Enterprise Modelling and Information Systems Architectures
February 2018. DOI:10.18417/emisa.si.hcm.26

Special Issue on Conceptual Modelling in Honour of Heinrich C. Mayr

The HBMS Story

Past and Future of an Active Assistance Approach

Judith Michael[*,a], Claudia Steinberger[a], Vladimir A. Shekhovtsov[a], Fadi Al Machot[b], Suneth Ranasinghe[a], Gert Morak[a]

[a] Institute for Applied Informatics, Alpen-Adria-Universität Klagenfurt, Austria
[b] Leibniz Center for Medicine and Biosciences, Research Center Borstel, Germany

Abstract. *The aim of the Human Behavior Monitoring and Support (HBMS) project has been to actively assist individuals in activities of daily living and other situations using users' own episodic knowledge. This knowledge is represented and preserved in HBMS in the HCM, the Human Cognitive Model, expressed in the domain specific modelling language HCM-L. HCM also forms the base for reasoning, model matching and support state visualization. Moreover, in the HBMS-System conceptual models are also used to define interfaces to activity recognition systems, support clients and data available in the Semantic Web. Thus, we see HBMS-System as an application of the Model Centered Architecture (MCA) paradigm. This paper describes how the project evolved over time, its main challenges and milestones, its main processes and their dependencies, and what is going to happen in the next future.*

Keywords. Conceptual Modeling • Active Assistance • Human Behavior Monitoring and Support • Model Centered Architecture

1 Introduction

Motivation. This paper tells the story of a project, in which the authors have been working over the last few years. The motivation for this project can be explained best by retelling a story a researcher and later HBMS project manager told as his vision at the very project beginning:

Imagine your mother suffering from incipient dementia. As time goes by, she loses her episodic memory more and more. For example, she forgets how to use her video recorder to watch beloved music videos and begs you to explain it to her. You are going to show it to her repeatedly and she can do it for a while, but then she will forget it again. Forgetting happens faster and faster. After some more time, you realize she does not ask you anymore to explain it to her. She does not want to burden you. If you try to explain it to her again, she tells you to stop, because she knows she won't remember. She stops watching her beloved videos at all. And you know she also stops doing other things she likes, because she can't remember how to do. In this way her autonomy decreases very quickly. Wouldn't it be great if she had a permanent electronic assistant to help her to accomplish her tasks of daily living and to support her to live autonomously as long as possible?

The researcher with this vision was Heinrich C. Mayr and the idea was transformed into the Human Behavior Monitoring and Support (HBMS) project, funded by the Klaus Tschira Stiftung gGmbH in two project parts from March 2011 to January

* Corresponding author.
E-mail. judith.michael@aau.at
We thank the Klaus Tschira Stiftung GmbH for their project funding and all colleagues and students involved in the development of the HBMS project.
Note: Although Heinrich C. Mayr is not an author of this article, the ideas and work presented in this article were developed together with him. (Mayr et al. 2016)

2016 with a funding amount of approx. 1.000.000 Euro.

Research idea. The idea of HBMS was to develop a system which provides ambient assistance to support the autonomy of a person in case of a decreasing memory. This is to be achieved through the conceptualization of the person's episodic memory, the establishment of the model of this knowledge and the use of this model for support. Giving support means to help people to remember how they performed a certain activity by reactivating already existing memory anchors. A memory anchor is a simple stimulus that influences the state of mind. By providing retrieval cues, it is easier to remember experienced situations (cued recall) (see Strube 1996, p. 204 and p.196). Thus, the person is supported by the HBMS-System with his or her own behaviour knowledge by providing retrieval cues.

The aim. The HBMS project aims at deriving support services from integrated models of abilities, current context and episodic knowledge that an individual had or has, but has temporarily forgotten. The core of the HBMS-System is the Human Cognitive Model (HCM) which preserves the episodic memory (Tulving 1972) of a person as conceptual models of behaviour linked to context information related to these activities (see Figure 1). The acronym HCM was not coincidently chosen - it was named after the nickname of Heinrich C. Mayr (derived from his initials).

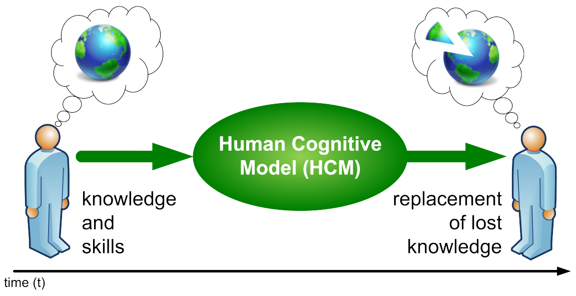


Figure 1: The main HBMS idea

How to meet the aim. The developed system should be able to monitor actions, identify activities, create conceptual models out of these activities, transform them into an ontology representation which forms the knowledge base, draw conclusions out of this knowledge base, and use this knowledge to provide support. Thus, the HBMS team has created an active assistance (AA) system, which realized these aims step-by-step.

Outline. First, an overview of the project is presented in section 2, sketching our key activities during the project, which will be examined more closely in the subsequent sections. Thus section 3 presents our approach for end user involvement and performed evaluations. The whole HBMS project was accompanied by a continuous analysis of the state-of the-art of all relevant areas, which is presented in section 4. In section 5 our Domain Specific Modelling Method including our modelling language HCM-L and its chronological development is treated. The HBMS-System development is presented in section 6: The HCM-L Modeller (Section 6.1), observation aspects such as the HBMS observation interface, an alternative to real human observations via a simulator component (Section 6.2), solutions to handle business intelligence like reasoning approaches and a real-time visualization for behaviour models (Section 6.3), as well as different user interfaces (Section 6.4) complete this section. The last section summarizes the projects' impact and discusses future ideas.

2 Past and Future of HBMS

This chapter aims to sketch what happened in HBMS previously and what else we have planned. As conceptual modellers, we present this of course in form of a conceptual model: the development of HBMS, the processes we were involved in, the deliverables and responsibilities. We have documented what has happened in HBMS until now and what is going to happen in the next future.

Figure 2 conceptualizes our top level research processes and structures the work which has been done so far. To keep Figure 2 as simple as possible we modified BPMN slightly for our purposes:

- Our HBMS process model is both prescriptive and descriptive. Thus we have added a timeline showing past, present and future activities.
- To represent the academic impact of HBMS clearly, we have added our deliverables published in the course of time to our process model as data outputs on the timeline.
- All activities in Figure 2 can be seen as sub-processes. The sub-processes are detailed in more depth in the following subsections of this paper.
- The most important interactions between sub-processes of different pools are sketched using message flows.

Figure 2 comprises the following pools:

Evaluation. It has been a core part of HBMS to analyse user requirements and to evaluate the practicability and the benefits of HBMS throughout the entire project. User centred requirement analysis workshops were mainly conducted with elderlies in small groups of up to 25 persons. Surveys were conducted among participants of all ages to obtain feedback on design and user interface approaches. Practicability evaluations were done with volunteers mainly in user experience labs, which were organized primarily as part of ‘long night of research’ (LNF) events, which attracted thousands of people to our University. Evaluation processes were influenced by other processes and influenced processes in other pools throughout the entire project, especially to a high degree during the first third of the project period.

Analysis. During our analysis process we investigated the current state of the art in the domains of active and assistive systems, modelling languages, context and process ontologies, activity recognition approaches, and observation technologies. This pool also hosts processes, which define requirement specifications as a basis for processes in other pools. Inputs came mainly from evaluation processes as mentioned above.

Domain Specific Modelling Method (DSMM) Design. HBMS aims to assist individuals in their activities of daily living and other situations using their own episodic knowledge as well as knowledge about their context including the resources users have at hand. This knowledge is preserved in HBMS as a Human Cognitive Model (HCM), which contains behavioural and contextual elements. We have worked on the domain specific modelling language, called Human Cognitive Modelling Language (HCM-L), and developed several incremental versions over time.

This work was mainly influenced by analysis and evaluation processes and affected essentially the development process of the modelling tool (HCM-L Modeller).

Today, the actual version of HCM-L comprises really powerful context modelling elements for user profile modelling, resource modelling and environment modelling.

The development of the HBMS-System included extensive research activities to create the HCM-L Modeller, to develop appropriate user interfaces, to handle user observations and to sharpen the HBMS-Systems’ intelligence. The architecture of the HBMS-System corresponds now to the Model Centred Architecture (MCA) paradigm, which was proposed by Heinrich C. Mayr in (Mayr et al. 2017). Details regarding the HBMS-system architecture are available in section 6.

HCM-L Modelling Tool. The HCM-L Modeller has been developed as a modelling tool that supports HCM-L to enable computer aided generation, integration, modification and management of behaviour and context models. A basic version of the HCM-L Modeller is available via the Open Models Initiative for download.

User Interface. HBMS user clients were intended to help users interactively and unobtrusively in performing their activities. During project progress the clients and their user interfaces have developed from rigid, mainly text oriented web-applications to personalised multimodal apps running on various mobile devices. Today the communication between the HBMS-System and the user clients works unidirectionally and environmental sensors are required to feedback user

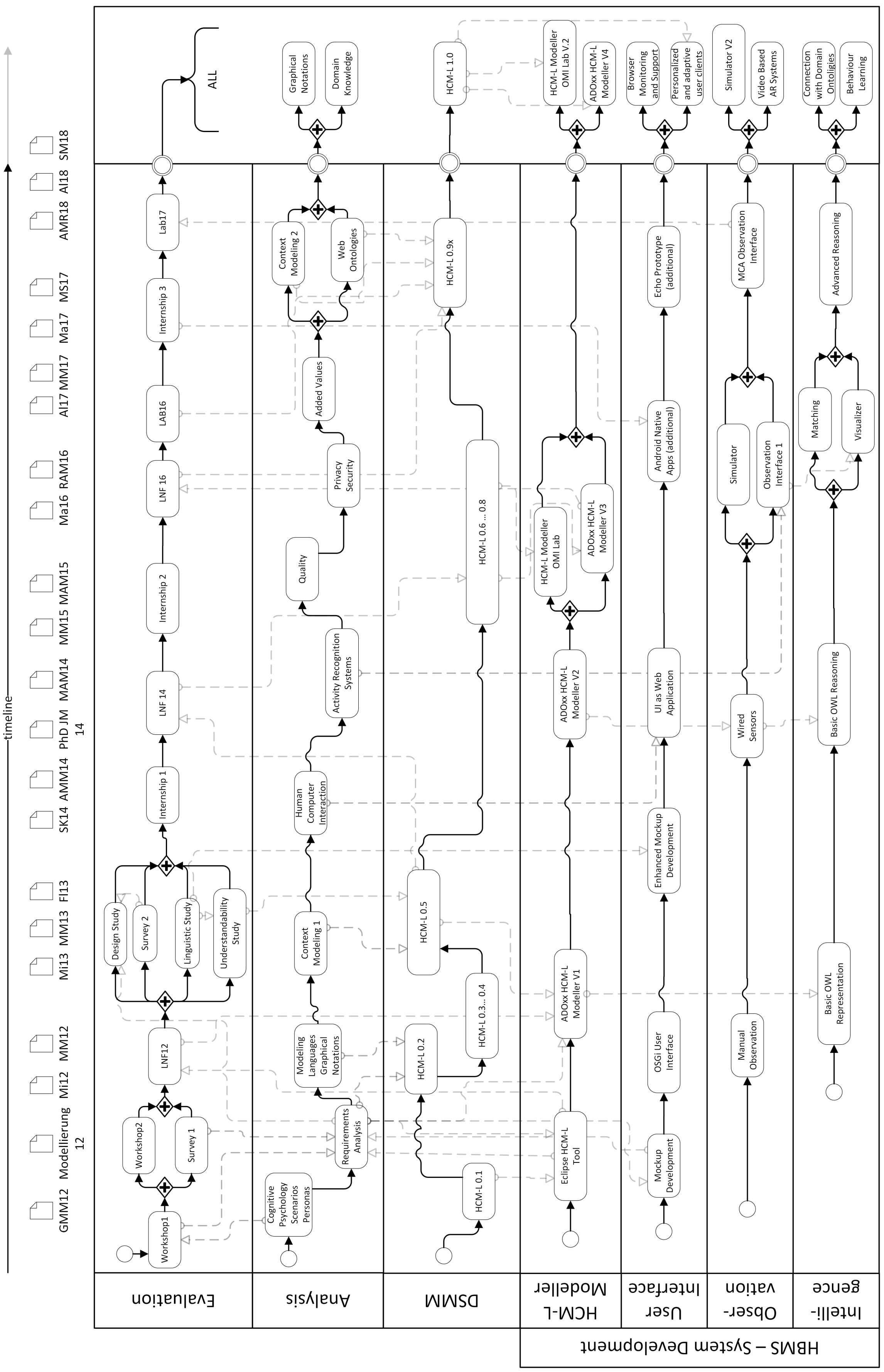


Figure 2: Past and Future Model of the HBMS lifetime

behaviour into the support system again. However, now we are going to expand the user client functionality in order to feedback user input in a bidirectional form e. g. via voice assistance to bridge the current activity recognition gap (Steinberger and Michael 2018).

Observation. Observation processes dealt with possibilities to monitor a user and to infer and recognize inhabitants' activities, changes or anomalies, continuously in real time in a progressive way based on activity models. We set up different labs using various networked sensor technologies and context management platforms and even developed an activity recognition interface to integrate latter. In the beginning of HBMS, we had high expectations regarding current activity recognition capabilities, but we had to realize at the end that current activity recognition approaches are too coarse to monitor all needed details for HBMS. As explained above, now we are going to add a user client to the HBMS-System which interacts with the user adaptively and in a smart way – to find out if a proposed activity has been done (as an alternative user activity recognition service) – and communicates the answer back to the cognitive assistance system. Intelligence. The HBMS-System increased its level of intelligence step-by-step by developing reasoning, matching and visualisation mechanisms. The first HBMS-System prototype included basic reasoning mechanisms within the HCM-L Modeller. The reasoning approaches to improve activity recognition were tested on laboratory datasets. The latest prototype covers the matching of recognized actions with operations in the knowledge base as well as a direct visualization of the matching results for end users.

The next sections will focus on the processes described above in more detail.

3 User Involvement and Evaluation

HBMS has been a user centred project and user involvement has been an integral part of the project from the very project beginning. The goal of our user evaluations was to derive system requirements from the user's perspectives mainly in the field of acceptance, interfaces, usability, interaction modes and model comprehensibility.

The evaluation pool in Figure 2 shows a rough overview what has been done for evaluation purposes:

During the first phase of the project, a particular attention has been paid to user involvement. Different types of user evaluations validated existing concepts and provided valuable input for requirements analysis. In a first **workshop**, we discussed with a group of about 40 senior students (attending the 'Senior Studium Liberale') their acceptance of AAL systems, possibilities for assistance and their ideas for user interfaces. In the second workshop, the group of senior students was extended by younger students and their desired assistance situations were discussed (see Figure 3).

Figure 3: Discussions in the second Workshop

Simultaneously an **empirical study (online survey)** with 203 persons of all ages was conducted on opportunities and obstacles of active assisted living (Mi13). By this way, the first HBMS prototype was strongly influenced by prospective users.

The *'Long Night of Research' (LNF)*, a biennial **public event** in Austria, where numerous national research institutions open their doors to the public, was repeatedly an important stage for us to present our intermediate results and get feedback on it. LNF has been welcomed by the public very well and the University attracted approx. 7000 visitors at each event. The scenarios used during our evaluations originated not only from the area of 'activities of daily living' (Katz 1983; Lawton and Brody 1969) but also from more technical ones (see Figure 4). LNF user feedbacks always enriched the subsequent project steps.

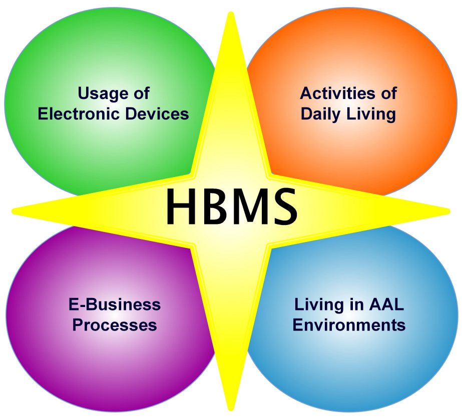

Usage of Electronic Devices
Activities of Daily Living
HBMS
E-Business Processes
Living in AAL Environments

Figure 4: Project relevant scenarios

In 2012 *(LNF12)* the first version of HCML-tool as well as a first support client were presented to approx. 50 volunteers. User's coffee preparation behaviour in cooperation with an automatic coffee machine was modelled, these models were discussed and different support possibilities were checked.

In 2014 *(LNF14)* an advanced prototype was presented to the public. HBMS had learned to sense some context information during the last 2 years and was presented in a first small experimental lab to approx. 50 volunteers. User interfaces on different clients were used to support some activities of daily living. The HCM-L Modeller was still a rapid prototype for analysis purposes integrating modelling and activity recognition capabilities.

In 2016 *(LNF16)* the HBMS-System had been grown up and was presented again in a particular lab environment where activities of daily living, the handling of electronical devices and online transactions were supported. Additionally, behaviour model visualizations and priorities for specific user clients were evaluated with approx. 100 heterogeneous volunteers.

As this lab environment had turned out to be very popular we evaluated system improvements based on LNF16 feedbacks a few month later again with 50 high school students.

To enhance the support client presented in LNF12, a **user design study** was performed, testing different user mock-ups in nine scenarios. The testing was done one at a time with 55 persons of all ages, gender and education levels and brought new insights into user surfaces and ways of user navigation and interaction (Strobl and Katzian 2014). Simultaneously an **empirical study (online survey)** was conducted to collect user client design ideas from 338 persons of all ages. The survey showed a high demand of autonomy of the end users, which means that they want to decide when to get support and how.

As support systems are supposed to interact with the user in a clear way and user manuals normally are not well suited to be used online or orally, linguistically reduced user instructions were tested in a **qualitative study** with app. 50 persons (Fliedl et al. 2013).

Moreover, a **qualitative study** including a qualitative content analysis (Mayring 2010) was carried out with 54 non-technology students to evaluate the intuitive understandability of HCM-L models and the notations of HCM-L concepts. (Michael and Mayr 2017)

Since 2013, we annually employed up to 6 interns for *evaluation purposes* during summer time. The group of internship 1 tested the tools, modelled different scenarios with the HCM-L Modeller and discussed their resulting models at home with friends and their relatives to investigate its comprehensibility. The group in internship 2 had to lay an eye on existing models and test the user client's surface, interaction and navigation capabilities. The group of internship 3 focused on possibilities for user modelling and robots as potential user clients.

Now, we are currently working to adapt the HBMS observation interface by integrating it with a wide variety of the existing activity recognition systems with traditional sensor based input and output formats. Evaluations to test its adaptivity and flexibility with the HBMS-System are currently running. Moreover, evaluations to test the observation interface with video based activity recognition systems have been planned for the next future.

We are also working on the evaluation of various notations for modelling the advanced HBMS context, namely user models, environment models and spatial models.

4 Analysis

The entire HBMS Project was accompanied by a continuous analysis of the *state-of-the-art of assistance systems*. *Projects* in the ambient assistance domain as well as *current research*, published at conferences and in journals, were systematically analysed according to their type of assistance. *Standards* in the AAL domain were evaluated and ongoing projects related to standards were pursued. Some of the topics, which were also of great interest to us, are listed below:

Cognitive Psychology. In an extensive analysis compendium, we investigated factors for the reduction of individual memory capacities and cognitive limits (e. g. stress, sleep deficits or depressions as age independent factors, as well as age related factors).

Scenarios. We analysed areas of assistance systems and identified the most important application scenarios of electronic devices, e- processes and everyday activities for our research, e. g. (Bonfiglio et al. 2008).

Personas. Personas are prototypical persons, who could be seen as future users of a system. Based on an analysis of the CURE Elderly Personas (Wöckl et al. 2011), we have identified relevant user groups for the HBMS-System.

Requirements Analysis. We identified requirements for our future modelling tool and investigated different technical solutions. (see Section 6.1)

Modelling Languages. As conceptual modelling played an important role for our project, we investigated which existing modelling languages were able to represent human activities and which language could be used as a base language for our DSMM to be developed.

Graphical Notations. Approaches for the intuitive understanding of modelling languages and graphical notations were investigated.

Context Modelling. We conducted a comprehensive analysis of context modelling approaches used in different domains. Whereas the first findings from (Kofod-Petersen and Cassens 2006) resulted in a general context structure focussing on behaviour (Michael and Mayr 2013), further work discussed the structural elements in detail (Michael and Steinberger 2017).

Knowledge Representation. We investigated different approaches for knowledge representation with a special focus on ontologies, semantic languages and related tools. We analysed different web ontologies for their usefulness to include general and domain knowledge into the HBMS knowledge base. Activity Recognition. Various systems for activity recognition were evaluated and categorized based on their recognition methodology. (Ranasinghe et al. 2016)

Human Computer Interaction. An important aspect for the development of user support clients in HBMS were useful technologies for human-computer interaction. Thus, speech recognition and synthesis, humanoid robots, and other technological possibilities were evaluated.

Quality. We analysed approaches for quality understanding, e. g. (Lindland et al. 1994; Schütte 1998), and defined useful quality categories for the HBMS project and techniques to reach a set of quality requirements.

As the project continued to make progress, we started to evaluate *'added values'*: In the initial phase of the HBMS-System, when the system needs to learn the behaviour of a person, no direct benefit is generated. Thus, we have evaluated existing products on the market on their compatibility with the HBMS-System and usefulness for end users.

Other important aspects for a real life usage of the HBMS-System were confidentiality, integrity and authenticity considerations. Thus, we evaluated the system and users' wishes and developed a *security and privacy concept*.

5 Domain Specific Modelling Method

It was clear from the very beginning, that conceptual models should build the core knowledge base of the HBMS-System. Thus, a Domain Specific Modelling Method (DSMM) including a modelling language called Human Cognitive Modelling Language (HCM-L) was created and further developed. The first stable version was presented in the dissertation "Cognitive Modelling for Assistance Systems" (based on Michael and Mayr 2013).

The first version of the HCM-L meta-model included operations, things and conditions as model elements and, as further specializations, different variants of operations and conditions as well as some sort of branching and merging concepts (called gateways). The modelling process was realized in a first prototype of the Modeller. Nevertheless, first evaluations showed us that several aspects were still missing.

Our former experiences in conceptual modelling with *KCPM (Klagenfurt Conceptual Predesign Model)*, a lean, user-centred language for software requirements modelling (Kop and Mayr 1998), helped us to carry out further improvements. Consequently, the KCPM concepts were evolved and adapted to cognitive modelling. Therefore, we could benefit from adopting from the KCPM approach the thing concept (a more intuitive way of abstracting from classes and attributes), as well as the approach for relating resources and actions.

Based on several project discussions, the second version of the meta-model was developed: It still focused on behavioural aspects and included an aggregation of several operations into an activity (called *Behavioural Unit, BU*), calling-, participating- and executing-connections between operations and things, person and location things as specializations of thing and different variants of connections like is-a, part-of and property connections. Obviously, this version (0.2) was hardly related to version 0.1. Most behavioural aspects remained stable up to the next versions.

HCM-L version 0.2 was published in (Mayr and Michael 2012), where we presented a semantic analysis based on control flow workflow patterns (van der Aalst et al. 2003). We showed that HCM-L was – for the domain of Ambient Assistance – sufficiently powerful and met the requirements for modelling human daily activities better than general purpose languages like BPMN or UML (see e. g. Wohed et al. 2005).

Figure 5 shows the excerpt of a BU model including two operations 'take a cup from the cupboard' and 'put the cup on the drip tray' as well as related things: the person 'Francis' as calling and executing thing and the 'cupboard' and 'coffee cup' as participating things.

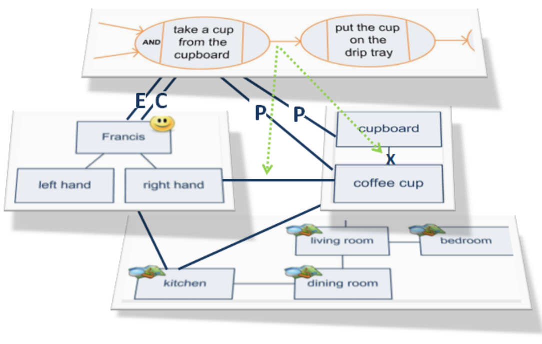


Figure 5: One step of the behaviour context in detail

Literature research in cognitive science and psychology took us to *activity theory* (Leont'ev 1978), which observed the nature of human activities on three levels: The level of the *activity* (the overall process), the level of the *action* (subtasks) and the level of *operations* that realize actions. While activities are driven by motives, each individual action pursues a specific goal and operations are related with instrumental conditions and constraints. Our HCM-L meta-model already included these levels but the essential concept of 'goal' was missing until HCM-L version 0.3.

We decided to move conditions into operations and realized that the concept flow, a connection between two operations, had to be added (version 0.4). One development stage further, we added some attributes to goals, operations and BUs and a relation between operations and properties of things. This version 0.5 of the HCM-L meta-model was published in (Michael and Mayr 2013).

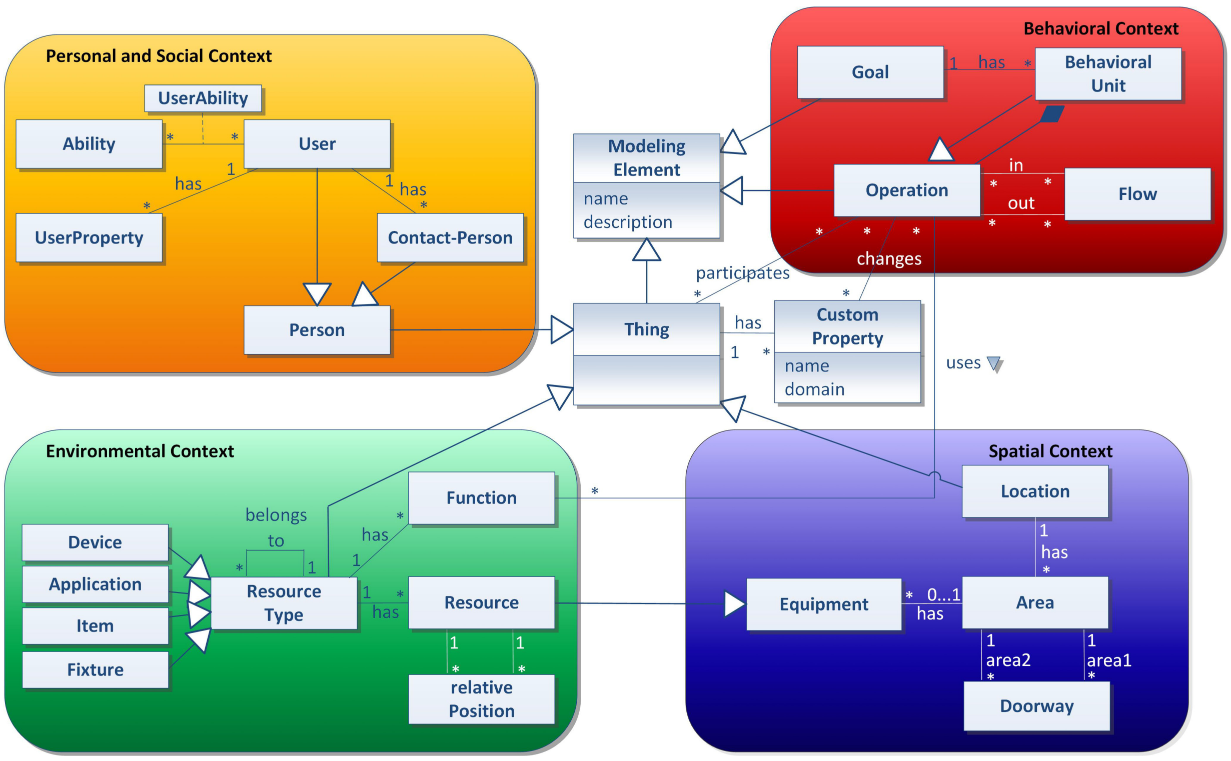


Figure 6: HCM-L Meta-Model v 0.93

The next changes were minor ones: some attributes were deleted, attributes of goals were move to the relation between operations and BU, a relation between goal and thing was added and a variant of things called service thing was added.

The next essential HCM-L enhancements were published in (Michael and Steinberger 2017), where the concepts regarding the structural context characteristics were defined in more detail. Accordingly, it was possible to model for example abilities of users (personal and social context), functions of different resources, different variants of resource types (environmental context) and equipment in different areas (spatial context). Figure 6 shows the current state of the HCM-L meta-model including the different context clusters (based on Kofod-Petersen and Cassens 2006).

One of the next HCM-L development steps was related to an approach to integrate semantic annotated user manuals into the HBMS-System: schema.org (Guha et al. 2016) was applied to semantically annotate such manuals. Corresponding mark-up data was used to import domain knowledge and "how to use"-resources, into the HBMS-System. Thus, the meta-model had to be partly adapted concerning instructions, warnings and problem situations (Steinberger and Michael 2018).

Next important HCM-L additions are in progress. They are related to *behaviour goals*: The concept goal needs to be investigated in more detail to be able to express, e. g., relations between different goals, hierarchies, and relations to domain and fundamental ontologies as a compensation for the semantic memory (general knowledge about the world and its' concepts).

Together with the HCM-L meta-model also our graphical notation changed. The elements were revised in several iterations based on evaluation feedbacks and because of findings from (Moody 2009), e. g., icons were added, the colour and thickness of different connections were changed for a higher visual distance and new graphical elements were added accordingly to additions in the HCM-L meta-model.

To make it clear, how to use our DSML systematically for creating models, we described the modelling procedure (Michael et al. 2015) based on the work of Karagiannis and Kühn (2002). It treated e. g., with which diagram type the modelling process should start or what aspects were to be modelled first.

Additionally, we have taken our experiences on creating a DSMM on a more general level (Michael and Mayr 2015) by describing the way of how to create a modelling method for ambient assistance, based on the work of Frank (2013).

In parallel to HCM-L development, we created a HCM-L modelling tool, which enabled us to work in practice with the DSML (see section 6.1).

6 HBMS-System Development

The development of the HBMS-System took place in *several stages*, whereby the prototypes were regularly evaluated (see section 3). This section sketches the most important HBMS-System versions from the first prototype architecture to the final model centred architecture of HBMS-System and particularly emphasizes the development of the HCM-L Modeller, the observation capabilities of HBMS-System and the intelligence of HBMS-System during the development process.

HBMS-System Version 1- Eclipse and OSGI

The focus in the first HBMS development phase was on the following aspects:

- Tool support in HCM-L modelling.
- Mapping of HCM-L models to a suitable knowledge representation for future reasoning purposes.
- First HCM knowledge base with suitable access interfaces
- First user support clients.

Figure 7 shows the architecture of the first HBMS (rapid) prototype evaluated at LNF 2012.

It included an Eclipse-based modelling tool, an ontology and a server and dialog component. The prototype realized the following functionalities: modelling the user behaviour by means of the first HCM-L version, transformation of these models into OWL-Lite (using JENA framework), storing the results in a filesystem, manual upload of the OWL files into a service domain, and user support via first mobile user clients.

HBMS-System Version 2 - ADOxx

Based on our experiences and evaluation results the next significant HBMS-prototype changed its paradigm. The next significant version of the HCM-L Modeller was based on the meta-modelling platform ADOxx (Fill and Karagiannis 2013), reasoning mechanisms were included directly in the HCM-L Modeller, and the user support client was extended to a web-based version. This prototype included also the following new functions: construction of more complex behaviour models according to the next HCM-L version, behaviour animation, embedding of media data, model export and import interfaces, transformation of models to OWL DL, and a first simple observation functionality using wired sensors. This prototype was very helpful to analyse and evaluate our requirements but was too monolithic and inflexible in its architecture.

HBMS-System Version 3 - MCA

The next prototype of the HBMS-System (which was presented and evaluated at the LNF16) was implemented according to the *Model Centred Architecture (MCA)* paradigm (see Figure 8). MCA was proposed in (Mayr et al. 2017) as a generalization of our experiences from several projects, including HBMS. According to the MCA paradigm, we understand information system development processes as mere modelling processes and focus on models (formed with the means of DSML) at any development stage up to the running system. The models are taken as the core of a system for both the application functionality and the system's interfaces.

According to (Mayr et al. 2017, Shekhovtsov et al. 2018), the MCA paradigm is defined on the following three levels:

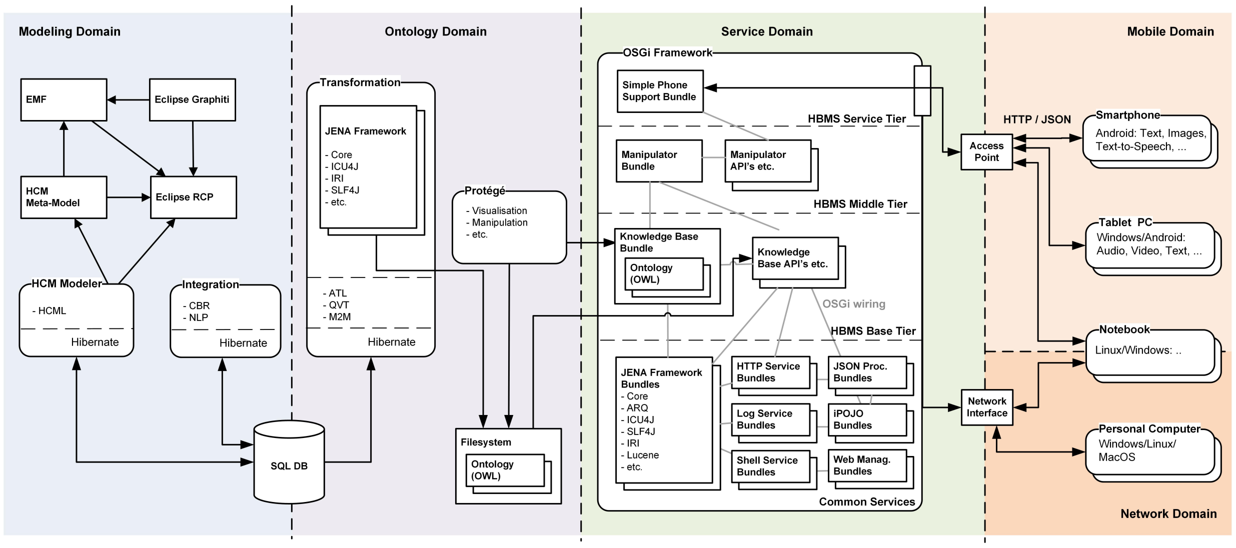


Figure 7: Architecture 2012 (adapted from (Michael et al. 2013))

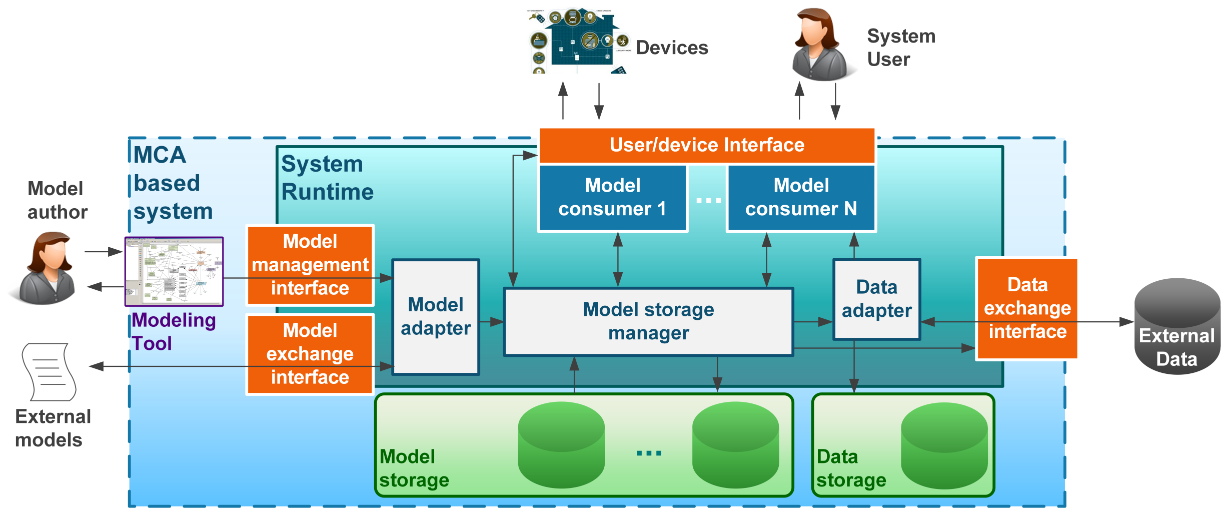


Figure 8: MCA patterns (adapted from (Mayr et al. 2017))

1. as a meta-model hierarchy on MOF levels: M2 (the DSML meta-model definition based on a meta-meta-model), M1 (the specification of the system components and its data) and M0 (models of concrete objects, functions and processes); we call this definition a *semantic core* (Object Management Group OMG: n.d.);
2. as a language specification hierarchy representing the core semantics using different syntactic constructs of the appropriate representation languages. The hierarchy descends from grammar definitions, to concrete grammars for the languages representing semantic concepts, to concrete language representations of those concepts. This concrete language representations can even refer to constructs from different levels, which means the concrete languages can be used to represent instances, models, meta-models, and meta-meta-models, even together in the same project;
3. as a set of architectural patterns including the *Modelling Tool* pattern (components used for

creating models), the *Model Transfer Interface* and the *Model Adapter* patterns (for the components providing the models to the consumer system), and the *Model Consumer* pattern which describes the components which use the models to provide the functionality of the MCA-based solution. It also includes the *Device and User Interface* patterns which are the model-based interfaces to the model consumers and the *Data Storage* pattern (see Figure 9).

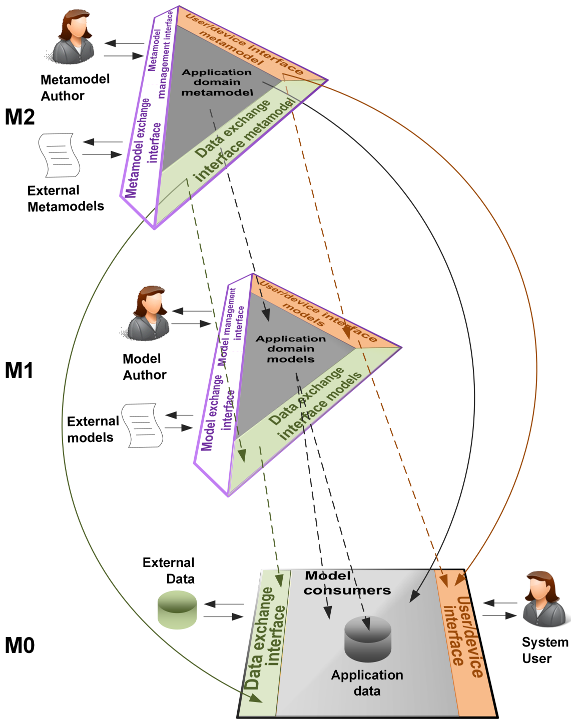

Metamodel Author
M2
External Metamodels
Metamodel exchange interface
Metamodel management interface
User/device interface metamodel
Application domain metamodel
Data exchange interface metamodel
M1
Model Author
External models
Model exchange interface
Model management interface
User/device interface models
Application domain models
Data exchange interface models
M0
External Data
Data exchange interface
Model consumers
Application data
User/device interface
System User

Figure 9: Model Centered Architecture (adapted from (Mayr et al. 2017))

In version 3, the HBMS-System was redefined to comply with MCA. The corresponding architecture is depicted on Figure 10.

According to this architecture, the HBMS-System includes the following components:

1. **HCM-L Modeller** is used for creating and maintaining HCM-L models and the transformation of the created models to the HBMS kernel using the HBMS model transfer interface.
2. **HBMS Observation Interface (HBMS-OI)** includes an activity recognition system adapter (HBMS ARS adapter). Together, they form a middleware listening to inputs coming from the activity recognition system and making it HBMS-compliant; these components implement the Device Interface MCA pattern.
3. HBMS Kernel is the central component of the system. It is accessible via the set of kernel interfaces and contains the following internal components (all implementing Model Consumer MCA pattern):
   a) *HBMS Observation Engine* responsible for communicating to the Activity Recognition System (ARS) through HBMS-OI;
   b) *HBMS Behaviour Engine* responsible for handling the behaviour data arriving from the HBMS Observation Engine in context of the current HCM;
   c) *HBMS Support Engine* responsible for controlling the behaviour of the assisted users;
   d) *HBMS Data Management Subsystem* responsible for managing HBMS data;
   e) *HBMS knowledge base* which stores the current (HBMS situational cache) and historical data (HBMS case base) together with its description (HCM storage) in a knowledge-oriented format;
4. **HBMS kernel clients** (monitoring client, simulator client and care worker client) to support monitoring and administrative tasks based on the information provided by the kernel.
5. **HBMS support clients** which implement multi-modal interfaces for end user support.

For evaluation purposes, it is possible to install and run all HBMS-System components on one computer as 'HBMS-System in a box' and to use the 'HBMS Simulator' to 'emulate the user behaviour in a smart lab' virtually (see Section 6.2).

The next sections describe the development stages of the HCM-L Modeller (section 6.1), the

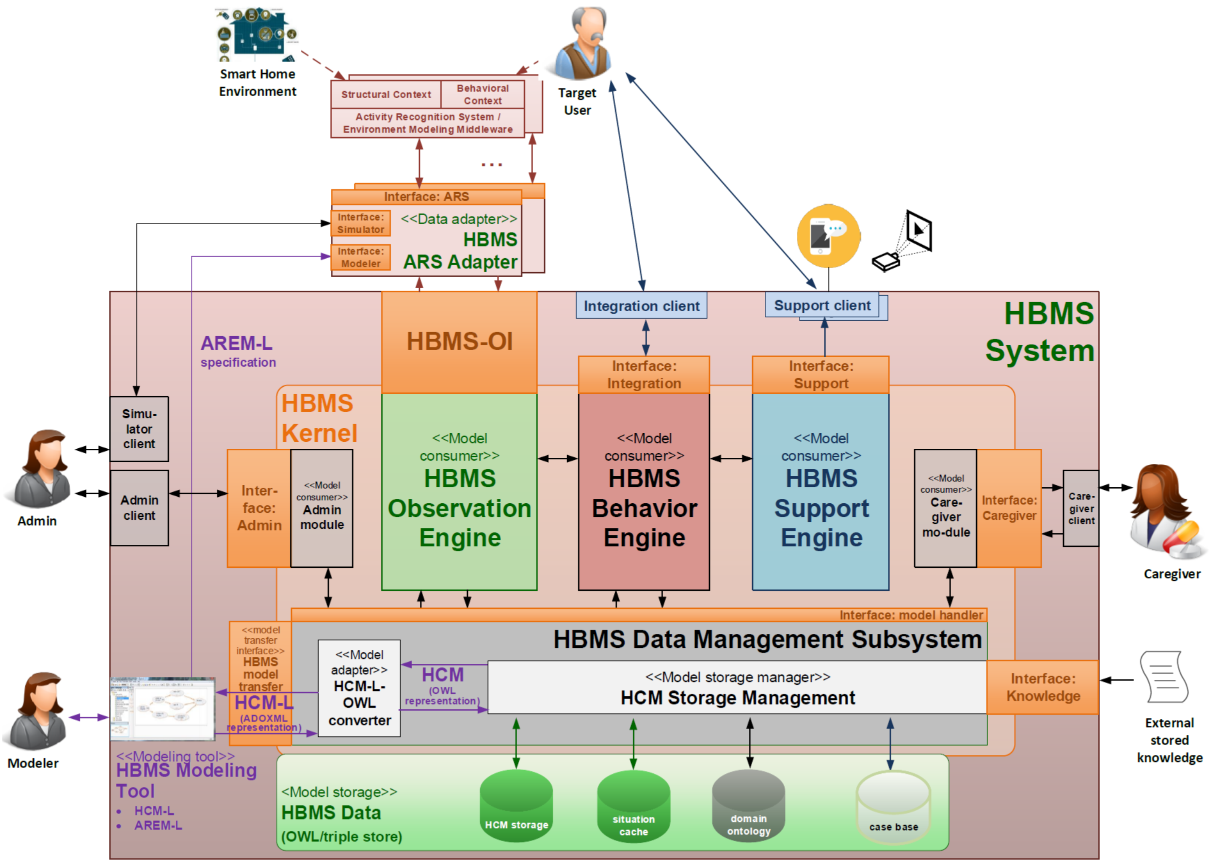

Smart Home Environment
Structural Context
Behavioral Context
Activity Recognition System / Environment Modeling Middleware
Target User
...
Interface: ARS
Interface: Simulator
Interface: Modeler
<<Data adapter>> HBMS ARS Adapter
Integration client
Support client
HBMS System
AREM-L specification
HBMS-OI
Interface: Integration
Interface: Support
HBMS Kernel
Simu-lator client
Admin client
Admin
Inter-face: Admin
<<Model consumer>> Admin module
<<Model consumer>> HBMS Observation Engine
<<Model consumer>> HBMS Behavior Engine
<<Model consumer>> HBMS Support Engine
<<Model consumer>> Care-giver mo-dule
Interface: Caregiver
Care-giver client
Caregiver
Interface: model handler
HBMS Data Management Subsystem
<<model transfer interface>> HBMS model transfer
<<Model adapter>> HCM-L-OWL converter
HCM (OWL representation)
<<Model storage manager>> HCM Storage Management
Interface: Knowledge
External stored knowledge
Modeler
HCM-L (ADOXML representation)
<<Modeling tool>> HBMS Modeling Tool
• HCM-L
• AREM-L
<Model storage>> HBMS Data (OWL/triple store)
HCM storage
situation cache
domain ontology
case base

Figure 10: Architecture 2017 (adapted from (Mayr et al. 2017))

observation components (section 6.2), the HBMS-System intelligence (section 6.3) and the user interfaces (section 6.4) in more detail.

6.1 HCM-L Modeller

The HCM-L Modeller was one of the first system components realized, as we needed concrete models, created accordingly to the HCM-L meta-model definition, to be able to move on with the work on other components and to define interfaces.

A first prototype was realized with the Eclipse Rich Client Platform (RCP). The HCM meta-model was created in Eclipse Modelling Framework (EMF) format. EMF also supported “models to text” and “models to models” transformation (Oldevik et al. 2005), which was necessary for model validation and analysis purposes. For persistence layer implementation, we used a MySQL database and Hibernate.

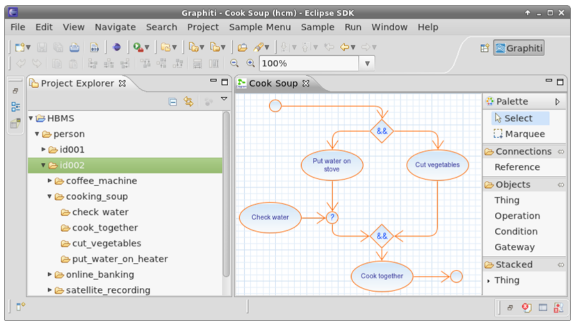


Figure 11: Modelling Tool in Eclipse

Figure 11 shows a screenshot of this prototype. As described in section 5, we had several severe changes in the meta-model of our DSML and realized, that such an incremental development process was connected with enormous workloads for changing the meta-model and graphical representations in the Eclipse prototype. Moreover, some improvements to achieve a better end-user understanding were not feasible within its' graphical interface.

After this first prototypical requirements analysis, we performed a systematic collection of requirements for the next version of our HCM-L Modeller along the characteristics listed in (Kaschek and Mayr 1996).

Based here-on we decided to test the ADOxx Meta-Modelling Platform (Karagiannis and Kühn 2002) and finally chose it to implement the next version of HCM-L Modelling Tool (Michael et al. 2014, 2015).

The crucial factors for this decision were:

- The possibility of fast prototyping: a first stable version was completed in a month,
- Ease of changes in the meta-model and automatic tool adaptation,
- Availability of a simulation component,
- Easy to combine with external software, e.g. for reasoning (Al Machot et al. 2014),
- Availability of analysis functionalities,
- Easy to realize consistency checks,
- Highly professional general software and developer support.

The HCM-L meta-model elements were mapped to classes of the ADOxx meta-model and described using the proprietary ADOxx Library Language (ALL). ALL used the constructs defined in the ADOxx meta-meta-model (level M3) (Fill and Karagiannis 2013). ADOxx already included export and import functionalities in and from the ADOxx Description Language (ADL) format or a generic XML format. We adopted these to allow the transformation of HCM-L models to other representation formats, as used e. g. by inference or reasoning tools.

The HBMS team joined the *Open Modeling Initiative (OMI)* (see Karagiannis et al. 2007) and the related *Open Models Laboratory (OMiLAB)*: The HCM-L Modeller (see Figure 12) is, in a basic version, freely available at the OMiLAB webpage[1]. This version includes the manual creation of sequences, task context, structural context and BU models (BUMs) as well as macros. Furthermore, it is possible to step through BUMs and related sub-processes by using the graph analysis functionality of ADOxx. Model transfer between HCM-L Modeller and HBMS-System kernel was realized in this version via the adapted export functionalities mentioned above.

In 2016/17, the HCM-L Modeller was enhanced based on the MCA paradigm as a concretisation of the *Modelling Tool* architectural pattern (see Figure 8). As such concretisation, besides supporting the latest HCM-L constructs, it was able now to communicate with the HBMS kernel by transferring the XML representation of the selected subset of models (describing the context of the supported person) through the communication channel controlled by that kernel instance (e. g. represented as a web service call).

To control such transfers, we extended HCM-L with the new Configuration Workspace Model which connected the references to HBMS deployment sites (kernel installations) to the references to context models to be deployed at these sites. On modeller's demand, a model is employed to find the server over the network and conduct the model transfer by means of the deployment script.

6.2 Observation

The HBMS-System should be able to learn user's behaviour and to apply this knowledge to support the user unobtrusively in real-time. Thus, we investigated possibilities to observe users to recognizing their activities. Additionally, we developed an activity simulator tool which allowed us to generate sensor data for our scenarios via a web interface to be able to test the system independently from activity recognition systems.

6.2.1 Observation Interface

During the first project phase, we analysed existing user observation technologies and tools. The first HBMS-System prototype, however, was not able to automatically observe a user and to

[1] http://austria.omilab.org/psm/content/hcml/info

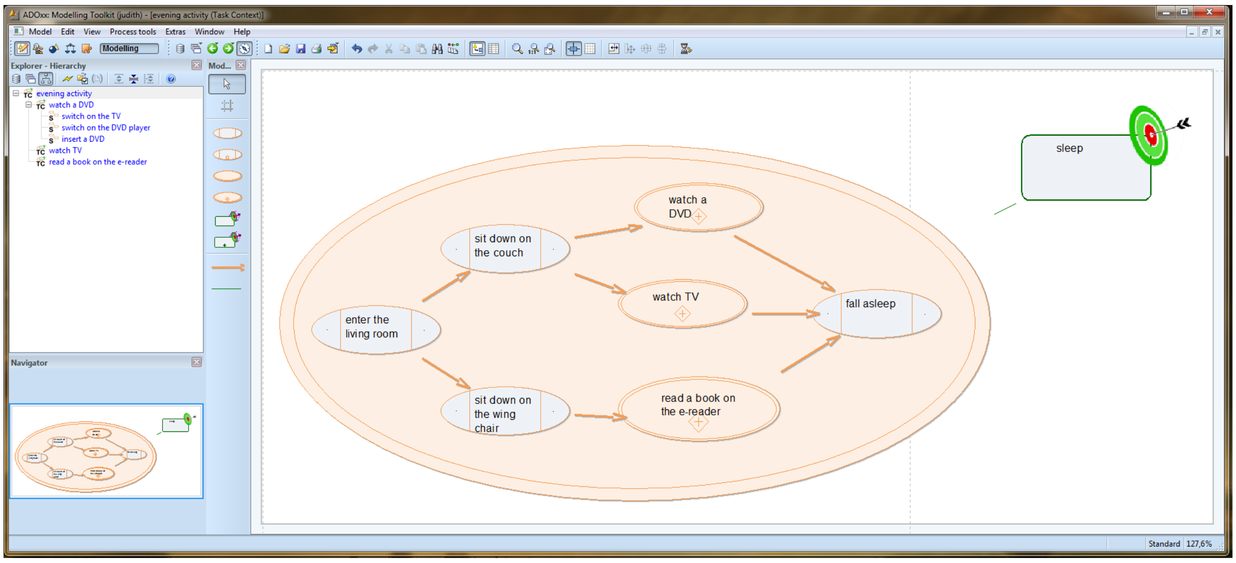


Figure 12: HCM-L Modeller 2016

detect his activities. The user behaviour had to be observed manually and communicated to the HBMS-System.

The first form of automatic observation was introduced in the HBMS-System prototype which was evaluated during LNF14. Simple wired sensors were connected directly to the HBMS-System, which also implemented a simple context management system component. This component was able to identify simple activities of a user and to forward them to provide required user support.

In developing the next observation prototype, we followed the MCA paradigm. Monitoring the user and handling the captured user context was outsourced to specialized context management middleware. The communication between this middleware and the HBMS kernel took place by means of the *HBMS observation interface (HBMS-OI)*. The observations obtained through this interface were subsequently processed and analysed by the HBMS observation engine, as shown in Figure 10.

We tested this HBMS-System prototype in our lab environment (see Figure 13) and used Nimbits[2] and FHEM[3] as middleware systems. To monitor user behaviour, we used a network of wired and wireless sensors supporting different protocols. The sensor network consisted of motion, pressure and contact sensors. All sensor data was gathered by means of the Arduino Coordinator [4] . The coordinator was responsible for identifying relevant sensor data changes and for writing the data corresponding to these changes to the relevant Nimbits channels. Also, the coordinator was responsible for comparing sensor values with given threshold values, e. g. if pressure sensor value exceeded 700 mbar it sent value “ON” or “1” to the relevant channel.

In our setup, for the identification purposes, we categorized Nimbits channels into two categories as toggling and firing channels.

Toggling channels were responsible for storing binary values, i.e. “1” or “0”, based on the ON/OFF value produced by the binary sensors. Toggling channels were mainly used to track events such as opening/closing doors, putting/lifting things etc. which typically generate binary values from the corresponding sensor devices. When the toggling channel detected a value change, e.g. from “0” to “1” or from “1” to “0”, it generated an event. For example, if a person “opens a door”, the

[2] https://www.nimbits.com/
[3] https://fhem.de/
[4] https://www.arduino.cc/

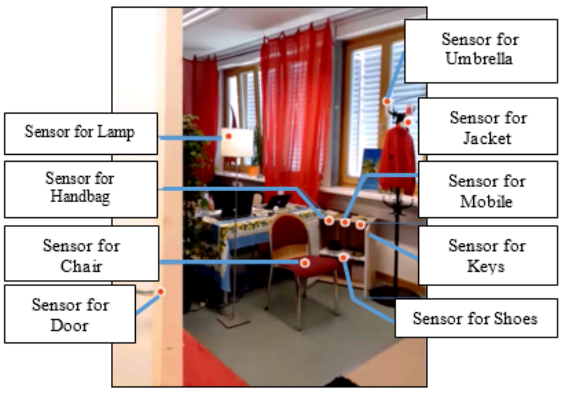


Figure 13: Sensors in the Lab

sensor connected to the door sent the corresponding signal to the coordinator, which stored the sent value in the given toggling channel. Consequently, the toggling channel generated an "open door" event and created meaningful semantic data to describe the event, combining it with the predefined meta-data.

Firing channels are used to store sensor data values which are typically static. Such channels use the channel timestamp change to generate a corresponding event. For example, a remote control button can be connected to the sensor which sends a predefined value on every click e.g. “user pressed the STOP button”. It is hard to recognize an activity by looking only at such value, so firing channels looks at the timestamp change to generate an event.

The problem with the approach described above was, that the low-level data available in a channel was directly accessed through the HBMS-OI. As a result, at that point in time the coupling between the middleware system and the HBMS-OI was still very high. Besides that, we were not satisfied with this solution, as only very simple atomic activities could be observed. Accordingly, we were looking for a way to make arbitrary activity recognition systems interoperable with HBMS-OI.

To overcome these deficiencies, we introduced a language to describe human behaviour observations called *AREM-L (Activity Recognition Environment Modelling Language)* (Mayr et al. 2017).

The main process of integrating the observations with HBMS-OI based on AREM-L is shown in Figure 14.

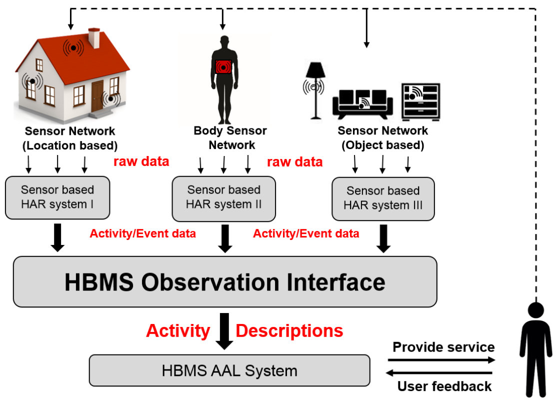

Sensor Network (Location based)
Body Sensor Network
Sensor Network (Object based)
raw data
raw data
Sensor based HAR system I
Sensor based HAR system II
Sensor based HAR system III
Activity/Event data
Activity/Event data
HBMS Observation Interface
Activity
Descriptions
HBMS AAL System
Provide service
User feedback

Figure 14: Main processes of the HBMS-OI

According to the MCA paradigm, in the current version of HBMS-OI, AREM-L is used to serve as a conceptual foundation for converting the data produced by the activity recognition systems (such as sensor based human activity recognition, HAR systems of different kind, as shown on Figure 14) to the semantic structures suitable for further processing in HBMS kernel (Mayr et al. 2017). The process includes the following steps:

1. Both human observation data and the contextual information are sent in real time from the HAR systems to the HBMS-OI;
2. HBMS-OI produces human observation recognition semantic structures out of this data based on the available AREM-L models;
3. These semantic structures are sent to the HBMS-System to be processed with a goal of providing the required assistance.

Evaluation activities to test the adaptivity and flexibility of the HBMS-OI are currently under way. Additionally, for the next future, we are going to evaluate the observation interface with video based activity recognition systems.

We are currently working on the integration of this version of the HBMS OI and the HBMS-System.

6.2.2 Simulation

Despite of all the successes with the observation interface for HBMS, we had to state the following: we had high expectations into current activity recognition capabilities in the beginning of HBMS but we had to realize at the end that current activity recognition approaches are too coarse to monitor all needed details for HBMS.

In order to overcome the weakness of the current activity recognition possibilities, we developed an activity simulator tool which allowed us to generate sensor data for our scenarios via a web interface. This generated data was forwarded to our context management middleware system in the same way as 'real' sensor data. This also allowed us to set up 'virtual labs' to test the HBMS-System. The HBMS simulator contained a set of simulator controls organized into control groups.

We distinguished the following types of simulator controls:

1. Toggling controls used to send one of the two alternative values when clicked, the value sent is the opposite of the current value.
2. Firing controls used to send the same value on every click.
3. Value-based controls used to send the user-selected value on request.

Figure 15 shows the controls which can send a value selected by means of radio button or select list. For every control, it was also possible to see the value which corresponded to the current value of a real sensor (if one was connected). This value was updated automatically when the sensor value was updated.

This tool allowed us also to simulate temperature and humidity meters to provide the flow of indoor and outdoor temperature and humidity data. The simulator provided the data as coming from these meters and was used to simulate different environmental settings, e.g. hot/cold or rainy/dry weather. The data from the simulator then went to our context management middleware which made it available to the HBMS-OI through additional specific channels (providing the currently simulated value when asked).

6.3 Intelligence

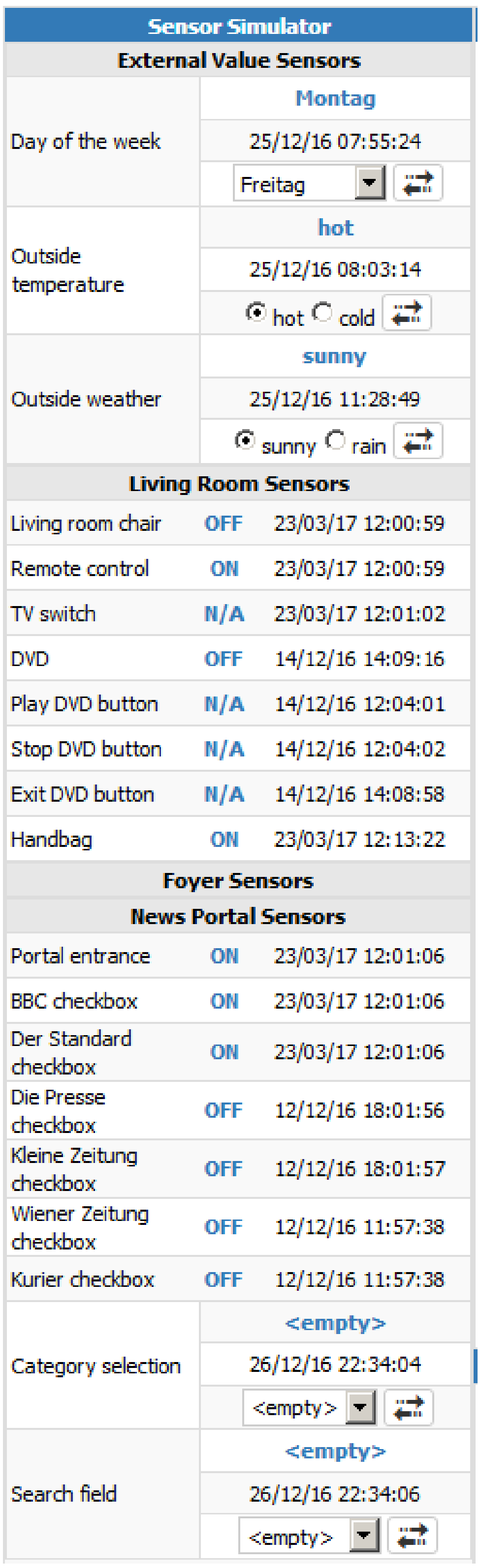
The first version of the HBMS-System had a *low intelligence level*: Models were created by hand including the integration of different models, only a part of the model concepts were translated into OWL, the structure accepted by the dialogue component was very tight and it included only text and no pictures or videos.

The next prototype of the HBMS-System included *basic reasoning mechanisms* which were included in the HCM-L Modeller. Sensors were connected with the HCM-L Modeller and the first support hints were presented at LNF14. Additionally, models were exported in XML format, which was used as an input for the knowledge base of the answer set programming solver. This reasoning module was able to calculate which next operation users should execute (see 'Reasoning for human support' in section 6.3.1 for details). Moreover, reasoning approaches for supporting activity recognition were tested on laboratory datasets (see 'Reasoning for Human Activity Recognition' in section 6.3.1 for details).

The *latest prototype*, presented at LNF16, included the matching of detected actions with the actions in the knowledge base and a direct visualization of the results. The matching component tries to find observed behaviour in formerly detected and saved behavioural units and returns if the action was correct, wrong in relation to its order in a behavioural unit or the conditions performed, a reverse action or a yet unrecognised one (see section 6.3.2 for more details). The Visualiser, as part of the kernel monitoring client, provides a view on the current state of the matching session (see section 6.3.3 for more details).

6.3.1 Reasoning

The reasoning within the HCM-L Modeller has been divided into two major modules; (a) the reasoning module to recognize human activity and (b) the reasoning to choose the optimal step



Sensor Simulator

External Value Sensors		
Day of the week	Montag	
	25/12/16 07:55:24	
	Freitag	
Outside temperature	hot	
	25/12/16 08:03:14	
	● hot ○ cold	
Outside weather	sunny	
	25/12/16 11:28:49	
	● sunny ○ rain	
Living Room Sensors		
Living room chair	OFF	23/03/17 12:00:59
Remote control	ON	23/03/17 12:00:59
TV switch	N/A	23/03/17 12:01:02
DVD	OFF	14/12/16 14:09:16
Play DVD button	N/A	14/12/16 12:04:01
Stop DVD button	N/A	14/12/16 12:04:02
Exit DVD button	N/A	14/12/16 14:08:58
Handbag	ON	23/03/17 12:13:22
Foyer Sensors		
News Portal Sensors		
Portal entrance	ON	23/03/17 12:01:06
BBC checkbox	ON	23/03/17 12:01:06
Der Standard checkbox	ON	23/03/17 12:01:06
Die Presse checkbox	OFF	12/12/16 18:01:56
Kleine Zeitung checkbox	OFF	12/12/16 18:01:57
Wiener Zeitung checkbox	OFF	12/12/16 11:57:38
Kurier checkbox	OFF	12/12/16 11:57:38
Category selection	<empty>	
	26/12/16 22:34:04	
	<empty>	
Search field	<empty>	
	26/12/16 22:34:06	
	<empty>	

Figure 15: Simulator including different 'sensors'

for human support based on HCM-L models. The aim of the first module is to identify the activity of the observed person to advise her/him about what is the optimal operation to be executed using the second module.

Reasoning for Human Activity Recognition. Regarding human activity recognition, different approaches have been developed in the frame of the HBMS project independently to be applied for real life scenarios. The proposed approaches show promising results using not only the HBMS laboratory dataset but also the well-known human activity CASAS dataset (Cook et al. 2013). The datasets are collected based on real life scenarios using different types of non-intrusive sensors that are mounted in real smart homes. The approaches can be summarized as follows:

1. A reasoning method based on answer set programming that uses different types of features for selecting the optimal sensor set, and a fusion approach to combine the beliefs of the selected sensors using an advanced evidence combination rule of Dempster–Shafer theory (Al Machot et al. 2018a) (see Figure 16).
2. A reasoning method based on a windowing approach for analysing sensor data to identify the best fitting sensors that should be considered in the selected window of sensor events. The second contribution proposes a set of different statistical spatio-temporal features to recognize human activities using Support Vector Machines (Al Machot et al. 2017).
3. A human activity recognition approach based on a Recurrent Neural Networks (RNN) model, which is trained based on dynamic systems perspective during the weight initialization process (Al Machot et al. 2018b).

Reasoning for human support. Regarding the human support, an approach based on Answer Set Programming (ASP) has been proposed. The aim was to solve an optimization problem depending on the following factors; (a) the importance of performing an operation according to the user history; (b) the cost value of choosing an operation

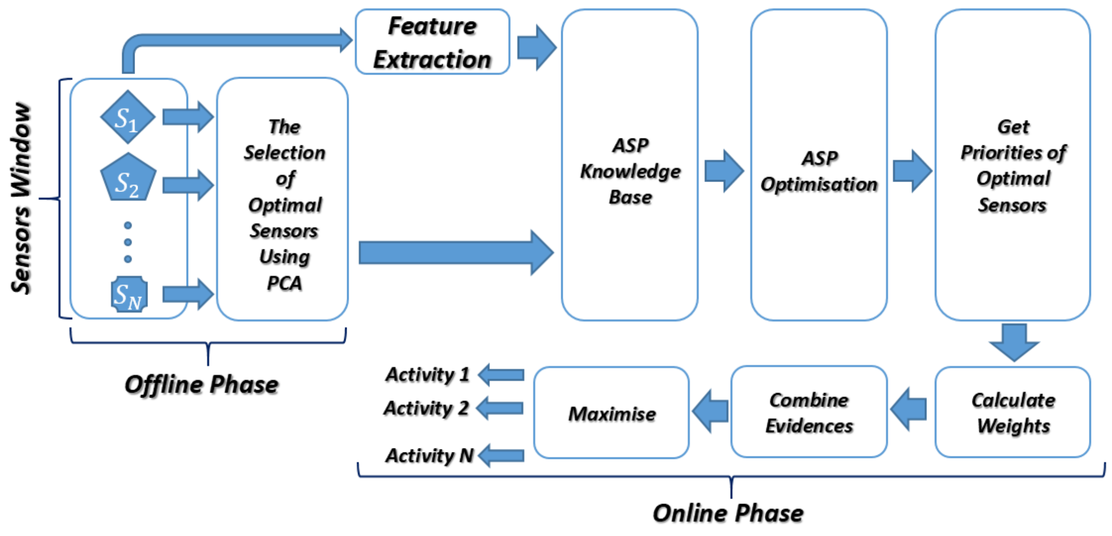


Figure 16: Reasoning system structure for activity recognition

based on the similarity between the current user profile and other users; (c) the time when the operation should be performed (Al Machot et al. 2014).

6.3.2 Matching

Another important development step was the implementation of the matching component *(HBMS matcher)*. Its aim is to match the semantic structures corresponding to the recognitions of the observed behaviour against the current HCM, to find a behavioural scenario (behavioural unit) including the observed action and a position of the action inside this scenario. Finding such match is a prerequisite for predicting next actions and providing support.

The matcher recognizes the following four categories of actions based on the recognition information coming from the observation engine:

1. Correct actions which are performed correctly based on the given BU;
2. Wrong actions which are present in the model but have been performed in the wrong place or under wrong conditions (i. e.. taking the shoes before going to the foyer);
3. Reverse actions which has been performed to revert the effect of the previous (possibly incorrect, but also correct) action e. g. "put the handbag back ";
4. Unrecognised actions which are not present in the model.

We based the matching algorithm on two main categories of checks:

1. Finding the set of possible matches based on changes in state (*state-based checks*). This is supported through accompanying the operations in the model with capturing state snapshots: every such snapshot describes the set of state configurations which are considered correct for entering the specific operation (capturing pre-operation snapshot) and for exiting this operation (capturing post-operation snapshot). The match is found by comparing the observed state snapshots of the observation data structure coming from the observation engine with the capturing snapshots of all operations defined for the available behaviour models.
2. Narrowing or modifying the set of matches based on checking preconditions (*condition-based checks*). This is supported through accompanying the operations in the model with parsed precondition specifications which include the internal representation of the pre-

condition code. The conditions from the possible matched operation are checked for the observation data structure, the match is found if this condition evaluates to true as a result of this check.

Based on the above set of checks, the correct and wrong actions can be recognized as follows:

1. The action is recognized as correct if the state coming with the observation completely matches the state accompanying the particular snapshot in the model, the changes introduced into the state are the same for both observation and the model operation, and the preconditions for this action are fulfilled.
2. The incorrect actions are recognized by matching the changes introduced into the state but missing the match for the precondition and post-condition state as a whole.

All recognized actions are encapsulated into the match result structures containing the type (correct, wrong etc.) and the timestamp of the match, and with the information about the matched operation and the existing reverse operation; the matcher also interacts with the prediction component to include the information about the predicted next operation into the match result.

As a part of the matching session, the matcher maintains two separate session queues collecting *"match result"* objects: the right action queue for correct actions; and the wrong action queue for the incorrect actions. The reverse actions are not added to these queues, they lead to deleting the last action from the queue. As a result, it is possible for the user to revert the damage caused by incorrect actions by performing the reverse actions in the reverse order, and go back as much steps as he/she wishes.

The match result is transferred to the support engine to serve as a basis for providing support.

6.3.3 Visualising

The next step was implementing the *kernel monitoring client*. This client is shared by both admin and caregiver interfaces of the kernel. It allows to monitor the events processed by the kernel, execute simulations of the sensor events, and perform administrative activities. It is implemented as an AngularJS client application communicating with the kernel by means of JSON-based API.

The main part of the monitoring client is the *HBMS Visualizer*. It allows for the user to see the current state of the matching session by:

1. Showing the graphical representation of the available BUs;
2. Reacting to the changes initiated by the HAR system through the HBMS-OI by highlighting specific diagram elements;
3. Providing a scalable display making possible to zoom to a specific subset of operations.

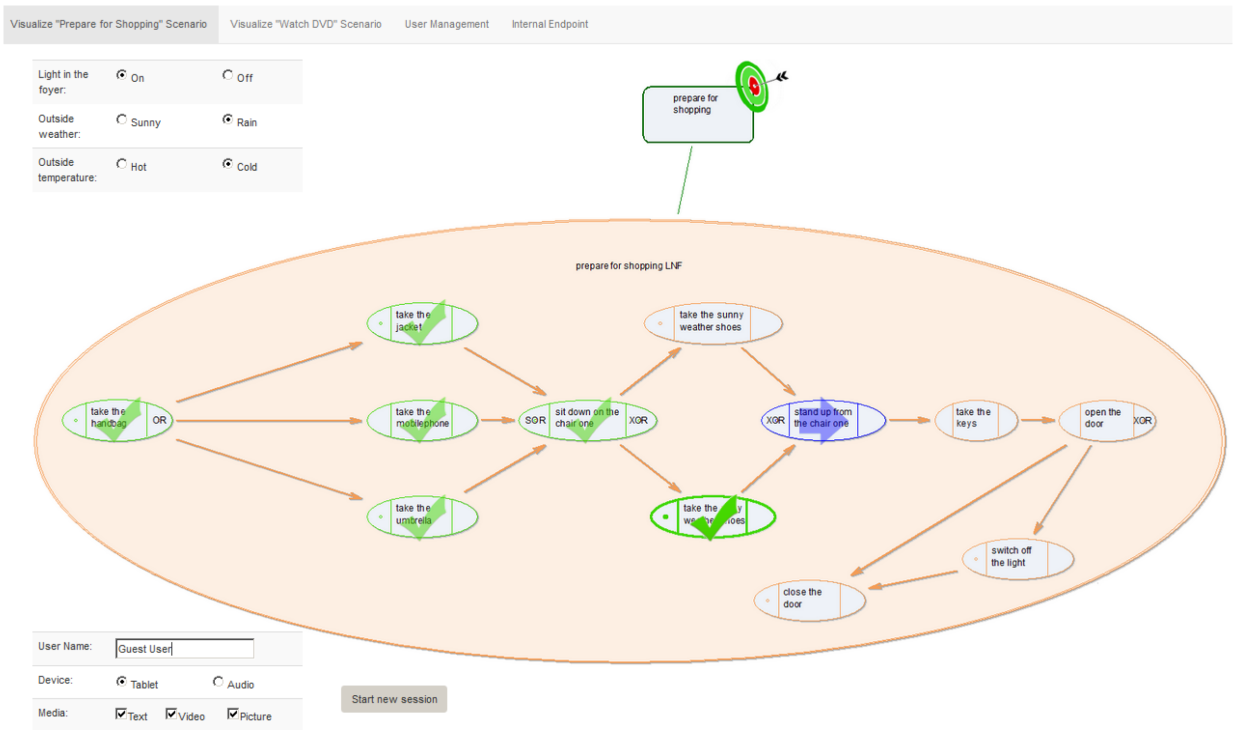
The Visualizer supports six different states of the operation element and three states of the goal element, a state (e. g. “correctly/wrongly matched as a session’s past operation”, “correctly/wrongly matched as a current operation”, “predicted”) corresponds to the specific colour and thickness of the element’s outline.

The Visualizer queries the state of the matching session several times per second and changes the display based on the obtained information. As a result, it is possible to see:

1. The position of the user in the current BU;
2. If the user made a mistake in handling the scenario;
3. The set of operations the user has been performed in the current session;
4. The number of steps the user has to go back to fix the wrong actions;
5. The predicted next operation or operations;
6. If the user has reached the goal.

On Figure 17, the user has just performed "stand up from the chair" operation after correctly performing "take the sunny weather shoes" and other preceding operations in the behavioural unit, the "take the keys" operation is shown as predicted.

The Visualiser also allows switching between behavioural units by means of a list of graphical



Light in the foyer:	[✓ On]	Off
Outside weather:	Sunny	[✓ Rain]
Outside temperature:	Hot	[✓ Cold]

User Name:	Guest User		
Device:	[✓ Tablet]	Audio	
Media:	[✓ Text]	[✓ Video]	[✓ Picture]

Figure 17: HBMS Visualizer showing the current state of the matching session

depictions of the behavioural unit models (at the bottom of Figure 17). Every such depiction is a scaled down version of the Visualizer display, it is updated dynamically when matching activities are performed by the kernel.

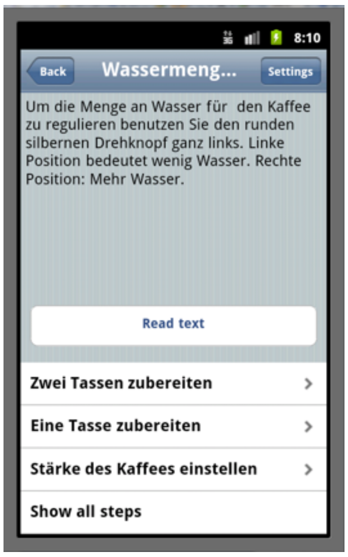
6.4 User Interface

The first dialogue component of HBMS to help users interactively and unobtrusive in performing their activities was based on first results of our requirements analysis. This HBMS *'Support 1'* was developed as a prototype for user evaluations in the context of a master thesis. It was supported by android tablets and mobiles and developed using PhoneGap, OSGi and webservice technologies.

The texts and next steps were derived in a lean OWL representation directly from the conceptual models in the HCM. This information was transformed into a representation of one single step and navigation possibilities to possible following steps (see Figure 18). Existing text-to-speech technologies were used to provide a voice output. This first support client prototype was used to evaluate the HBMS-System in LNF12: The participants used a coffee machine in our station at one end of the university campus, where their behaviour was manually modelled by team members. Afterwards, they moved to our station at the other end of the university campus where exactly these sequences were presented step-by-step on a tablet or a mobile phone.

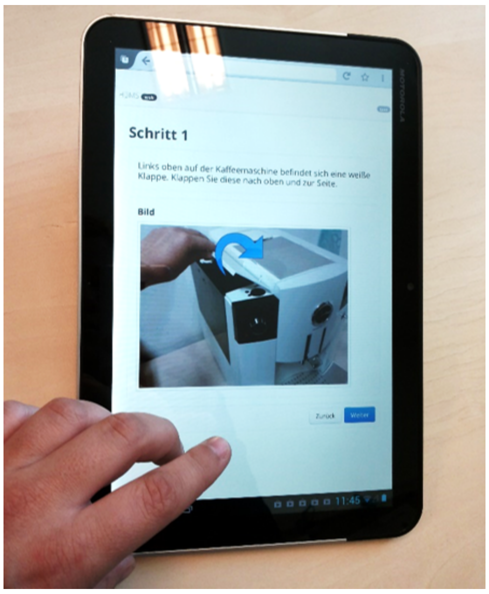
Learning from the first user feedbacks (e. g. variable text sizes, louder voice outputs, including graphics) we developed *enhanced support clients* for HBMS and their appropriate user interfaces. We have evaluated these ideas in 2013 by developing different user interface mock-ups, which were tested in a design study (Strobl and Katzian 2014) and stronger discussed in a master thesis. Figure 19 presents one of the screens for the evaluation.

For the LNF14 further improvements of the user interface were provided including the outcomes of the design study: To stay platform independent and interoperable, the second HBMS support prototype was realized as a *web application* using the



Um die Menge an Wasser für den Kaffee zu regulieren benutzen Sie den runden silbernen Drehknopf ganz links. Linke Position bedeutet wenig Wasser. Rechte Position: Mehr Wasser.

Figure 18: First UI prototype



Schritt 1

Links oben auf der Kaffeemaschine befindet sich eine weiße Klappe. Klappen Sie diese nach oben und zur Seite.

Figure 19: Mock-Ups of the UI for evaluations

.NET framework and JQuery Mobile. The communication of this support client and the current HBMS core system was done via XML interfaces. Thus, this web-based solution was presentable *on every device with an internet browser*. Media data was already included and there was an acoustic support possibility provided, which started automatically. Nevertheless, the speech output was presented on another device to be able to synchronize it with the texts.

As the HBMS architecture developed over time to a *Model Centred Architecture*, we enhanced our mobile support clients and made them additionally available as *android native apps*. In doing so, mobile functions for audio, vibration and localization enhanced the support quality: it is possible to start speech output automatically based on default setting, vibration feedback is possible, it is possible to zoom into explanations and pictures and it is easier to adopt the graphics accordingly on each android device. In an additional navigation bar, preferences for support could be selected and personal context data is presented in a user friendly way. The prototype was again tested at LNF16.

Moreover, we investigated devices like robots, smart watches and smart TV-devices to be used for user support connected to the HBMS-System via a support client interface and made a proof-of-concept for a support scenario with Amazon Echo as an output device including the language assistant Alexa.

7 Ideas and Future prospects

The HBMS project team can look back on nearly 7 successful years. What started with a *visionary idea* resulted in a *stable beta-version of an active assistance system*.

The impact. Within HBMS, 1 PhD thesis and 6 master theses were completed. 4 PhD theses and 4 master theses are under development. 12 school students were involved in HBMS during their holiday work placement. The HBMS-System and its components and ideas were evaluated with more than 250 people in 4 prototype evaluations, 2 workshops with 60 people, 2 online surveys with more than 540 people, a user design study with 55 people and 2 qualitative studies with more than 100 participants. Since 2014, our modelling approach was taught to more than 200 students at the Next Generation Enterprise Modelling (NEMO) summer school. The project members published more than 25 papers at conferences and articles in journals, held tutorials on Ambient Assistance and Modeling as well as on IT- Based Ambient Assistance and initialized the AHA workshop series (ER-Workshop on Conceptual Modeling for Ambient Assistance and Healthy Ageing).

Nevertheless, the work on the project vision will continue in future:

DSMM and the HCM-L Modeller. Further development of the modelling method is always connected with enhancement of the modelling tool. Aspects, which are important for the DSMM are further investigations of the concept goal (see e. g. goal modelling languages in requirements engineering as iStar (Yu 1997), GoalML (Overbeek et al. 2015) or (Rolland and Salinesi 2005)), the grounding of the HCM-L in a foundational ontology (see e. g. Guizzardi et al. 2008) or taking ideas of multi-perspective modelling (Frank et al. 2017) into consideration. The graphical notation has to be adapted accordingly to additions in the meta-model, e. g. for already done refinements of the context (Michael and Steinberger 2017). Additionally, the HCM-L modeller has to be adapted accordingly to the changes of the meta-model. A new version of the modelling tool should be provided at the OMiLAB website. Other open issues are related with the graphical interface of the modelling tool: a 3D visualisation for BUs and related structural elements would support end users to better understand their preserved behaviour.

Observation. For the next future, further evaluations are planned, to test the observation interface with video based activity recognition systems.

Intelligence. The semantic annotation of web-user interfaces would make it possible to provide meaningful support of their use. Furthermore, the development of the simulator component would make an the automated creation of the simulation interface by using social context information possible. Another missing aspect is the integration of sequences of behaviour models into existing ones in the knowledge base. To finalize this aspect, process mining approaches as e. g., (van der Aalst 2016) might be helpful.

User Interfaces. Further improvements onto the development of active multimodal support for human behaviour include a DSML for multimodal support, the test of further devices e. g., augmented reality approaches, and evaluations of these devices together with end users.

Analysis and Evaluation. An ongoing state-of-the-art analysis and regular evaluations of the technical prototypes are taken for granted. Currently, we are planing the evaluation of various notations for modelling the advanced HBMS context, namely user models, environment models and spatial models.

Moreover, the developed HBMS-System is applicable in several other domains, e. g. *medicine* (Czaplik et al. 2016), *cyber-physical systems* (Berardinelli et al. 2017), *production systems in the area of Industry 4.0* or *IoT applications*. Further investigations in these areas are planned.

Last but not least, such a system is not developed with so much ambition that it remains only in the prototype stage and (as so often in scientific developments) is forgotten after a few years. We hope to be able to bring it to market in the next few years together with a development partner, so that the HBMS system can fulfil its original vision and be *helpful for people*.

Acknowledgments

We would like to thank Heinrich C. Mayr for his visionary ideas, his engagement in raising project funds, his critical feedback and for being our project and team leader as well as scientific mentor.

References

Al Machot F., Haj Mosa A., Ali M., Kyamakya K. (2017) Activity Recognition in Sensor Data Streams for Active and Assisted Living Environments. In: IEEE Transactions on Circuits and Systems for Video Technology

Al Machot F., Mayr H. C., Michael J. (2014) Behavior Modeling and Reasoning for Ambient Support: HCM-L Modeler. In: Proceedings of the International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA-AIE 2014). Lecture Notes in Artificial Intelligence

Al Machot F., Mayr H. C., Ranasinghe S. (2018a) A Hybrid Reasoning Approach for Activity Recognition Based on Answer Set Programming and Dempster-Shafer Theory. In: Recent Advances in Nonlinear Dynamics and Synchronization. Springer, pp. 303–318

Al Machot F., Ranasinghe S., Plattner J., Jnoub N. (2018b) Human Activity Recognition based on Real Life Scenarios. In: Proc. of the IEEE International Conference on Pervasive Computing and Communications, CoMoRea workshop. IEEE

Berardinelli L., Mazak A., Alt O., Wimmer M., Kappel G. (2017) Model-Driven Systems Engineering: Principles and Application in the CPPS Domain In: Multi-Disciplinary Engineering for Cyber-Physical Production Systems: Data Models and Software Solutions for Handling Complex Engineering Projects Biffl S., Lüder A., Gerhard D. (eds.) Springer International Publishing, pp. 261–299 https://doi.org/10.1007/978-3-319-56345-9_11

Bonfiglio S., Bekiaris E., Panou M., Sala P., Bautista J., Agües M., Robledo M. G., Van Isacker K., Cabrera F., Mixco V. J., de la Maza C., Staehli B. (2008) Use Cases and application scenarios for independent living applications

Cook D. J., Crandall A. S., Thomas B. L., Krishnan N. C. (2013) CASAS: A smart home in a box. In: Computer 46 (7)

Czaplik M., Nazari P. M. S., Roth A., Rumpe B., Voigt V., von Wenckstern M., Wortmann A. (2016) Der Weg zur Modellbasierten Evolution und Adaption medizinischer Leitlinien. In: Software Engineering Workshops 2016. CEUR-ws.org, pp. 195–200

Fill H.-G., Karagiannis D. (2013) On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform. In: Enterprise Modelling and Information Systems Architectures Vol. 8, pp. 4–25 http://eprints.cs.univie.ac.at/3657/1/Fill_Karagiannis_EMISA_2013.pdf

Fliedl G., Gratzer W., Strobl T., Winkler C. (2013) Mobile Instruction Apps Based on Linguistic Text Reduction. In: Rault J.-C. (ed.) Proceedings of ICSSEA 2013

Frank U. (2013) Domain-Specific Modeling Languages: Requirements Analysis and Design Guidelines. In: Reinhartz-Berger I., Sturm A., Clark T., Cohen S., Bettin J. (eds.) Domain Engineering. Springer Berlin Heidelberg, Berlin and Heidelberg, pp. 133–157

Frank U., Reinhartz-Berger I., Sturm A., Clark T. (2017) A Multi-level Approach for Supporting Configurations: A New Perspective on Software Product Line Engineering. In: Cabanillas C., España S., Farshidi S. (eds.) Proc. of the ER Forum 2017 and the ER 2017 Demo Track co-located with the 36th International Conference on Conceptual Modelling (ER 2017), pp. 170–178

Guha R., Brickley D., Macbeth S. (2016) Schema.org: Evolution of structured data on the web. In: Communications of the ACM 59 (2), pp. 44–51

Guizzardi G., Falbo R., Guizzardi R. S. (2008) Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The case of the ODE Software Process Ontology. In: Lencastre M., Falcão e Cunha, João: Valecillo, Antonio (eds.) 11th Iberoamerican Workshop on Requirements Engineering and Software Environments, pp. 127–140

Karagiannis D., Grossmann W., Höfferer P. (2007) Open Model Initiative: A Feasibility Study. www.openmodels.at

Karagiannis D., Kühn H. (2002) Metamodelling Platforms. In: Bauknecht K., Tjoa A. M., Quirchmayr G. (eds.) E-Commerce and Web Technologies. LNCS Vol. 2455. Springer, p. 182

Kaschek R., Mayr H. C. (1996) A characterization of OOA tools. In: IEEE International Symposium on Assessment of Software Tools, pp. 59–67

Katz S. (1983) Assessing self-maintenance: Activities of daily living, mobility, and instrumental activities of daily living. American Geriatrics Society, New York NY

Kofod-Petersen A., Cassens J. (2006) Using Activity Theory to Model Context Awareness. In: Roth-Berghofer T., Schulz S., Leake D. (eds.) Modeling and Retrieval of Context. Lecture Notes in Computer Science Vol. 3946. Springer, pp. 1–17 http://dx.doi.org/10.1007/11740674_1

Kop C., Mayr H. C. (1998) Conceptual predesign bridging the gap between requirements and conceptual design. In: 1998. Proceedings. 1998 Third International Conference on Requirements Engineering, pp. 90–98 http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=667813

Lawton M. P., Brody E. M. (1969) Assessment of older people: self-maintaining and instrumental activities of daily living. In: The Gerontologist 9(3), pp. 179–186

Leont'ev A. N. (1978) Activity, Consciousness, and Personality. Prentice-Hall, Englewood Cliffs, NJ

Lindland O., Sindre G., Solvberg A. (1994) Understanding quality in conceptual modeling. In: IEEE Software 11(2), pp. 42–49

Mayr H. C., Al Machot F., Michael J., Morak G., Ranasinghe S., Shekhovtsov V., Steinberger C. (2016) HCM-L: Domain-Specific Modeling for Active and Assisted Living. In: Karagiannis D., Mayr H. C., Mylopoulos J. (eds.) Domain-specific conceptual modeling. Springer, pp. 527–552 http://link.springer.com/chapter/10.1007/978-3-319-39417-6_24

Mayr H. C., Michael J. (2012) Control pattern based analysis of HCM-L, a language for cognitive modeling. In: International Conference on Advances in ICT for Emerging Regions (ICTer2012). IEEE, pp. 169–175

Mayr H. C., Michael J., Ranasinghe S., Shekhovtsov V. A., Steinberger C. (2017) Model Centered Architecture In: Conceptual Modeling Perspectives Cabot J., Gómez C., Pastor O., Sancho M. R., Teniente E. (eds.) Springer International Publishing, pp. 85–104 https://doi.org/10.1007/978-3-319-67271-7_7

Mayring P. (2010) Qualitative Inhaltsanalyse. In: Mey G., Mruck K. (eds.) Handbuch Qualitative Forschung in der Psychologie. VS Verlag für Sozialwissenschaften, Wiesbaden, pp. 601–613

Michael J., Al Machot F., Mayr H. C. (2014) A Behavior Centered Modeling Tool Based on ADOxx. In: Proceedings of the CAiSE'14 Forum. CEUR

Michael J., Al Machot F., Mayr H. C. (2015) ADOxx based Tool Support for a Behavior Centered Modeling Approach. In: Proceedings of the 8th PErvasive Technologies Related to Assistive Environments (PETRA) conference. ACM

Michael J., Grießer A., Strobl T., Mayr H. C. (2013) Cognitive Modeling and Support for Ambient Assistance. In: Mayr H. C., Kop C., Liddle S., Ginige A. (eds.) Information Systems: Methods, Models, and Applications: Revised Selected Papers.. LNBIP Vol. 137. Springer, Heidelberg

Michael J., Mayr H. C. (2013) Conceptual Modeling for Ambient Assistance. In: Ng W., Storey V. C., Trujillo J. (eds.) Conceptual Modeling - ER 2013. Lecture Notes in Computer Science (LNCS) Vol. 8217. Springer, pp. 403–413

Michael J., Mayr H. C. (2015) Creating a Domain Specific Modelling Method for Ambient Assistance. In: International Conference on Advances in ICT for Emerging Regions (ICTer2015). IEEE, pp. 119–124

Michael J., Mayr H. C. (2017) Intuitive understanding of a modeling language. In: Proceedings of the Australasian Computer Science Week Multiconference (ACSW'17). ACM, New York NY, USA, pp. 1–10

Michael J., Steinberger C. (2017) Context Modeling for Active Assistance. In: Cabanillas C., España S., Farshidi S. (eds.) Proc. of the ER Forum 2017 and the ER 2017 Demo Track colocated with the 36th International Conference on Conceptual Modelling (ER 2017), pp. 221–234

Moody D. (2009) The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. In: IEEE Transactions on Software Engineering 35(6), pp. 756–779

Object Management Group OMG: (n.d.) Meta Object Facility (MOF) Core. last accessed 2018-01-12 http://www.omg.org/spec/MOF/

Oldevik J., Neple T., Grønmo R., Aagedal J., Berre A.-J. (2005) Toward Standardised Model to Text Transformations. In: Hartman A., Kreische D. (eds.) Model Driven Architecture – Foundations and Applications. LNCS Vol. 3748. Springer Berlin, Heidelberg, pp. 239–253

Overbeek S., Frank U., Köhling C. (2015) A language for multi-perspective goal modelling: Challenges, requirements and solutions. In: Computer Standards & Interfaces 38, pp. 1–16

Ranasinghe S., Al Machot F., Mayr H. C. (2016) A review on applications of activity recognition systems with regard to performance and evaluation. In: International Journal of Distributed Sensor Networks 12(8)

Rolland C., Salinesi C. (2005) Modeling Goals and Reasoning with Them. In: Aurum A., Wohlin C. (eds.) Engineering and Managing Software Requirements. Springer-Verlag, pp. 189–217

Schütte R. (1998) Vergleich alternativer Ansätze zur Bewertung der Informationsmodellqualität. In: Fachtagung: Modellierung betrieblicher Informationssysteme, Fachgruppe MobIS

Shekhovtsov V. A., Mayr H. C., Ranasinghe S. (2018) Model-Based Interfacing to Activity Recognition

Steinberger C., Michael J. (2018) Towards Cognitive Assisted Living 3.0. In: Advanced Technologies for Smarter Assisted Living solutions: Towards an open Smart Home infrastructure (SmarterAAL workshop) at Percom 2018. IEEE, pp. 1–6

Strobl T., Katzian A. (2014) Darstellungsvarianten für handlungs-unterstützende Apps: Ergebnisse einer Evaluationsstudie im Rahmen des Projekts HBMS. In: Wohnen-Pflege-Teilhabe – 7. Deutscher AAL-Kongress. VDE

Strube G. (1996) Wörterbuch der Kognitionswissenschaft. Klett-Cotta, Stuttgart

Tulving E. (1972) Episodic and semantic memory. In: Tulving E., Donaldson W., Bower G. H. (eds.) Organization of memory. Academic Press, New York, pp. 381–403

van der Aalst W. M. P. (2016) Process Mining: Data Science in Action, 2nd ed. Springer https://doi.org/10.1007/978-3-662-49851-4

van der Aalst W., ter Hofstede A., Kiepuszewski B., Barros A. (2003) Workflow Patterns. In: Distributed and Parallel Databases 14(1), pp. 5–51

Wöckl B., Yildizoglu U., Buber-Ennser I., Aparicio Diaz B., Tscheligi M. (2011) Elderly Personas: A Design Tool for AAL Projects focusing on Gender, Age and Regional Differences. In: Partnerships for Social Innovations in Europe

Wohed P., van der Aalst W. M., Dumas M., ter Hofstede A. H., Russell N. (2005) Pattern-based Analysis of BPMN. In: http://eprints.qut.edu.au/2977/

Yu E. S. K. (1997) Towards modelling and reasoning support for early-phase requirements engineering. In: Proceedings of the Third IEEE International Symposium on Requirements Engineering, pp. 226–235