Extented Entity Relationship notation

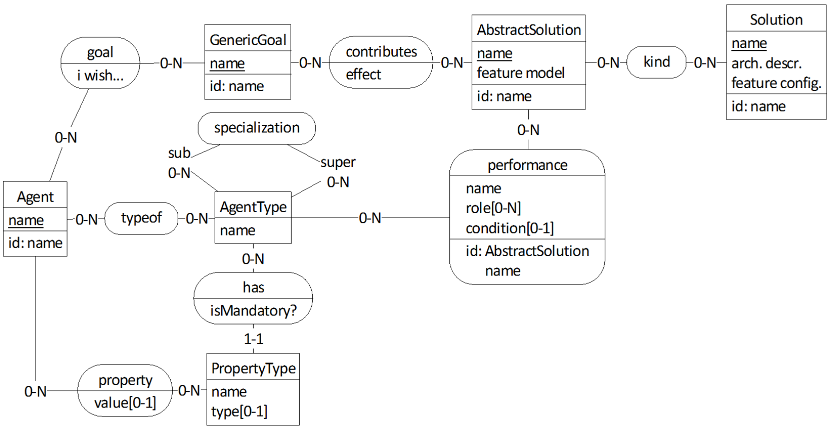

Fig. 2. Metamodel of the Problem Space

[7]. This meta-model specifies the knowledge about which kinds of solutions, (*AbstractSolution* AS), are available to address certain kinds of goals (*GenericGoal* GG). An AS represents an idea about a possible solution that solves problems encountered by users, where each user (*Agent*) may play a role (*Performance*) depending on his nature (*AgentType*). An agent is described with a set of *AgentTypes* and valued *properties* related to his/her types. An AS may contribute to a *GenericGoal* either positively or negatively (e.g. a solution that increases the volume of a TV may alter the goal `quiet environment`). The condition of a performance denotes a statement on the agent, e.g. `not deaf` where `deaf` would be a property.

The technical space meta-model (see Fig. 3) describes the available solutions (considered here as a shelf of reusable technological assets — COTS). A *Solution* is considered as a prototypical assembly of services (*Service*), software components (*Soft Component*), or hardware (*Equipment*). The *role* of each one is defined as well as its possible occurrences (*min-max*) by the *usage* relationship. The *equipment* can be arbitrarily complex and have IO devices. IO Devices, software components and services can have provided/required API. API can be defined by an implementation or be defined in an ontological way, independently from any provider's constraint. If an API subsumes another one, then a *wrapper* can be defined with a set of *matching rules.* A solution can denote a very simple assembly (a plug and its wireless switch) as well as more complex solutions based on sensors, actuators, hubs, computers, software and cloud applications. A *solution* can be defined as a composite of other solutions, where the assembling mechanism is defined with a set of *composition rules.* A solution can be deployed and customized for a set of agents identified by their role. This information can be used to ensure that the safety preconditions are met.

These both meta-models are two facets of a global meta-model where the *solution* concept is the hinge. This meta-model does not intend to be exhaustive but

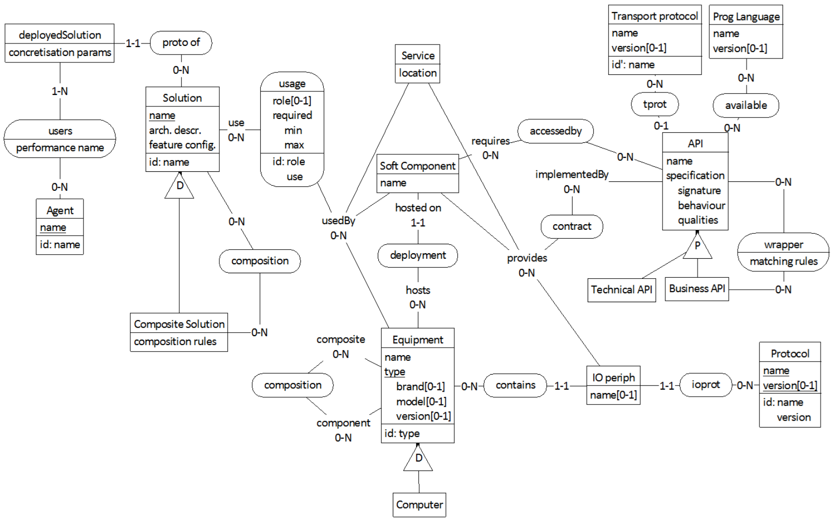

Fig. 3. Metamodel of the Technical Space

rather to be a transversal backbone among several concerns and offers alignment to other modeling languages such as goal modeling languages (GRL `http://www.cs.toronto.edu/km/GRL`, I* `http://www.cs.toronto.edu/km/istar`, KAOS urlhttp://www.objectiver.com/index.php?id=25), architecture description languages, other DSL for IoT for instance (ThingML[1]), or other languages (SySML[2]). The DSML has been defined with three specific goals in mind: designing user-centred AAL solutions by tuning or assembling existing solutions (i) at the lowest cost (ii) and with a good ROI (iii) in order to have a viable business model.

3 Running Example

We demonstrate the Geras framework with the following use case: *Albert, 70, and his wife Beth, 68, are a fictitious couple that lives in the country side of Belgium. Both are capable of using a smart home solution. Albert is deaf and can only hear sounds in a limited spectrum. Hence, he needs a smart home solution that assists him to be notified when his phone rings. Beth has muscular problems and needs to be assisted whenever she falls.* This use case is captured in Fig. 4: it proposes a concrete instance of the previous meta-models (Figs. 2 and 3). *Albert* is an instance of the *Agent* with *Deaf Person* as *AgentType*: a deaf person is characterized by the fact that they only hear a portion of the sound spectrum. Agents' properties detail their characteristics according to the property types declared for their agent type. For example, Albert's property `able` is set to `true`, indicating his ability to assist other persons (moving and calling) in their

---

[1] http://thingml.org/

[2] http://sysml.org/

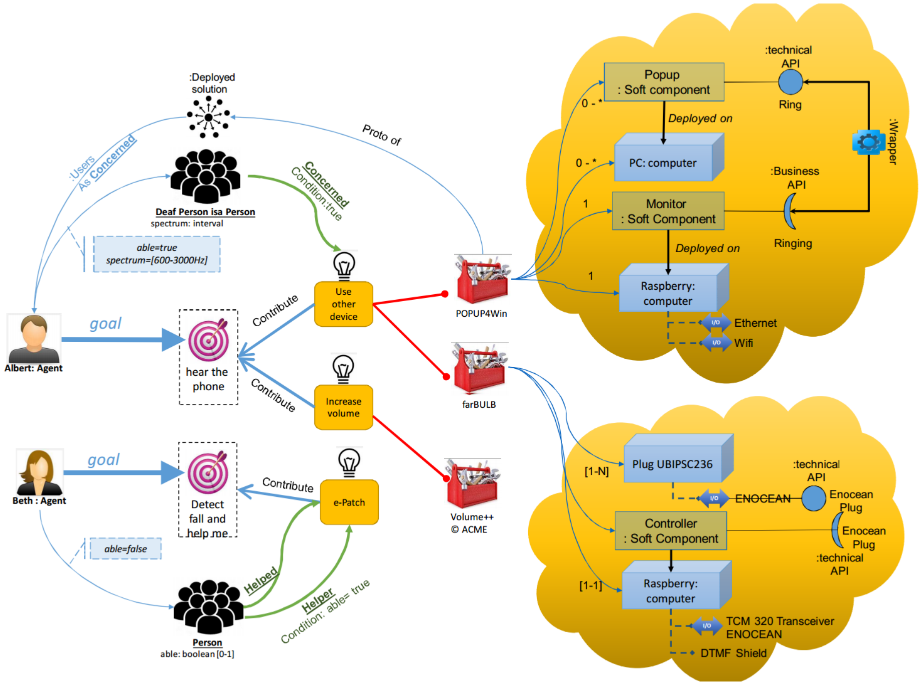

Fig. 4. Use Case

daily life; and property `spectrum` is set to `600-3000Hz`, indicating, for a deaf person, which sound spectrum they can hear.

One of the goal of a deaf person, as an agent type, is to be notified of phone calls when they occur: `increase volume` could be a potential abstract solution to achieve this goal, which can be operationalized into the concrete solutions shown in Fig. 4 (red icons). In the `POPUP4Win` solution, `Popup` and `Monitor` are software components that are deployed resp. on PC and Raspberry Pi. The Popup software component provides a technical API that lets its environment notify Albert when the phone is ringing, whereas the Monitor software component relies on a business API to notify other components when a call is detected. When the required and provided API do not match, the framework may deliver a wrapper to adapt the APIs. Other elements' explanation is omitted here for the sake of brevity, but they follow the same principles. When a solution is selected for deployment (`Deployed solution`), it can be customized to meet the users' constraints. The `concretisation parameters` of this deployment can then be used by the factory to automatise the process.

4 Related Work

The main challenge of IoT-based smart home solutions is an efficient management of interconnected resource-constrained (i.e., low power, small size) devices. The IoT reference architecture [2] eases the development of interoperable IoT applications by providing a common structure and guidelines for dealing with core aspects of developing, using and analyzing IoT systems. In [14], Pramudianto *et al.* develop the IoTLink toolkit by adopting an MDE approach and the IoT reference architecture. The MDE approach is used to ensure portability, interoperability and reusability of applications through separation of concerns [3]. The IoTLink is not a context-aware system. The context awareness capability is crucial for a smart home solution in order to leverage information about the end users and improves the quality of their interaction [16]. In [5], Fleurey *et al.* use a model-driven approach to develop adaptive firmwares. However, their work does not fully support the contextual information presented in [4].

The aforementioned work focus on using the MDE approach to ensure interoperability, but they provide little support for modeling the underlying motivation of the user in terms of stakeholder concerns and the high-level goals that address these concerns. Massacci *et al.* present a goal-oriented access control model for ambient assisted living [10], but their work has a limited scope and it does not use goal modeling techniques to generate and refine smart home solutions. In [15], Rafferty *et al.* adopt the goal oriented modeling approach to develop a smart home. However, the authors do not apply the MDE approach in their work that could ensure interoperability among heterogeneous smart home applications.

5 Conclusion and Future Work

AAL appears to become a widely adopted solution to mitigate the increasing costs of health care for elderly people. In this context, we propose a new framework that aims at (i) capturing high-level concerns through abstract goals people living in their smart home encounter in their daily life; (ii) associating these concerns to abstract solutions that describe a way to overcome these concerns; and (iii) operationalizing these solutions according to the equipment the smart home possess. We proposed a conceptual meta-model capturing all aspects of our framework, and demonstrated its use through two illustrating examples coming from real-life situations.

We are currently focusing on two aspects. The first one targets interoperability between the many devices a smart home can contain. In order for our concrete solutions to work, independently of the APIs, protocols, manufacturers or concrete capabilities the devices in the smart home present, we plan to design a DSML targeting the centralized inter-communication of devices, based on business rules that implement the requirements expressed by abstract solutions.

At a earlier stage in our framework, the second aspect targets the capture of high-level concerns through the elicitation of the residents' concerns expressed as goals. This crucial step requires an explicit modeling of the user profiles, the many contexts of an abstract solution, spanning from the devices' network to a lightweight description of the home's environment. In turn, this explicit representation should allow to analyze adequately the compliance of the proposed abstract solutions, and the potential conflicts that will inevitably occur throughout the use of our framework.

References

1. Apel, S., Batory, D., Kästner, C., Saake, G.: Feature-oriented software product lines: concepts and implementation. Springer Science & Business Media (2013)
2. Bauer, M., Bui, N., De Loof, J., Magerkurth, C., Nettsträter, A., Stefa, J., Walewski, J.W.: IoT reference model. In: Enabling Things to Talk. Springer (2013)
3. Belaunde, M., Casanave, C., DSouza, D., Duddy, K., El Kaim, W., Kennedy, A., Frank, W., Frankel, D., Hauch, R., Hendryx, S., et al.: Mda guide version 1.0 (2003)
4. Dey, A.K., Abowd, G.D., Salber, D.: A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Human-computer interaction (2001)
5. Fleurey, F., Morin, B., Solberg, A.: A model-driven approach to develop adaptive firmwares. In: Proce. of the 6th SEAMS (2011)
6. Fleurey, F., Morin, B., Solberg, A., Barais, O.: MDE to manage communications with and between resource-constrained systems. In: Model Driven Engineering Languages and Systems. Springer (2011)
7. Hainaut, J., Hick, J., Englebert, V., Henrard, J., Roland, D.: Understanding implementations of IS-A relations. In: ER. Lecture Notes in Computer Science, Springer (1996)
8. Herradi, N., Hamdi, F., Métais, E., Ghorbel, F., Soukane, A.: Personlink: An ontology representing family relationships for the captain memo memory prosthesis. In: Advances in Conceptual Modeling. Springer (2015)
9. Lê, Q., Nguyen, H.B., Barnett, T.: Smart homes for older people: Positive aging in a digital world. Future internet (2012)
10. Massacci, F., Nguyen, V.H.: Goal-oriented access control model for ambient assisted living. In: Privacy and Identity Management for Life. Springer (2009)
11. Meyers, B., Vangheluwe, H.: A framework for evolution of modelling languages. Sci. Comput. Program. (2011)
12. Organization, W.H., et al.: World report on ageing and health. World Health Organization (2015)
13. Perumal, T., Ramli, A.R., Leong, C.Y., Mansor, S., Samsudin, K.: Interoperability for smart home environment using web services. In: Int. Journal of Smart Home (2008)
14. Pramudianto, F., Kamienski, C.A., Souto, E., Borelli, F., Gomes, L.L., Sadok, D., Jarke, M.: IoTLink: An internet of things prototyping toolkit. In: UTC-ATC-ScalCom. IEEE (2014)
15. Rafferty, J., Chen, L., Nugent, C.: Ontological goal modelling for proactive assistive living in smart environments. In: Ubiquitous Computing and Ambient Intelligence. Context-Awareness and Context-Driven Interaction. Springer (2013)
16. Robles, R.J., Kim, T.h.: Review: context aware tools for smart home development. Int. Journal of Smart Home (2010)