Behavior Modeling for Ambient Assistance

Judith Michael, Volodymyr Bolshutkin, Stefan Leitner, Heinrich C. Mayr
Department for Applied Informatics / Application Engineering
Alpen-Adria-Universität Klagenfurt
Klagenfurt, Austria
{judith.michael, stefan.leitner, heinrich.mayr}@aau.at, vboslhut@edu.uni-klu.ac.at

***Abstract* - This paper reports on research done within the project HBMS - Human Behavior Monitoring and Support - which aims at compensating cognitive deficiencies within the context of Ambient Assisted Living. Modeling of human everyday life behavior is crucially dependent on user-involvement, i.e. requests an appropriate easy-to-understand modeling language. The paper introduces such a language, exhibits the meta-model of that language and discusses the concept of a Human Cognitive Model from which knowledge units may be derived within a concrete support situation. As a proof of concept a tool prototype will be sketched.**

Keywords-component; Healthcare Information System; Elderly in-home assistance; Modeling of Behavior; Meta-Modeling; Model Integration

I. INTRODUCTION

The main goal of Ambient Assisted Living (AAL) projects is to enable elderly people to live as long as possible autonomously in their domestic environment. Whilst most IT-based AAL approaches aim at predefined support for typical situations we focus on supporting people by their own (former) knowledge [1]. This means that we have to learn that knowledge, and that in turn means that we have to model a person's behavior and to integrate models of behavior sequences to a (conceptual) Human Cognitive Model. In this paper we present the modeling approach chosen in our AAL-research project Human Behavior Monitoring and Support (HBMS). Its' aim is to provide means for supporting a person with knowledge about his/her previous preserved behavior,

The paper is structured as follows: Chapter 2 introduces our research project and outlines the main ideas; chapter 3 explains the modeling language and its underlying meta-model; chapter 4 shows, as a proof of concept, a prototypical implementation; chapter 5 sketches the mapping behavior model instances to a generalized model; a conclusion and a short outlook on subsequent research steps close the paper.

II. THE HBMS PROJECT

Highly sophisticated technologies ease our everyday live to the same degree as they cause new challenges: e.g., for new technical devices, we need to understand their possible applications and their use, as well as we have to learn their handling; or, software-upgrades are come with new interaction paradigms and additional or removed functionalities. Sometimes it is easier, sometimes it is harder to adopt such innovations; sometimes they even force us to behavioral changes. However, as studies show (e.g., [2]), the adoption to innovations generally becomes harder when advancing in years. In particular, this applies to not regularly used devices and functions.

Our vision is to assist people by their own previous knowledge about mastering everyday tasks, and by using their own words for providing support. The project HBMS (Human Behavior Monitoring and Support) is a first step to realize that vision. It focuses on modeling a person's course of action in daily routines in order to preserve her/his individual episodic memory and to hand it back to that person when needed. The particular routine models will be integrated (and stored) in a cognitive model called Human Cognitive Model (HCM).

At a first project stage we selected four types of scenarios, in which our approach could provide assistance and which therefore are appropriate for a proof of concept: (1) the usage of technical devices, (2) activities of everyday life, (3) business processes and (4) living in ambient environments. The main target groups are people who are independent of others in their Activities of Daily Living (ADLs). For example Basic ADLs like feeding or dressing and Instrumental ADLs like the ability to use a telephone, to do food preparation, housekeeping or shopping. [3]

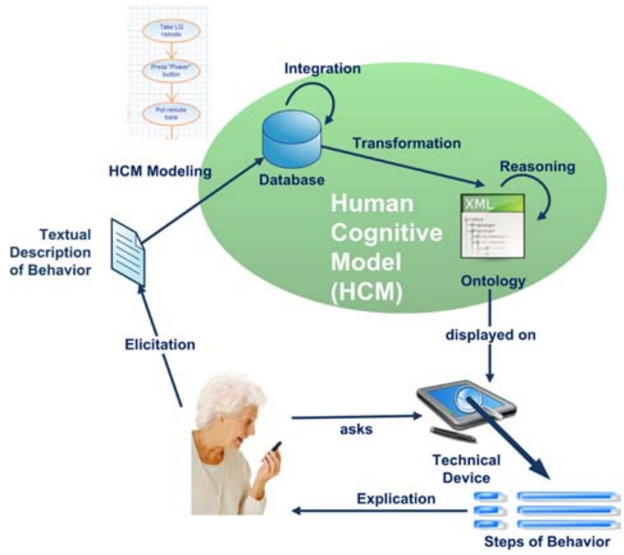

Figure 1. Overview of the HBMS process in the first phase of realization

In a first realization phase (see figure 1) a description of a persons' behavior is mapped to and integrated into a cognitive

Funded by the Klaus Tschira Stiftung, Heidelberg

model (HCM, Human Cognitive Model) step-by-step. This model then is transformed to a formal ontology which allows inferring, by reasoning, the knowledge that is needed for assistance in a given situation. A backend component explicates that knowledge in a form that is appropriate to the given situation.

Figure 2 sketches the process model of our approach. This paper focuses on the first two steps: Elicitation and Integration.

Figure 2. The five steps of the HBMS process

III. Modeling Human Knowledge

Knowledge Management Systems provide mechanisms for systematically managing knowledge. To get knowledge into such a system, knowledge modeling is needed [4] from elicitation, conceptual model development, model integration up to model transformation. Dimensions to be modeled are of dynamic, static, functional nature as well as views [5]. For modeling human factual knowledge, modeling of behavior is the most relevant part.

There exists a variety of modeling notations and modeling tools for this purpose. We have compared some existing graphical notations, such as UML Activity Diagrams, KCPM and BPMN in respect of their readability, verbosity and ability to express real-life situations.

In simple cases, such as modeling sequences with loops and choices, UML Activity Diagrams and BPMN are quite useful. However, the HBMS project goals require the modeling of complex concurrent processes of only one main actor. Paper [6] provides a comparison of BPMN and UML based on workflow patterns. Unfortunately, both of them cannot express patterns necessary for our models, such as Interleaved Parallel Routing (unordered sequence), without an overhead.

Clearly, user-centered development is a specific demand in AAL environments. We use, for that purpose, a language derived from a modeling language developed for user-centered software requirements engineering.

A. Klagenfurt Conceptual Predesign Model (KCPM)

The Klagenfurt Conceptual Predesign Model (KCPM) approach is designed for software requirements modeling and provides means for automatic transformation of KCPM to conceptual models. KCPM offers semantic concepts for modeling Universe of Discourse (UoD) static and dynamic aspects [7].

KCPM was also designed for easy to validate models as well as for construction models out of natural language texts by language natural processing techniques. The model is based on an ontology which understands UoDs as systems consisting of interrelated elements (things) that are able to perform services (operations) and that are activated by events (messages sent by other things).

KCPM modeling concepts for static aspects are thing type, connection type, perspective and constraint. The concept of *thing type* (see figure 3) is a generalization of conceptual notions such as entity type, class and attribute or value type. Thing types usually are referred to by noun phrases in textual requirements specifications. Typical things (instances of thing types) are natural or juridical persons, material or immaterial objects, abstract notions and descriptive characteristics (e.g. a customer name, a product number or description) [8].

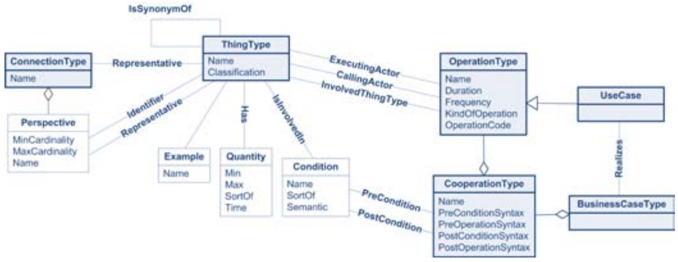

Figure 3. Overview of the meta-model of KCPM including the main concepts

Connection types represent relationships between two or more things. Perspectives are used to describe these different aspects of a connection type from the point of view of all of the involved thing types (e.g. Author writes a book / Book are written by authors) [9].

Operation types are used to model functional services that can be called via messages (service calls). This is similar to the notions use case, activity, action, method, service etc. An instance of an operation type (characterized by references to so-called thing types) can be called or executed by actors and may have parameters [10].

The basic building block of KCPM process models is the concept of *co-operation type*. A co-operation type consists of a set of concurrent operation types (at least one) which are executed under certain pre-conditions leading to post-conditions. Matching equivalent post- and preconditions of different co-operation types lead to a network which, in the simplest case represents a sequence of operation types.

Based on these dynamic modeling concepts KCPM models may be mapped to UML-diagrams (e.g. UML Activity Diagrams). The mapping is achieved by applying a set of heuristic rules.

B. Human Cognitive Model

In HBMS we are evolving KCPM to a Human Cognitive Model (HCM) for modeling human behavior in the ambient assistance environment. In particular we have to add explicit concepts for capturing temporal and spatial dependencies. On the other hand, the concept of concurrency turned out to be too powerful and consequently too complex given the fact that a person's actions mostly are performed sequentially. We therefore replaced the concept of co-operation type by the concepts of *gateway* (similar to BPMN) and *condition*. Semantically, gateways may be specified to 'and', 'or' and 'xor' respectively, thus allowing for some parallelism. This approach proved to be more intuitively to understand by the intended user groups.

The execution of operations is controlled by pre-conditions and induces post-conditions. Conditions are distinguished by their pragmatic property: A *push-condition is* set by some external actor and is immediately accessible to a human sensory perception (e.g., ringing of a door bell). A *poll-condition* has to be evaluated by a person before recognition.

The KCPM concepts for modeling static aspects remained unchanged.

To evaluate the power of the resulting HCM a modeling tool prototype has been developed.

IV. Prototype of the Modeling Tool

Following [11], a modeling tool has to feature a set of characteristics. Clearly, the underlying meta-model and notation concepts have to be realized appropriately and completely. Likewise, standard functionality like the creation, edition and display of models, import/export functions and model persistence has to be provided.

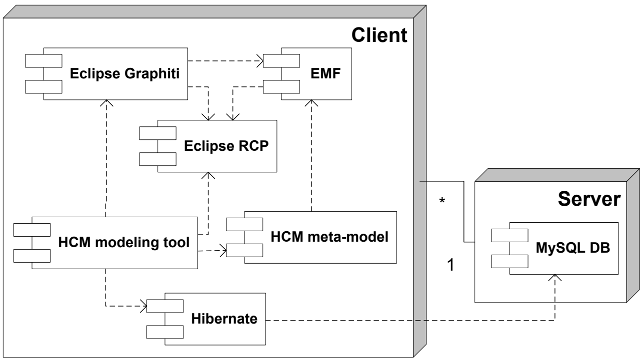

Figure 4. Client-server architecture of the HCM modeling prototype

Figure 4 shows the HCM modeling tool prototype architecture which follows the client-server paradigm. For the implementation we decided to use the Eclipse Rich Client Platform (RCP). RCP is based on Equinox, an implementation of the OSGi core framework specification, and has lightweight and powerful plug-in mechanism. The Eclipse community provides a lot of useful projects, including Eclipse Modeling Framework (EMF), which is especially useful for conceptual modeling. The HCM meta-model was created in EMF format.

EMF also supports “models to text” and “models to models” transformation [12]. These features may become necessary for model validation and analysis.

Moreover, Eclipse provides a modeling infrastructure evolving around EMF. There are several graphical editors out of which we choose Eclipse Graphiti as the most balanced framework. Graphiti enables rapid development of state-of-the-art diagram editors for both, EMF-based domain models as well as any Java-based objects. It provides its own API on the top of GEF and Draw2d and helps to avoid low-level operations. Graphiti also provides a set of very useful editing features, such as automatic layout and aligning, drag-and-drop and direct editing support. It also allows expanding sub-models, to create models on different levels of abstraction.

For persistence layer implementation we use a MySQL database and Hibernate, a popular Java ORM framework. Semantic model data and data related to graphical representation are stored separately. Presentation data is stored in a single database column in XML format.

A screenshot of the elaborated graphical modeler for the HCM is shown on figure 5. The left hand side lists existing models and model elements for a person in a tree-view. On the right hand side a components palette and a drawing area for the graphical representation is provided. It is possible to create models for static aspects as well as for dynamic ones, which are related to each other. The current prototype version supports copy-paste and drag-and-drop features to allow reusing already created model elements in new models.

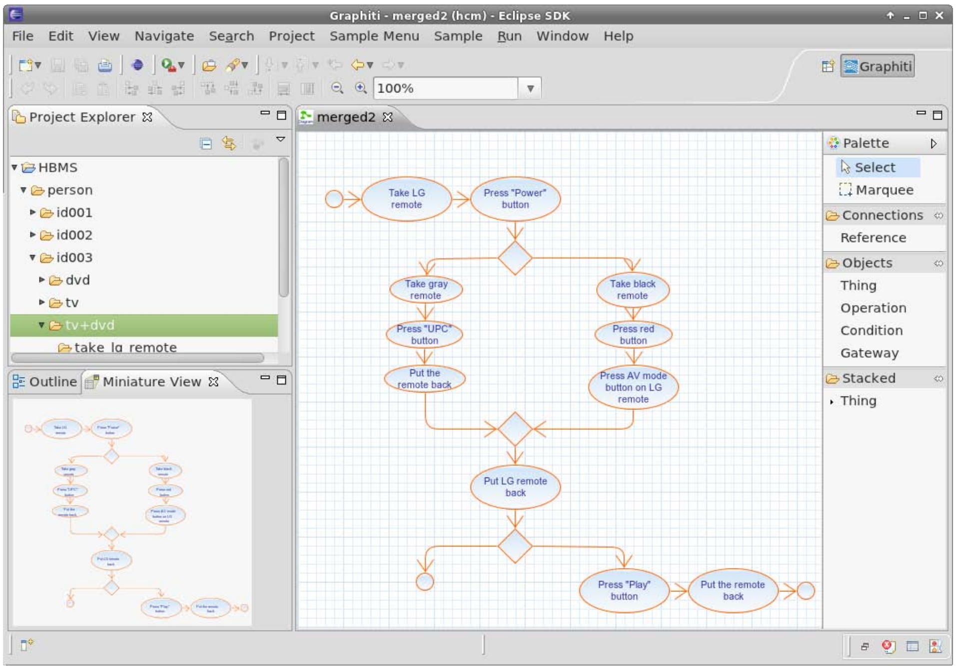

Figure 5. Screenshot of the HCM modeler prototype

V. INTEGRATION OF BEHAVIORAL SEQUENCES

The HCM Modeler supports modeling of real life behavioral sequences like turning on a TV, a satellite receiver or a DVD player, recording a film or switching the channel. Such processes are difficult to execute for humans and are getting the more difficult, the more technical devices are included. Each remote control has different turn on/switch off buttons, different locations for volume adjustment and in addition lot of buttons the semantics of which can only be figured out by reading the operating manual.

A particular action is modeled as a sequence of steps. Thus, monitoring and modeling actions and behavior over a period of time will result in a set of sequences stored in the database, each of them representing a particular correct behavioral sequence of a given person.

Figure 6 shows two sequence models: 1) turning on a TV and a satellite receiver, and 2) turning on a TV and a DVD player. The perspective of these sequences is always that of one individual person.

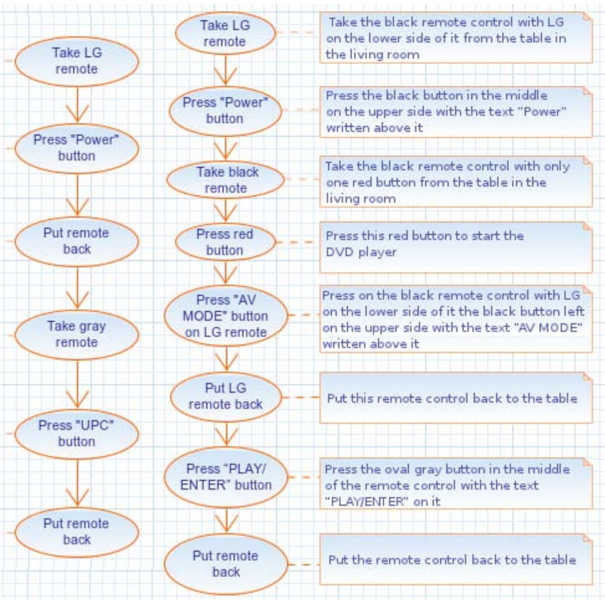

Figure 6. Screenshot of two sequences: “Turn on the TV and the satellite system” and “Turn on the DVD player”.

To integrate these sequences we apply the rules discussed in [13] in order to identify matches and mis-matches as well as conflicts and inconsistencies [14]. Figure 7 shows the result of such an integration process.

VI. FURTHER WORK

As the project proceeds, more and more tasks will be automated. Currently we assume that all modeled sequences are correct. In future we need to check if there are faults in a person’s way to perform an action – and to differentiate these from simple changes in some steps of that action. This might be realized with methods of Case-Based Reasoning (CBR) [15].

Additional work will have to be done re integrating of multimedia information into the meta-model. To cope with the complexity of different thing types it may be necessary to introduce special thing types (e.g. hyperlink).

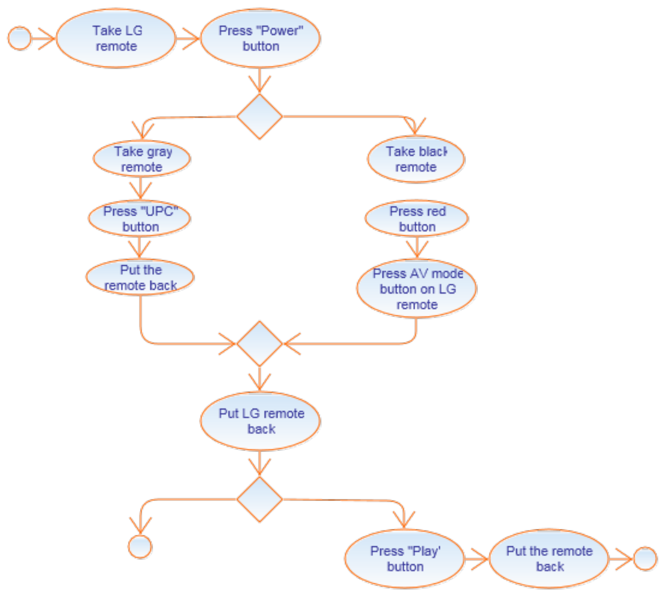

Figure 7. Integrated model based of the two sequences in Figure 6.

The elaborated modeling tool is now only a prototype and many improvements are possible. Creating nested sub-models is not supported now, but it should be implemented in future. Other important task is to implement model validation and transformation to text (e.g. documentation) or to other models. The integration will be implemented as a background process and the usability of the prototype will be improved as well as the graphical user-interface [16].

REFERENCES

[1] Griesser, A.; Michael, J. and Mayr, H.C.: Verhaltensmodellierung und automatisierte Unterstützung im AAL Projekt HBMS, 5. Deutscher AAL-Kongress, 2012.

[2] Reischies, M; Lindenberger, U.: Grenzen und Potentiale kognitiver Leistungsfähigkeit im Alter. Die Berliner Altersstudie. Akad.-Verl., Berlin, 2010.

[3] Bucks, R.S. et al.: Assessment of Activities of Daily Living in Dementia: Development of the Bristol Activities of Daily Living Scale, Age and Ageing. 1996, pp. 113-120.

[4] Wielinga, B., Sandberg, J. and Schreiber, G.: Methods and Techniques for Knowledge Management: What Has Knowledge Engineering to Offer? Expert System With Application. Vol. 13, No. 1, 1997, pp. 73-84.

[5] Batini, C.; Ceri, S. and Navathe, S.B.: Conceptual Database Design: An Entity Relationship Approach, Benjamin Cummings, 1991.

[6] Wohed, P. et al.: Pattern-based Analysis of BPMN – an extensive evaluation of the Controlflow, the Data and the Resource Perspectives. 2006.

[7] Mayr, H.C. and Kop, C.: A User Centered Approach to Requirements Modeling. In: M. Glinz et al. (Eds.): Modellierung 2002. Modellierung in der Praxis - Modellierung für die Praxis. Bonn: Köllen Verlag, Lecture Notes in Informatics (LNI) P-12, 2002, pp. 75-86.

[8] Kop, C. and Mayr, H.C.: Conceptual Predesign – Bridging the Gap between Requirements and Conceptual Design. In Proceedings of the 3rd International Conference on Requirements Engineering. Colorado Springs Colorado, April 6-10, 1998.

[9] Shekhovtsov, V.A.; Kop, C. and Mayr, H.C.: Capturing the Semantics of Quality Requirements into an Intermediate Predesign Model. SIGSAND-EUROPE, 2008, pp. 25-38.

[10] Fliedl, G.; Kop, C. and Mayr, H.C.: From textual scenarios to a conceptual schema, Data & Knowledge Engineering, v.55 n.1, 2005, pp. 20-37.

[11] Kaschek, R. and Mayr, H.C.: A characterization of OOA tools. Proceedings of the Fourth International Symposium on Assessment of Software Tools (SAST '96), 1996, pp. 59-67.

[12] Oldevik, J. et al.: Toward Standardised Model to Text Transformations. In: A. Hartman and D. Kreische (Eds.): Model Driven Architecture – Foundations and Applications. Springer Berlin / Heidelberg, 2005, pp. 239-253.

[13] Vöhringer, J. and Mayr, H.C.: Integration of schemas on the pre-design level using the KCPM-approach. In: A.G. Nilsson et al. (Hrsg.): Advances in Information Systems Development: Bridging the Gap between Academia & Industry. Heidelberg: Springer Verlag, 2006.

[14] Bellström, P.; Vöhringer, J.: Towards the Automation of Modeling Language Independent Schema Integration. Proceedings of the 2009 International Conference on Information, Process, and Knowledge Management, IEEE Computer Society, 2009, pp. 110-115.

[15] Aamodt, A. and Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, AI Communications, Vol. 7, 1994, pp. 39-59.

[16] Hellman, R.: Nutzerfreundliche Benutzeroberflächen für Menschen mit Gedächtnisproblemen, 5. Deutscher AAL-Kongress, 2012.