Challenges in an Assistance World

Veit Köppen
University of Magdeburg
39106 Magdeburg, Germany
veit.koeppen@ovgu.de

Thomas Kirste
Rostock University
18051 Rostock, Germany
thomas.kirste@uni-rostock.de

Gunter Saake
University of Magdeburg
39106 Magdeburg, Germany
gunter.saake@ovgu.de

Abstract

The development of small and embedded devices, smart interaction techniques, and integration in an ubiquitous data access environment enable the support of humans in daily life. A systematic engineering of situation-aware assistance systems needs methods, models, and techniques that use knowledge on human actions. In this paper, we address challenges of engineering assistance systems: Models of structures of actions, reactions on situations, systems' reliability due to uncertainty of reality and human actions, and architecture of assistance systems network. A holistic engineering concept that maps reality to a world model is not in scope of these challenges. However, best practices and patterns for the development of models and systems can enhance engineering assistance systems.

1 Introduction

In an ubiquitous environment a human is supported via a network of inter-operating smart devices. The smart kitchen, see for instance [Beetz *et al.*, 2008], and smart homes, *e.g.*, for the support of persons with dementia [Serna *et al.*, 2007], are already available. However, the engineering of these environments is only at the beginning: for each application scenario the systems are developed from scratch. Assistive systems need knowledge on structures of human actions. This is due to the fact that these systems have to support actions on basis of sensor. An important property of situation-aware assistive systems is the availability of explicit machine processable knowledge on structures on actions.

Today, assistive systems are developed for specific applications. A tailor-made system uses domain knowledge on users activities in a hard-wired way. Therefore, the application cannot be reused for other application scenarios. This makes the engineering process of such systems inefficient. In software engineering methods, such as patterns, have been used to increase engineering efficiency, cf. [Gamma *et al.*, 1995] for object-oriented design patterns.

A model is always a depiction of the real world, where information has to be filtered to focus on the relevant details. This reduction of complexity is also necessary for the development of situation-ware assistive systems. In object-oriented software development UML models can be used. There exist 13 different diagram types to create models that represent a view on the system context. For integrating information of human-machine interaction as well as human behavior modeling additional modeling concepts are necessary. Another important aspect is the handling of uncertainty which comes from measuring the environment and the identification of human actions.

We see the need for an integrated view on these addressed models as well as a developer's support in engineering assistive systems. We reduce complexity where at the same time an integrated model is available via the usage of ontologies. Another support for developers can be achieved if all elements of th models are described and related in a meta level. Therefore, we restrict ourselves in this paper to this meta level.

As one outcome we target at the re-usage of existing models from different domains such as software engineering, decision support, and human behavior modeling. The integration of the interdisciplinary domains is required to develop effective assistive systems. Furthermore, we see the need for research on refactoring existing models (from different applications) and an model abstraction to provide developers with a pattern approach, see for this requirement [Hein *et al.*, 2009].

One branch ubiquitous computing research looks at supporting a user in performing activities by proactive assistance: by building smart environments that are aware of their inhabitants, and react to their behavior without explicit interaction, such as Smart Homes [Brumitt *et al.*, 2000], Smart Health Care environments [Agarwal *et al.*, 2007], Smart Laboratories [Thurow *et al.*, 2004], or Smart Rooms [Heider and Kirste, 2002], or by building mobile assistants that support persons in performing mobile activities, such as helping persons with mild cognitive impairment following a travel itinerary [Patterson *et al.*, 2004] or supporting elder care nurses providing out-patient care [Umblia *et al.*, 2009].

In this paper we present three important issues for engineering and usage of assistance environments:

- Does a consistent and integrated model for assistive systems exist? Can we create a modularized structure for all model components?
- Which structural requirements have the models to fulfill?
- Does a concept exist that can execute all involved system parts and models?

In the following, we assume that a generic framework for developing assistive systems is feasible, We restrict ourselves to systems where explicit knowledge on states and structures is available. For the decision of the assistance function we assume that cost and use are computable and the cognitive state of a human can be represented via variables which include non-rational behavior.

2 Conceptual Framework

On the one side, we present a conceptual framework for assistive systems in this section. On the other side, we state a proof of fundamental combinability of models.

2.1 Decision Theory for Assistance

We use the conceptual framework of decision theory to provide some tangibility to discussion of the relevant modeling mechanisms. For uncertain reasoning we use probability theory[1]. We assume that an assistive system has some alphabet of actions A from which to choose and there is a set of *situations*, X. In addition, there is a *reward* $r(a, x)$ that, for some $a \in A$ and $x \in X$ gives the "utility" of performing a, for simplicity we assume $r(a, x) \in \mathbb{R}$.

Usually, an assistive system has no direct knowledge about the true state of affairs (that is, the $x \in X$ that currently holds in reality). Instead, it has to infer the possible state from a sequence of observations $y_{1:t} = \langle y_1, \ldots, y_t \rangle$, where the individual observations y_i are elements of some observation space Y.

An inference mechanism has to be provided that assigns some numbers $p(x \mid y_{1:t}) \in \mathbb{R}$ to a potential state $x \in X$ based on an observation sequence $y_{1:t}$. To simplify the following discussion, we assume those numbers to be just probabilities.

Once such an inference mechanism is available, we can use it to select the best action given the current situation by computing that action which maximizes the expected reward. The latter is computed with respect to the density $p(x \mid y_{1:t})$, so that we get:

$$\begin{aligned} a_{opt}(y_{1:t}) &= \arg\max_{a \in A} E\left[r(a, x) \mid y_{1:t}\right] \\ &= \arg\max_{a \in A} \int_X r(a, x) p(x \mid y_{1:t})\, dx \end{aligned}$$

It proves beneficial to define the reward of performing action a in state x based on the utility of the state $x' = a(x)$ reached by this action. The reason is that *devices* know about things they can do for a person, but a *person* knows what state is helpful for her. Let us assume there is a function $u(x) \in \mathbb{R}$ defining the utility of a state. We can then simply define $r(a, x) := u(a(x)) + c(a, x)$, where $c(a, x)$ are the costs for performing a in state x. For the further discussion we do not consider c and just focus on u.

For computing the quantities $p(x \mid y_{1:t})$, we need additional information about how states relate to observations (*i.e.*, how states *cause* observations) and how states at some time t can cause other states at time $t+1$. This information is usually captured by the *system model* and the *observation model*.

- The observation model $p(y \mid x)$ gives the probability that an observation y occurs when the state is x.
- The system model $p(x' \mid x)$ gives the probability that at the next time the state is x' if the current state is x.

In addition, we need some information on the probability of the initial state, given by the *prior density* $p(x_1)$. One can then employ the machinery of Bayesian filtering for computing the new state density $p(x_{t+1} \mid y_{1:t+1})$ from the current state density $p(x_t \mid y_{1:t})$ and a new observation y_{t+1}.

[1] There are of course other approaches to uncertain reasoning. However, our point is not the specific inference strategies, but rather the ontological structure of the *modeling entities* that can be given to such reasoning strategies.

This probabilistic reasoning concept is quite well established in the area of activity recognition and ambient assisted living; many projects use some kind of Bayesian filtering for estimating states or activities, usually employing some variants of hidden Markov models or dynamic Bayesian networks [Patterson *et al.*, 2004; Gottfried and Aghajan, 2009]. Markov decision processes are commonly used in research as well as practice. For an overview see [Puterman, 1994; Bertsekas, 2005]. In the following we use partially observable Markov decision processes, see for an introduction [Monahan, 1982; Etzioni, 1991]. Even full decision theoretic approaches using partially observable Markov decision processes have been discussed [Boger *et al.*, 2006].

However, from the viewpoint of systematic engineering of assistive systems, the "processor" per se is not the real question – it is rather the *programming language* that can be used to get this processor to work.

With respect to the above framework, we are asking for a language that we can use to model the following information:

- The state space X
- The observation space Y
- The action alphabet A
- The utility function $u(x)$, for $x \in X$
- The prior probability $p(x_1)$, for $x_1 \in X$
- The observation model $p(y \mid x)$, for $y \in Y, x \in X$
- The system model $p(x' \mid x)$, for $x', x \in X$

A model M can thus be formally considered a seven-tuple $M = \left(X, Y, A, u, p_X, p_{Y|X}, p_{X'|X}\right)$, where p_X, $p_{Y|X}$, and $p_{X'|X}$ denote the prior, the observation model, and the system model, respectively.

Application areas for assistive systems are for instance helping persons make efficient use of a complex instrumented environment – this may be a conference room with multitudes of presentation support devices, as well as a contemporary private home with all its entertainment, heating, ventilation, air condition, and lighting systems. And if persons with cognitive impairments are target users, even "normal" environments suddenly appear to be complex.

If the application domains are restricted well enough – as it is the typical practice in current research – the state space becomes so small (some dozen bits) that it can be handled by some Markov model, which is a well understood modeling paradigm. This means that at the scale of current research the real problem is simply not yet visible.

But as soon as one leaves a well confined application scenario, just a few hundred bits of state suffice[2], one arrives at complexities that can not any longer be handled by paradigms that rely on an explicit enumeration of states.

Clearly, using modern sampling based methods one can perform inference on such models – but how do we get them into the system into the first place? As very first step, we need a better understanding of the ontological structure of the world such assistive systems have to live in: It is this ontology that wen need to render comprehensible to the assistive system. We present some ideas for an integrated model approach in Section 2.2.

[2] Just remember: $2^{300} >$ number of elementary particles in the universe.

2.2 Assistive Systems need an integrated Model Approach

The State Space

The state space X consists of at least three subdomains of interest:

- $\mathcal{X}_D$ Internal states of devices (for instance, the state of the media crossbar)

  An assistive system needs to know the device state at least for two reasons: for getting the device to change to a desired state, and for explaining to a user in what state a device is and what to do about this.

- $\mathcal{X}_P$ Internal states of persons (for instance, the objective of the speaker to give a presentation)

  Of course, assistive systems are about helping people. This means, the assistive system has to know what a person currently wants and needs. Specifically the first point – what a person wants – essentially refers to internal cognitive states. In this case to the intention of a person.

- $\mathcal{X}_E$ Environment states (for instance, the position of persons and devices)

  The environment is the world where actions of devices and persons meet and where observations have their origin. An assistive system can help a person only by changing the environment through a device action in such a way that the change in environment supports the person in achieving his intention.

  Likewise, only via the observation of effects in the environment (*e.g.*, a change in position) do a person's intentions become indirectly visible to the system. Intentions that do not lead to observable changes in behavior effectively can not be inferred.

For simplicity, we assume a state $x \in X$ to be a (partial) function $x : N(X) \to V(X)$ that maps state variables $n \in N(X)$ to some values $v \in V(X)$. The set $N(X)$ can be considered the *names* of the *state variables* of the state space X, or the *ontology* of X. For different state spaces X and Z one can then ask whether $N(X) \cap N(Z) \neq \emptyset$ – that is, whether their ontologies overlap. We call $N(X) \cap N(Z)$ the *overlap ontology*.

Combining Models

The state space is directly connected to the system model, which can be regarded as a non-deterministic function[3] mapping states to states.

For simple devices, the system model can easily be described by some kind of – possibly nondeterministic – finite state machine. Such a finite state machine can readily be converted to a corresponding probabilistic model, basically by using the adjacency matrix of the machine's transition graph as transition matrix of a Markov model.

On the other hand, in various experiments human behavior has been modeled by (hierarchical) Markov models, by production system rules [Serna *et al.*, 2007], by hierarchical plans [Roy *et al.*, 2007; Naeem and Bigham, 2008], by process calculi [Burghardt *et al.*, 2011], etc. For a comprehensive overview of different behavior modeling approaches see [Yordanova, 2011]. These models are quite different in syntactic structure, but can basically always be translated into labeled transition systems (LTS) on some suitable state space.

In order to build assistive systems in a methodical and modular way, it would be helpful if individual models for devices, for human behaviors (resp. behavioral components), and for other dynamic aspects of the world could be modeled independently of each other. For instance, when modeling a smart meeting room one would like to create the model describing the functionality of a projector device D independently of the (presumable) behavior of a person P giving a presentation at a meeting. There would be two model seven-tuples: M_D and M_P. The action alphabets of both models, A_D and A_P, would encode the set of actions available for an assistive system having access to objects of type M_D and M_P; presumably A_P would be empty, encoding the fact that an assistive system can not use a person as agent for delivering assistive actions[4].

The question is now of course, how to *combine* both models M_D and M_P into a joint model $M_{D\cup P}$ that gives the assistive system the required information to decide by which of the projector's actions to support the presenter in what situation.

How this could be done is not immediately obvious: Let $p_{X'_D|X_D}$ and $p_{X'_P|X_P}$ be the system models of projector and presenter. Surely, in order to make this an interesting situation (with respect to model combination), we assume than $N(X_D) \cap N(X_P) \neq \emptyset$. Otherwise the state spaces of projector and presenter would be defined on disjoint ontologies – and there would be no way for them to influence each other, they'd live in separate worlds. (A side remark: one should assume that $N(X_D) \cap N(X_P) \subseteq N(\mathcal{X}_E)$.) For instance, one would like a variable such as *information i projected on screen s* to be an element of the overlap ontology.

This raises two challenges: the first one is to identify mechanisms that ensure that a name n where $n \in N(X_D) \cap N(X_P)$ means the same thing to both worlds (that is: for both designers of M_D and M_P, the variable n represents the same concept). The second one is the computation of the joint system model from the component models. Here we briefly outline that at least formally a consistent concept of model combination can be derived. This combination strategy is not intended to be *practical*; its objective is to show that it is *possible* to provide sound combination of assistance models, thereby justifying research on how this can be done practically.

Consider two states $x_D \in X_D$ and $x_P \in X_P$. If for all variables $n \in N(X_D) \cap N(X_P)$ we have that $x_D(n) = x_P(n)$, we can compute a consistent joined state simply by defining $x_{D\cup P} = x_D \cup x_P$, which then gives us $X_{D\cup P}$. This is just the standard database join. Now consider two states $x', x \in X_{D\cup P}$. How do we define $p(x' \mid x)$ from the given models p_{X_D} and p_{X_P}?

Clearly, since both x' and x are results from state joins, there must be states $x'_D, x_D \in X_D$ and $x'_P, x'_P \in X_P$ such that $x' = x'_D \cup x'_P$ and $x = x_D \cup x_P$. We now ask for the relation between the required value $p(x' \mid x) = p(x'_D \cup x'_P \mid x_D \cup x_P)$ and the given values $p(x'_D \mid x_D)$ and $p(x'_P \mid x_P)$. This is in general *not* given by

$$p(x'_D \cup x'_P \mid x_D \cup x_P) = p(x'_D \mid x_D)\, p(x'_P \mid x_P),$$

since x'_D and x'_P are clearly not independent. However, by

[3] To be taken with a pinch of salt. From a mathematical viewpoint, a non-deterministic function is an oxymoron; requiring X to also contain the current world time solves this.

[4] Although this could be matter of debate.

defining $x'_{P\setminus D} := x'_P \setminus x'_D$ the laws of probability give us

$$p(x'_D \cup x'_P \mid x_D \cup x_P) = p(x'_D, x'_{P\setminus D} \mid x_D \cup x_P)$$

using the fact that $p(a,b) = p(a \mid b)\, p(a)$

$$= p(x'_D \mid x'_{P\setminus D}, x_D \cup x_P) p(x'_{P\setminus D} \mid x_D \cup x_P)$$

assuming state variables in both models are *independent* from variables of the respective other model outside the overlap ontology, gives

$$= p(x'_D \mid x_D) p(x'_{P\setminus D} \mid x_P)$$
$$= p(x'_D \mid x_D) \int_{D\cap P} p(x'_P \mid x_P) dx'_{D\cap P}.$$

By the last line we see that only information already provided by the component models is required. This result clearly shows that a combination of models at least *formally* is well defined. The challenge is that computing the integral[5] can be rather expensive once state spaces become interesting. Once we consider state variables with continuous value domains – *e.g.*, spatial position and attitude of a person's head in 6D coordinates – with analytical representations of the transition function (*e.g.*, by a simple linear transform), and try to combine this with discrete value domains and LTS-based transition functions, a unified handling becomes a significant challenge.

For combining observations, we just reuse the above derivation:

$$\begin{aligned} & p(y_{D\cup P} \mid x_{D\cup P}) \\ &= p(y_D \cup y_P \mid x_D \cup x_P) \\ &= p(y_D, y_{P\setminus D} \mid x_D \cup x_P) \\ &= p(y_D \mid y_{P\setminus D}, x_D \cup x_P) p(y_{P\setminus D} \mid x_D \cup x_P) \\ &= p(y_D \mid x_D) p(y_{P\setminus D} \mid x_P) \\ &= p(y_D \mid x_D) \int_{D\cap P} p(y_P \mid x_P) dy_{D\cap P}. \end{aligned}$$

The combination of the utility functions is the conceptually least challenging task. For each entity E – device or person, the entity's model M_E provides a utility function $u_E : X_E \to \mathbb{R}$. Now, in order to compute the value of u_E in the joined state X, we simply extract those state variables that belong to X_E and compute u_E based on this restricted state.

For multiple entities, the system requires a function for combining the different individual utility values to a global value; the system's objective then is to maximize this global value.

[5]The integral above is a short hand notation for the following idea: the objective is to marginalize $p(x'_P \mid x_P)$ over all variables $n \in N(X_P) \cap N(X_D)$, the density resulting from this marginalization is then just $p(x'_{P\setminus D} \mid x_P)$. If we define

$$I(x_P, D) := \{\, x \in X_P \mid \forall n \in N_{P\setminus D} : x(n) = x_p(n) \,\}$$

where $N_{P\setminus D} := N(X_P) \setminus N(X_D)$, we can write this integral with more rigor as

$$\int_{I(x'_P, D)} p(x \mid x_P) dx.$$

In a first instance, the global utility may be a weighted sum of the individual utilities, possibly subject to a nonlinear transform. Finding a good utility combination function thus is no conceptual problem, but rather a question of providing a good understanding of the application domain to which these utility functions pertain.

3 Refined Investigation of Model Requirements

In this section, we address challenges which developers of assistive systems have to cope with for creating new methods and techniques, tools and models.

Modeling of human actions, activities, and processes requires on the one side, that a technical system can process the models in such a way that sensor data is used to identify and classify a situation. On the other side a priori knowledge is required that can be interpreted by an assistive system to derive causal relations of human actions. Models of Activities should not only be used for detection of actions but also incidents of causal relations. At this point an assistive system can can target-oriented support the user.

An assistive system offers the user in a given situation possible alternatives or a support in his actions. Therefore, it is necessary to specify situations or activities and their corresponding assistance functions. Another question is in which context and form the assistance is carried out. Furthermore, a user sometimes wants to disable support or requests additional information. The interaction of human and machine is ambient environment a complicate task.

Assistive systems have to work in an imperfect world: sensor data is not error-free and often ambiguous. Models of activities can be imprecise and misleading. In the case a user counts on the correct support this can have a dramatical outcome in the case of fault. Analyses of reliability and Safety have to be done already in the engineering phase. A system should be constructed (and validated) in such a way that a minimization of the worst case scenario is applied.

An assistive system consists of hardware and software. An system architect has to answer questions on the involved components, interface between components, and model integration. In software engineering, integrated development environments (IDE) are used to create a holistic view on the system in the development phase. Analyses and implementation are supported at the same time. For the development of assistive systems a framework could enhance the development process. Further questions arise in the context of cooperation of assistive systems, the integration of these as well as the evolution of such systems.

An assistive system supports a human in daily life. It consists of sensors that measure the environment, models that represent information on the context as well as assistance functionalities, an inference mechanism to deduct objects, activities, and support information. Although sensors measure the environment the current state of the human is not observable. Furthermore, information in the assistive system are also not directly accessible for the user. Therefore, there exist three worlds, as shown in Fig. 1.

In Section 2.2 we already describe the state space. This we map in Fig. 1. In *World I* the cognitive states of the persons are represented. Theses states are hidden for the system and cannot be directly assessed in an assistive scenario. We also can interpret these states as believes of the human. From the conceptual frameworks World I is represented as $\mathcal{X}_P$. *World II* represents the device states (cf.

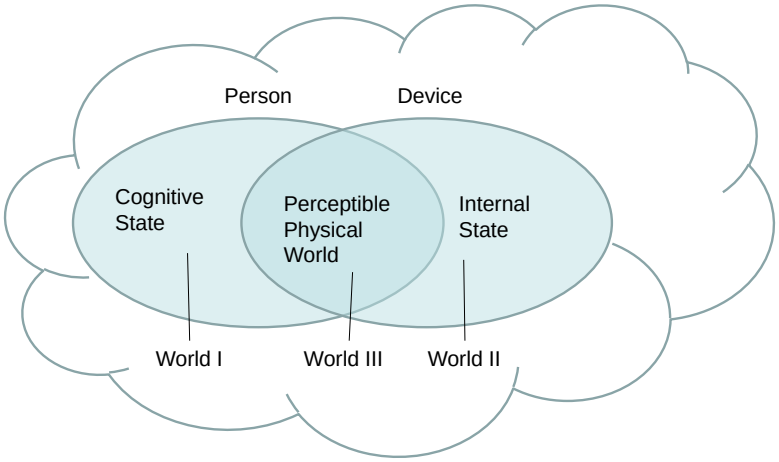

Figure 1: Three Worlds in Assistance Support

$\mathcal{X}_D$). Due to the fact that normally the engineer is different than the supported user the device systems states are also hidden and the user cannot directly access them. The jointly accessible physical states are included in *World III*. The conjoint environment state is depicted by $\mathcal{X}_E$. Here actions and observations are made by persons and devices. Note, due to measurements of sensors and perception of humans a different outcome of person and device is possible. However, we assume that these outcome differences are constituted with the respective hidden states of person and device. For details on intentions and assistance refer to [Kirste, 2011].

In the following sections, we depict some of these challenges with the focus on an abstract level of assistance despite domain or application specific developments as for instance in [Kautz *et al.*, 2002].

3.1 Object Identification from Sensor Data

The identification of objects is investigated already a long time. Besides technical development of sensors and embedded systems, new methods are used and developed to improve the identification and tracking of objects. However, an object description is required to recognize an object. Although a human can recognize a multitude of objects with small effort, in computer vision the identification of objects is a challenging task. Besides the definition of suitable object properties, different sensor types such as camera, touch, or laser, sonar, or RFID can be used. A meta model is required to map these different measurements into the properties such as shape, id, or volume information.

Sensor fusion, see for instance [Nagla *et al.*, 2010], can be used to improve results in dynamic environments. This enhances not only the use of sensors in complex and complicate environments but also identifies an object with sonar sensors via its shapes.

However, two problems have always to be addressed: On the one side an object has to be known to the identifying system. This includes information on the object's properties as well as meta information describing the properties in a machine processable description. On the other side an inference mechanism is required to map the sensor data to objects. Typical representatives of inference mechanism are Bayesian or neural networks.

A detection of a set of human activities is presented in [Sung *et al.*, 2011]. The authors use a RGBD image sensor to obtain data that is mapped into a specified set of activities. Depending upon a learning phase the hierarchical maximum entropy Markov model can identify approximately 80 percent of activities. Note, the overall set of activities consists of only twelve activities and without a learning phase the identification is heavily reduced.

Different sensors can be used to measure the environment and to identify or track objects or persons. There exist different applications the use sensors and models to assist persons in daily life. In [Hub *et al.*, 2003] the authors use a sensor system to support in orientation blind person in an indoor scenario. The user interface consists of keys and a text-to-speech system. [Baum and Edwards, 1993] develop the kitchen task assessment to measure cognitive performance in senile dementia of the Alzheimer's type. The functional measurement records the level of required support for simple tasks in cooking applications. [Serna *et al.*, 2007] describe smart homes in the same context. they use ACT-R models for human activities in daily life, derive potential errors, and obtain information on required support.

3.2 From Sensor Data to Activity

An observation is a mapping from real or virtual sensor data with help of an object model. We assume a database centric model for the object model. Therefore, we present the object model in this section. Furthermore, sensor data underly a measurement error. This has to be taken into account within the transformation. We use probabilistic methods for inference making and describe the possible observation space.

An object schema $\mathcal{OS}$ describes in the context of databases structures, object types, predicates, relations, and object properties. $\mathcal{OS}$ can be seen as static ($\mathcal{OS}_{static}$) and dynamic ($\mathcal{OS}_{dynamic}$) components. $\mathcal{OS}_{static}$ depicts the knowledge of static world elements. Sensor data that are received by a system are measurements and have to be mapped as deterministic elements into $\mathcal{OS}_{static}$. Note, $\mathcal{OS}_{static}$ is fixed for a domain. The dynamic part consists of sensor data and the corresponding belief: $\mathcal{OS}_{dynamic} =$ Sensor-Data$+Belief$. Sensor data are fixed at a given time t. However, sensor data are measurements and therefore, measurement errors have to be taken into account. This has to be considered also at the modeling level. Furthermore, $Belief$ is dependent upon time t and the related state space.

We assume that a model is always a view on reality and

there exist several views at the same time. Due to the character of abstraction and selection by creation of models, different models result although they represent the same reality. For interconnection of these different views we state, that there exists different worlds. A second requirement for introducing different world modes results from the probabilistic character. Two levels exist that introduce possible world states. On the one hand measurement errors (due to sensors) introduce possible mappings. On the other hand, inference making uses probability theory. Both aspects are included by the fact that each world representative w is accompanied by a probability p. A state s represents a world view at a time point. Again a state is observed via sensors and therefore a possibility function has to be added. We call this possible state $\langle s, p_s \rangle$.

- static knowledge is fixed in a database $w_{fixed} \in [\![\mathcal{OS}_{static}]\!]$
- possible situations , possible states
  semantics of $\mathcal{OS}$ written as $[\![\mathcal{OS}]\!]$ is a set $\mathcal{W}$ of worlds
- a situation $s \in \mathcal{S}$ is a model of $\langle \mathcal{OS}, \mathcal{C} \rangle$ (written as $s \in [\![\langle \mathcal{OS}, \mathcal{C} \rangle]\!]$) iff
  - it is a model of $\mathcal{OS}$
    $$s \in [\![\mathcal{OS}]\!]$$
  - it contains the static knowledge
    $$s \downarrow_{\mathcal{OS}_{static}} = s_{fixed}$$
  - it satisfies the constraints
    $$s \models \mathcal{C}$$

A situation s is a state of an activity. There exists a belief B that adds a probability space to all possible situations. We simplify to a discrete probability space although continuous is more realistic. However, this simplification enables us to explore our model in a manageable way and transfer it to Markov decision processes which are used for decision making, see Section 2.1.

We use an abstract view on transformation of states. These can be seen at different levels of granularity, from atomic actions to complex processes. An activity is a sequence ordered multi-set of atomic actions that describes a transformation from one state of a world to another. Note, that some transitions can yield to the same state again. We assume that an activity is an observable sequence and can be inferred via the current set of observations and information on the current state. An action is an elementary (atomic) state transformer. In reality, hierarchies exists where actions form an activity and a set of activities describe one process.

The identification of atomic actions as well the abstract typical actions is a challenging task which can only be solved if the complete context is available. An atomic action is a not further segmented for a given monitoring environment. An abstract typical action can be used in all applications or domains. An example are basic action in computer adventure games such as put, take, talk, and walk. However, in monitoring an environment via sensors it is possible that not all required information is processable. Reasons might be incorrect data, unavailable sensors, occlusions, or belief that lead to wrong information. Therefore, we have to handle these in a suitable way, *e.g.*, including uncertainty information.

4 Architectural Framework

In this section we present the architectural framework for engineering assistive systems. This includes the support of an integrated model approach as well as the concepts and requirements that have to be applied for developing these systems.

4.1 Engineering Assistance Systems

An architecture for assistive systems consists of several components. We show a possible architecture in Fig. 2. An assistive systems needs to interact with persons. Therefore, a component that is for instance responsible for graphic and touch is required. Furthermore, sensors deliver measurements of persons and their environment. these measurements have to be mapped to objects that are known to the assistive system (in the object model). The hidden cognitive state of the supported persons is stored in cognitive models, whereas the description on assistive situations and activities is stored in process models. The assistance is based on utility functions and the interaction of cognitive states and current process state. These information is modeled in assistance model component. There exists different data flows between these components such as from object model and cognitive model, where the current cognitive state is derived from the identified objects.

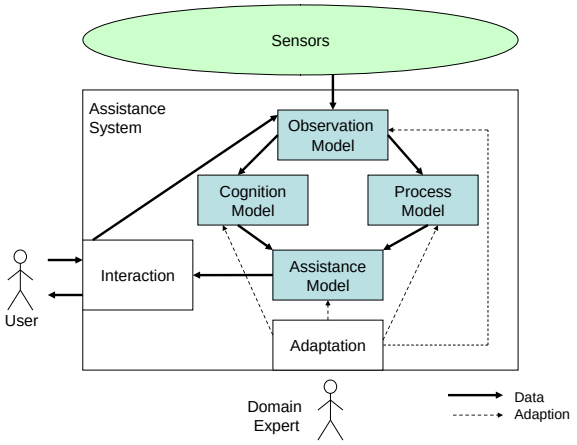

Figure 2: Architecture of Assistance System

An assistive system has to react on changing environments, persons, and new processes. Therefore an adaptive component has to be included for example to change models, enter new objects, or adapt the process model.

In the case of assistive systems in the daily use, we have some requirements for the models used in assistive systems:

Openness: We assume future assistive systems as ubiquitous software in daily life. Several vendors will share the market for such systems. Assistive systems should be easy to adapt to new hardware and functionality. Adding a new sensor or a new medical process requires extension of the used models in a transparent fashion.

Robustness: In daily use, assistive systems work without supervision of a highly-qualified researcher as it is the case of todays systems. Models of such systems have to be modified, extended, tailored by people not having a degree in formal logics and artificial intelligence. Decision models must be robust against small modeling errors because such errors are inevitable in such scenarios.

Adaptivity: Assistance models must be personalized without reprogramming. Furthermore, the usage of new or additional information, sensor data, and new processes should be supported. Therefore, and adaptation component is already at the development phase required, see Fig. 2.

Understandability: Because models are tailored by non-experts, we need explanation components and tools for "what-if" analysis. A common language is not available. However, the in Section 2 presented framework has to be applied to create a common basis for understanding.

Standardization: Exchange of data between the assistive system and other software systems, for example medical software can only be efficient if standards are respected. However, the functionality of the smart environment, *e.g.*, consisting of different assistive systems, should not be disturbed.

Further requirements in the development phase for building models exists. For decision making we need a formal background of the models as well as a traceability of decisions. So it is possible, to analyze not only the system but also to detect errors or identify potential for improvement. Using a model in different domains requires expressiveness, otherwise misinterpretation or attention of important parts might be The complexity of assistive systems and situations requires a modularization of models. For improvement and creating a common understanding a graphical knowledge representation might be helpful. A device uses knowledge on different abstract levels. Therefore, high-level knowledge modification operators are required. In smart environments different systems have to work in an integrated way, a support of cooperative assistance should already at the modeling level given.

5 Practical issues

In the discussion above, we have discussed at the semantic level – assuming that the semantics of a model is a decision theoretic problem.

Putting this to work on a real binary computer faces several practical challenges:

- Assistive systems often run distributed. In order to limit the required information exchange, such systems need to base their decisions on partial knowledge. How can the inference process and the exchange of information in such systems be designed so that the negative impact on – perceived and objective – system usefulness can be limited?

  Where can the analysis of the mathematical properties of models and utility functions guide the dynamic decomposition of a system into effectively independent partitions? (For instance: if the – expected – impact of the value of a state variable or a sensor observation is below a certain threshold, it can as well be ignored.)

- Doing a fully probabilistic reasoning is often too expensive computationally. Which simplifications can be used under what circumstances? (For instance, assuming independence between random variables, or using a sample-based representation – in the extreme of size one, the maximum likelihood estimate.)

- Performing operations – such as model combination – at the level of the model semantics is often prohibitively expensive or simply impossible. For instance, operations on mathematical functions by a computer are only possible for those functions that have a suitable finite syntactic representation: a lookup table, a parametric representation, a representation by samples, etc. These representations are used by an algorithm for computing a function value given a concrete argument and can also be used for some other operations on this function. Representations can also have different levels of abstraction – for instance, function-free first order logic can represent a set of possible worlds more compact than propositional logic, although both have the same representational power. Certain operations are easier at higher levels of abstraction – and vice versa.

  It is therefore an interesting question, what high-level languages and are usable for representing the models and what operations on models can be simplified by performing them at the syntactic level.

  In our own research, we look at utilizing PDDL for modeling causal behavior of users and devices. However, first results of modeling some simple meetings show that in order represent typical behavior, such models quickly become surprisingly complex: This complexity arises from the additional constraints that are required to capture the behavior space of ordinary humans: without these constraints, such models invent highly surprising – but strictly logical – explanations for observed sensor data. Therefore, we need additional modeling strategies that *naturally* contain the typical constraints of human behavior.

6 Summary

In this paper we showed the need for an abstract level for engineering assistive systems. Different domains have to integrate their models, data, and knowledge to create a new and holistic understanding for developing systems that incorporate cognitive modeling, sensor data, and decision support. Engineering assistance systems have to assemble human cognitive modeling, process modeling, and the inference from object recognition.

We see different challenges for developing and the usage of small and embedded devices as well as sensors to support and assist persons in daily life. An important aspect is the integration of different domains and their experts to enable an efficient engineering process. An improvement can only be achieved if all participants can easily access their models and objects whereas at the same time a consistent version is available. That means mappings of models and objects are necessary. Furthermore, requirements have to be used in all engineering steps. Although we doubt that there exists a world model or an efficient common language, the development of a common meta language is required and possible. The complexity of the development should be supported by a framework that already provides model and object mappings and enable the involved developers with a integrated tool.

References

[Agarwal *et al.*, 2007] Sheetal Agarwal, Anupam Joshi, Tim Finin, Yelena Yesha, and Tim Ganous. A pervasive computing system for the operating room of the future. *Mob. Netw. Appl.*, 12:215–228, March 2007.

[Baum and Edwards, 1993] C Baum and D F Edwards. Cognitive performance in senile dementia of the Alzheimer's type: the Kitchen Task Assessment. *The American journal of occupational therapy*, 47(5):431–436, 1993.

[Beetz *et al.*, 2008] Michael Beetz, Freek Stulp, Bernd Radig, Jan Bandouch, Nico Blodow, Mihai Dolha, Andreas Fedrizzi, Dominik Jain, Uli Klank, Ingo Kresse, Alexis Maldonado, Zoltan Marton, Lorenz Mösenlechner, Federico Ruiz, Radu Bogdan Rusu, and Moritz Tenorth. The assistive kitchen — a demonstration scenario for cognitive technical systems. In *IEEE 17th International Symposium on Robot and Human Interactive Communication (RO-MAN), Muenchen, Germany*, 2008.

[Bertsekas, 2005] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 3rd edition, 2005.

[Boger *et al.*, 2006] Boger, J. Hoey, P. Poupart, C. Boutilier, G. Fernie, and A. Mihailidis. A planning system based on markov decision processes to guide people with dementia through activities of daily living. *IEEE Trans. on Information Technology in Biomedicine*, 10(2):323–333, 2006.

[Brumitt *et al.*, 2000] Barry Brumitt, Brian Meyers, John Krumm, A Kern, and Steven Shafer. Easyliving: Technologies for intelligent environments. pages 12–29. Springer-Verlag, 2000.

[Burghardt *et al.*, 2011] Christoph Burghardt, Maik Wurdel, Sebastian Bader, Gernot Ruscher, and Thomas Kirste. Synthesising generative probabilistic models for high-level activity recognition. In Liming Chen, Chris D. Nugent, and Jit Biswas, editors, *Activity Recognition in Pervasive Intelligent Environments*, volume 4 of *Atlantis Ambient and Pervasive Intelligence*. Atlantis Press / World Scientific Press, 2011.

[Etzioni, 1991] Oren Etzioni. Embedding decision-analytic control in a learning architecture. *Artif. Intell.*, 49:129–159, May 1991.

[Gamma *et al.*, 1995] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1995.

[Gottfried and Aghajan, 2009] Björn Gottfried and Hamid Aghajan, editors. *Behaviour Monitoring and Interpretation – BMI Smart Environments*, volume 3 of *Ambient Intelligence and Smart Environments*. IOS Press, 2009.

[Heider and Kirste, 2002] Thomas Heider and Thomas Kirste. Supporting goal-based interaction with dynamic intelligent environments. In *Proceedings of the 15th European Conference on Artificial Intelligence*, pages 596–600, Lyon, France, JUL 2002.

[Hein *et al.*, 2009] Albert Hein, Christoph Burghardt, Martin Giersich, and Thomas Kirste. *Model-based Inference Techniques for detecting High-Level Team Intentions*, volume 3 of *Ambient Intelligence and Smart Environments*, chapter Model-based Inference Techniques for detecting High-Level Team Intentions, pages 257 – 288. IOS Press, Amsterdam, 09 2009.

[Hub *et al.*, 2003] Andreas Hub, Joachim Diepstraten, and Thomas Ertl. Design and development of an indoor navigation and object identification system for the blind. *SIGACCESS Access. Comput.*, pages 147–152, September 2003.

[Kautz *et al.*, 2002] Henry Kautz, Dieter Fox, Oren Etzioni, Gaetano Borriello, and Larry Arnstein. An overview of the assisted cognition project. In *AAAI-2002 Workshop on Automation as Caregiver: The Role of Intelligent Technology in Elder*, 2002.

[Kirste, 2011] Thomas Kirste. Making use of intentions. Technical Report CS-01-11, Institut für Informatik, Universität Rostock, March 2011.

[Monahan, 1982] George E. Monahan. A survey of partially observable markov decision processes: Theory, models, and algorithms. *Management Science*, 28(1):1–16, JAN 1982.

[Naeem and Bigham, 2008] Usman Naeem and John Bigham. Activity recognition using a hierarchical framework. In *Ambient Technologies for Diagnosing and Monitoring Chronic Patients Workshop, 2nd International Conference on Pervasive Computing Technologies for Healthcare 2008*, Tampere, Finland, January 29 2008.

[Nagla *et al.*, 2010] K.S. Nagla, M. Uddin, D. Singh, and R. Kumar. Object identification in dynamic environment using sensor fusion. In *IEEE 39th Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–4, 2010.

[Patterson *et al.*, 2004] Donald J. Patterson, Lin Liao, Krzysztof Gajos, Michael Collier, Nik Livic, Katherine Olson, Shiaokai Wang, Dieter Fox, and Henry Kautz. Opportunity knocks: A system to provide cognitive assistance with transportation services. In *In International Conference on Ubiquitous Computing (UbiComp*, pages 433–450. Springer, 2004.

[Puterman, 1994] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

[Roy *et al.*, 2007] Patrice Roy, Bruno Bouchard, Abdenour Bouzouane, and Sylvain Giroux. A hybrid plan recognition model for alzheimer's patients: interleaved-erroneous dilemma. In *2007 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, 2007.

[Serna *et al.*, 2007] Audrey Serna, Hélène Pigot, and Vincent Rialle. Modeling the progression of Alzheimer's disease for cognitive assistance in smart homes. *User Model. User-Adapt. Interact.*, 17(4):415–438, September 2007.

[Sung *et al.*, 2011] Jaeyong Sung, Colin Ponce, Bart Selman, and Ashutosh Saxena. Human activity detection from RGBD images. In *AAAI workshop on Pattern, Activity and Intent Recognition (PAIR)*, 2011.

[Thurow *et al.*, 2004] Kerstin Thurow, Bernd Göde, Uwe Dingerdissen, and Norbert Stoll. Laboratory information management systems for life science applications. *Organic Process Research & Development*, 8(6):970–982, 2004.

[Umblia *et al.*, 2009] Tobias Umblia, Albert Hein, Ilvio Bruder, and Thomas Karopka. Marika: A mobile assistance system for supporting home care. In *MobiHealthInf 2009 - 1st International Workshop on Mobilizing Health Information to Support Healthcare-related Knowledge Work*, Porto, Portugal, 12 2009.

[Yordanova, 2011] Kristina Yordanova. Toward a unified human behaviour modelling approach. Technical Report CS-02-11, Institut fr Informatik, Universitt Rostock, Rostock, Germany, May 2011. ISSN 0944-5900.