Partial Ordering Constraints for Representations of Context in Ambient Intelligence Applications*

Hedda R. Schmidtke and Woontack Woo

GIST U-VR Lab.
Gwangju 500-712, Korea
{schmidtk,wwoo}@gist.ac.kr

Abstract. We give an overview of a framework within which Ambient Intelligence (AmI) applications could be realised. We start from available sensors/actuators producing and consuming numerical data at high frequencies and available logic-based knowledge representation and reasoning systems using expressive logical languages. We argue that a constraint-based partial-order reasoning system for six aspects of context can be used as a central component that helps to bridge the gap between information from sensors and logic-based reasoning. Our work is motivated by, and oriented towards, cognitive representation and processing mechanisms.

1 Introduction

Ambient intelligence (AmI) has been suggested as a key application scenario for artificial intelligence techniques. In AmI scenarios (such as those compiled by Ducatel et al., 2001) smart devices – sensors, actuators, and information services – cooperate to provide services to users that make sense in a given situation and environment. Intelligent agent programs (see Wooldridge, 1999, for an overview) installed on such devices and connected by a network infrastructure can form coalitions, which together achieve (sub-)goals derived from user profiles or user-defined scripts.

A central component of such a spatially distributed intelligent system is a *context model*, i.e. a representation of the situation and environment that is at least partially shared among components. In particular, such a representation has to provide information of how the contexts of different components are related to each other and to the current physical context. This task can be formulated in terms of constraints on the relations between contexts.

Starting from the context model of Jang et al. (2005), we investigated a representation based on six parameters of context (for brevitiy called the *5W1H* parameters): the context of an interacting coalition of agents in this model is fully described if we know *who* interacts *when* and *where* with *what why* and *how*, that is, if we know the users, time, place, objects, events/actions, and the

* This research is supported by the UCN Project, the MIC 21C Frontier R&D Program in Korea.

states/conditions of an interaction. Following the analysis of Benerecetti et al. (2000), we assume that contexts correspond to portions of the world, which are perceived with a certain level of detail, and from a certain perspective. Accordingly, relations between contexts can be of three different types: mereologic, part-of relations (in a wide sense Bittner et al., 2004), approximation relations (particularly granularity, Schmidtke and Woo, 2007), and perspectival relations (such as cardinal directions in geographic space, Schmidtke, 2001).

In previous work (Hong et al., 2007), we introduced a logical language with mereologic relations for four of these six parameters, namely classes of users (*who*-domain) and classes of objects (*what*) with respective taxonomic relations, on the one hand, and temporal and spatial extents with mereologic relations, on the other hand. In Schmidtke et al. (2008), we extended the language to cover all six parameters, so that knowledge about states and events can also be represented. We characterised the six relations as partial orders. Consequently, a system for reasoning over such constraints on contexts can make use of partial order reasoning, which can be implemented in an efficient way using graph-based representations. In this paper, we show how such a reasoning system can be embedded into a cognitively motivated three-layered agent architecture (Lee et al., 2007) and we outline how AmI applications can be developed in this framework.

The paper has two main parts. In the first part (Sect. 2), we shortly introduce our cognitively motivated system architecture for AmI environments. We sketch how information from sensors is interpreted and translated into constraints on relations between sensor contexts and more abstract contexts; and we outline how this knowledge can be used to trigger or adapt behaviour of actuators and services, in order to reach sub-goals given from a planner or user-defined script. The second part (Sect. 3) gives details about the language and discusses how it can be used for representing knowledge about contexts. We sketch in how far our conceptualisation of context fulfils requirements from cognitive theories and qualitative reasoning. After discussing the architecture and language, we illustrate how AmI applications can be developed and integrated with a simple example (Sect. 4).

2 System Architecture

We developed a cognitively motivated layered multi-agent framework (Lee et al., 2007) to realise the requirements outlined above. We regard human cognition as the ideal model for AmI. In particular, we try to address the problem of how to bridge the gap between sensory data and logic-based representations with a cognitively motivated approach.

2.1 Processing Contextual Information

Agents in our model represent separate cognitive facilities of an intelligent system, which can be a single smart object or a whole smart environment composed

of many individual sub-systems. We assume three types of agents corresponding to layers of representational abstraction found in human cognition. Following the analysis of Gärdenfors (2005), we distinguish between three types of information processing and action of an intelligent system in its environment:

- In the case of *transduction*, behaviour is generated directly in response to perceptual input. Internal representations are not necessary. An example for this type of behaviour in animals is phototaxis.
- *Cued* representations are mental representations of objects or events that are perceptually directly accessible in the current context, or triggered by some recent situation. Using inference, e.g., categorisation, over these representations, an intelligent system can react to the situation.
- *Detached* representations are independent from the current context. Imagination about an object that does not exist or situations that have not, or not yet, happened are examples of inference over these representations. Planning requires detached thinking.

We identify these layers of abstraction with critical time frames of tasks in a dynamically changing computing environment.

Corresponding to transduction, the layer of *continuous responsiveness* is bound to the time frame of visual continuity, for which we assumed a maximal delay of 40ms. Sensors and actuators should react without any perceivable delay.

The layer of *immediate reaction*, on the other hand, is bound by the time frame of learnt reactions to change in an environment. Inference over representations cued by perceptual input are necessary to react to a situation. We chose a threshold of one second maximal delay for this layer. In the case of an obstacle appearing suddenly on a road, for instance, a driver would within one second be able to generate a more or less adequate reaction, such as breaking, steering, or a learnt sequence of these actions. Likewise in verbal communication, human beings usually require some response from a dialogue partner within the time frame of a second, even if it is just a nod or 'hmm' for signalling demand for a larger time frame. Our efforts were concentrated on realising this layer. In our implementation, the current context as retrieved from the responsive layer is sorted into a hierarchy of abstract contexts according to the constraints that can be derived from sensor readings.

The third layer is the layer of *pro-activity*. Intelligent actions in a changing but predictable world require computational processes of higher complexity. For reasoning and planning about anticipated situations, detached representations are necessary. Typical AI problems, which require expressive logic formalisms or planning are computed on this layer. In our current system, we do not implement this layer. We consider plans to be given to the system in terms of condition-action pairs and a partial ordering, for which the second layer provides reasoning support.

Figure 1 illustrates the three types of agents. In this example, a group of five agents is organised as a coalition for solving a certain problem: three responsive agents (smart sensors and actuators) are connected to a reactive component that

analyses sensor inputs and triggers accurate responses of actuator components; the analysed representation of the current context is also given on to a pro-active component, which generates a higher-level representation of the current context that can be used to modify the behaviour of the reactive component in case of unforeseen difficulties.

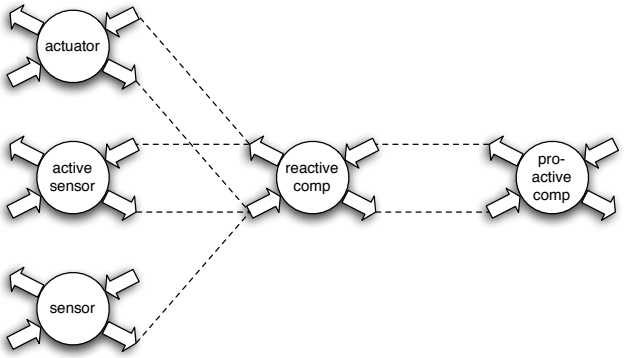


Fig. 1. Simple example for a coalition of five agents.

2.2 Representations on the Three Layers

The representations used on each layer have to differ so as to support the individual tasks within the required time frame and flexibility. Table 1 summarises the differences in perspective assumed for the three layers and illustrates the differences in representations for the example of spatial information.

Table 1. Representation and processing of contextual knowledge on the three layers. The example shows how spatial information could be handled on the three layers.

Layer	Representation	Processing	Example
Responsive	numeric, basic data types	procedural, non-symbolic	loc = $(35.226^\circ N 126.842^\circ E)$, at precision $\pm 0.001^\circ$
Reactive	context-oriented, qualitative relations	graph-based, constraint-based	[currC1 $\sqsubseteq_{\text{where}}$ Gwangju] $\wedge$ [Gwangju $\sqsubseteq_{\text{where}}$ SKorea]
Pro-active	logic-based	logic-based reasoning	$\forall x : \text{city}(x) \rightarrow \exists t :$ $\text{trainCon}(t) \wedge \text{reach}(t, x)$

On the layer of responsiveness, computation is performed directly on the mainly numerical input from sensors. This allows for especially fast processing, as required for algorithms at the sensor interface. The result of analysis can be

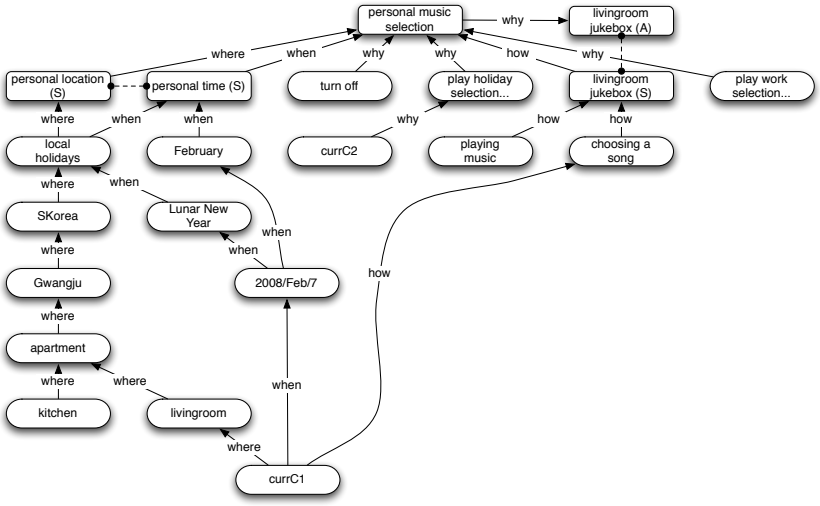


Fig. 2. The current context as retrieved from sensors can be sorted into a graph-based representation in which $\sqsubseteq_m$-constraints can be read from the edges. The statement [currC1 $\sqsubseteq_{\text{where}}$ Gwangju]$\wedge$[Gwangju $\sqsubseteq_{\text{where}}$ SKorea], from Tab. 1 can be retrieved from the above graph. Rounded rectangles represent agents, which are notified whenever their node in the graph is activated.

transferred directly to an actuator, or it can be transformed into a representation of constraints describing the *current context*. These constraints are distributed to recipient agents on the reactive layer.

On the reactive layer, information about parameters of the current context from different sources is integrated into a graph-based representation of the current state of the world around an agent (an example is shown in Fig. 2). Knowledge about the current context can be represented by activating nodes in the graph to which the current context is related. For long term storage, a new node representing the current context can be added to the graph by linking it to related nodes. Nodes in the graph represent previous contexts, knowledge about contexts given in application ontologies, and knowledge about contexts described in user profiles and scripts. In addition, nodes can be connected to agents. Activating such a node results in a description about the current context being sent to the agent.

Each edge of the graph corresponds to one of six specific relations, so that all information encoded in the graph can easily be translated into a logic-based format, which can be sent to recipients on the pro-active layer for further analysis. The identified situation can thus activate the next action to be executed from a given action plan, which in turn can automatically trigger or update actuators (see Sect. 4 for an example). A similar mechanism of spreading activation along edges with a specific semantics has been used by Kokinov (1999) to model a broad range of phenomena of context-dependency in human cognition. Our

variant for AmI differs mainly in two respects: first, the types of relations are restricted to the six partial ordering relations, so that the resulting graph for each relation is a directed acyclic graph; and second, nodes can be connected to arbitrary types of agents, so that arbitrary types of external applications can be triggered.

For the pro-active layer, logic-based knowledge representation systems can be used. The input from the reactive layer is sufficiently abstract, so as to be accessible to higher-level reasoning. Reasoning tasks necessary for AmI systems, such as planning or diagnosis, can thus be decoupled from the reactive systems. Information about the current context can be de-contextualised, in order to receive a more objective and far-reaching representation about ongoing and past processes. The behaviour of a reactive component can be generated or modified by a planner on the pro-active layer. Agents on the reactive layer can execute plans, given as partially ordered sequences of actions in the *why*-graph. However, the planning task itself can have a high complexity and might in general require more than one second, that is, more than the time frame of the reactive layer permits. Planning itself therefore belongs to the pro-active layer.

3 Representations of Context on the Reactive Layer

The reactive layer has been the focus of our research for three reasons. First, the time frame of one second demands a specialised fast reasoning mechanism for all domains of parameters of context. Second, the symbol grounding problem has to be handled on this layer, between the sensory input and the logic-based representations on the one hand, and between underspecified logic-based plans and the concrete parameters required by actuators, on the other hand. The third reason for highlighting the special role of the reactive layer, is that the notion of context is established and dealt with mainly on this layer. Agents on the responsive layer produce/consume absolute values, and agents on the pro-active layer de-contextualise the information they obtain. The cued representations, in contrast, are representations activated by, and relative to, the current physical context.

Crucial to realising this layer is an appropriate representation and reasoning mechanism that can handle the non-symbolic contextual information format of sensors/actuators as well as logic-based representations that can be processed on the pro-active layer. As sketched above, we assume that numeric values retrieved from sensors can be interpreted as basic constraints on the current context of the device. The context representation is based on partial ordering relations between six specific aspects of a context. With this approach we follow a similar strategy as in logics of indexicals (Forbes, 1989), which use a fixed, finite number of parameters. However, we represent time and location not with point-like time-stamps and coordinate locations but with a qualitative representation based on time intervals and spatial regions, respectively. In this manner, we can represent the relation between partial contexts as a containment relation and we can

allow for coarse location and time, such as "this country" and "this week" in combination with fine grained notions, such as "this room" and "this moment."

The relation of containment has been studied in theories of mereology. Properties stated in mereologic theories form the basic ontology of our framework, without implying that we only deal with part-whole structures in the narrow sense of spatial or temporal part-whole relations. We specify six partial ordering relations ($\sqsubseteq_{\text{who}}, \sqsubseteq_{\text{what}}, \sqsubseteq_{\text{when}}, \sqsubseteq_{\text{where}}, \sqsubseteq_{\text{why}}, \sqsubseteq_{\text{how}}$) relating between the six parameters that fully describe a context c in this approach.

The characterisation follows ideas in other approaches that combine knowledge about spatial relations with temporal knowledge or knowledge about concepts (Eschenbach, 2004; Donnelly, 2004; Bittner et al., 2004). However, we added reasoning about two other partial ordering relations for the domains of states and tasks: reasoning about conditions and states which we consider to be constrained by an implication relation (*how*-domain), and reasoning about tasks, actions and events in partially ordered causal structures (*why*-domain).

3.1 Logical Formalism

The logical language consists of the recursively defined context terms, representing contexts, constraints over *context terms* for relating contexts, and formulae for combining such descriptions. The set of context terms is defined as the smallest set over a set of atomic context terms that fulfils:

1. All atomic context terms and the special symbols $\top$ (called: the *maximal context*) and $\bot$ (the *impossible context*) are context terms.
2. If c and d are context terms then $-c$ (*complement*), $(c \sqcup d)$ (*summation*), and $(c \sqcap d)$ (*intersection*) are also context terms.

A context term is called *context literal* if it is an atomic context term or the complement of an atomic context term. A *context formula* is formed from two context terms with one of six sub-context relations:

1. If c and d are context terms, then $[c \sqsubseteq_{\text{who}} d]$, $[c \sqsubseteq_{\text{what}} d]$, $[c \sqsubseteq_{\text{when}} d]$, $[c \sqsubseteq_{\text{where}} d]$, $[c \sqsubseteq_{\text{why}} d]$, and $[c \sqsubseteq_{\text{how}} d]$ are atomic formulae.
2. If ϕ is a formula, then $\neg\phi$ also is a formula.
3. If ϕ and ψ are formulae, then $(\phi \vee \psi)$ and $(\phi \wedge \psi)$ are formulae.

Table 2 summarises the intended interpretations for the atomic formulae.

We introduce further relations as abbreviations of formulae. Here, and in the following we make use of schemata to abbreviate definitions: m denotes one of

Table 2. Syntax and reading of sub-context relations.

Syntax	Reading
$[c \sqsubseteq_{\text{who}} d]$	c is a social sub-context of d
$[c \sqsubseteq_{\text{what}} d]$	c is a conceptual sub-context of d
$[c \sqsubseteq_{\text{when}} d]$	c is a temporal sub-context of d
$[c \sqsubseteq_{\text{where}} d]$	c is a spatial sub-context of d
$[c \sqsubseteq_{\text{how}} d]$	c is a conditional sub-context of d
$[c \sqsubseteq_{\text{why}} d]$	c is a task sub-context of d

the six parameters of context, i.e. *who*, *what*, *when*, *where*, *why*, or *how*.[1]

$$[c \sqsubseteq d] \overset{def}{\Leftrightarrow} [c \sqsubseteq_{\text{who}} d] \wedge [c \sqsubseteq_{\text{what}} d] \wedge [c \sqsubseteq_{\text{when}} d] \wedge [c \sqsubseteq_{\text{where}} d] \wedge [c \sqsubseteq_{\text{why}} d] \wedge [c \sqsubseteq_{\text{how}} d] \tag{D1}$$

$$[c \bigcirc_m d] \overset{def}{\Leftrightarrow} \neg[c \sqcap d \sqsubseteq_m \bot] \tag{D2}$$

$$[c \bigcirc d] \overset{def}{\Leftrightarrow} \neg[c \sqcap d \sqsubseteq \bot] \tag{D3}$$

$$[c =_m d] \overset{def}{\Leftrightarrow} [c \sqsubseteq_m d] \wedge [d \sqsubseteq_m c] \tag{D4}$$

For two arbitrary context terms, c is a called a *sub-context* of d ($\sqsubseteq$) if it is socially, conceptually, temporally, spatially, and with respect to conditions, and tasks a sub-context of d (D1). Two contexts overlap with respect to the parameter m if their intersection is an m-sub-context (D2). The contexts c and d overlap if they have a common sub-context in any domain (D3), that is, if they overlap with respect to any one domain. Finally, (D4) defines that c and d are equal with respect to domain m if they are m-sub-contexts of each other.

Before we illustrate particular aspects of the six domains and illustrate the intended meaning for the 21 relations (the general $\sqsubseteq$, $\bigcirc$, together with identity $=$, and the six variants of $\sqsubseteq_m$, $\bigcirc_m$, and $=_m$), we shortly list some basic properties of partial ordering relations, which form the foundation for the semantics of the logical language (see Schmidtke et al., 2008, for a specification of the semantics based on Kripke frames). The below statements hold for arbitrary context terms x, x', x_1 independently from their meaning. We do not attempt an axiomatisation in this paper but refer to the wealth of known results from research on the properties of mereologic relations (cf. particularly Donnelly, 2004; Link, 1983).

For each of the relations $\sqsubseteq_m$, we state that $\sqsubseteq_m$ be reflexive (1) and transitive (2). Antisymmetry does not hold for $\sqsubseteq_m$ since two contexts that are identical with respect to one parameter may disagree with respect to another parameter. However, the relations $=_m$, which hold between contexts that agree on the parameter m, are equivalence relations, i.e reflexive (3), transitive (4), and

[1] For brevity, we also consider the additional logical conjunctives $\rightarrow$ and $\leftrightarrow$ to be defined as usual, and introduce rules for saving brackets. The following precedence of logical connectives is assumed $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$.

symmetric (5).

$$[x \sqsubseteq_m x] \tag{1}$$
$$[x_1 \sqsubseteq_m x_2] \wedge [x_2 \sqsubseteq_m x_3] \rightarrow [x_1 \sqsubseteq_m x_3] \tag{2}$$
$$[x =_m x] \tag{3}$$
$$[x_1 =_m x_2] \wedge [x_2 =_m x_3] \rightarrow [x_1 =_m x_3] \tag{4}$$
$$[x_1 =_m x_2] \rightarrow [x_2 =_m x_1] \tag{5}$$

The overlap relations $\bigcirc_m$ are reflexive for non-empty contexts (6) and symmetric (7) for any context term.

$$\neg[x \sqsubseteq \bot] \rightarrow [x \bigcirc_m x] \tag{6}$$
$$[x_1 \bigcirc_m x_2] \rightarrow [x_2 \bigcirc_m x_1] \tag{7}$$

With these properties stated, we can now illustrate their intended meaning with respect to the six domains with examples of statements in the logical language.

3.2 Modelling Knowledge about Individual Domains

Similar to the relation of containment between collections axiomatised by Bittner et al. (2004), the intended meaning for $\sqsubseteq_{\text{who}}, \sqsubseteq_{\text{what}}$ is *group inclusion* on groups of agents and groups of objects, respectively. With this interpretation, the properties stated above are intuitively plausible. An example for transitivity of $\sqsubseteq_{\text{who}}$, for instance, is given in (8): a group of users *admin* included in the group of users *staff* is also included in any group that includes the latter, such as *notificationRecipient*. Knowing that Bob is in the group of administrators, we know that he will receive a notification (9).

$$\begin{aligned}[admin \sqsubseteq_{\text{who}} staff] \wedge [staff \sqsubseteq_{\text{who}} notificationRecipient] \\ \rightarrow [admin \sqsubseteq_{\text{who}} notificationRecipient]\end{aligned} \tag{8}$$
$$[bob \sqsubseteq_{\text{who}} admin] \rightarrow [bob \sqsubseteq_{\text{who}} notificationRecipient] \tag{9}$$

In our mereologic framework, a single agent, such as *bob*, or a single object in an interaction is always interpreted as a group of one agent or object. That is, *bob* is not interpreted by a token corresponding to the user Bob but by the singleton set containing this token. This may seem to be a counter-intuitive by-effect of the mereologic axiomatisation. With respect to the cognitive motivation of our approach however, we might remark that this property has been shown by Link (1983) to have distinct advantages for the formal specification of nouns in natural language semantics: singular and plural meanings of a noun can be represented as having the same type with the mereologic, but not with a set-theoretic characterisation.

For time and space, we also use a mereologic interpretation for the two relations $\sqsubseteq_{\text{when}}, \sqsubseteq_{\text{where}}$. A mereotopologic characterisation of spatial entities, which starts from a mereologic basis, is discussed in detail by Casati and Varzi (1999).

For the domain of time, the discussion on interval-based calculi started by Allen (1984) serves as a reference. However, our notion is not restricted to convex intervals but covers arbitrary sums and intersections of intervals, such as generalised intervals (Ligozat, 1998), and moments, i.e. intervals with no extension.

Using this concept we could for instance represent a context c_{21} in which Allen and Beth are at Incheon Airport at January 1st in 2007.

$$[\mathit{allen} \bigcirc_{\text{who}} c_{21}] \wedge [\mathit{beth} \bigcirc_{\text{who}} c_{21}] \\ \wedge [c_{21} \sqsubseteq_{\text{when}} [\text{Jan 1, 2007}]] \wedge [\mathit{IncheonAirport} \bigcirc_{\text{where}} c_{21}]$$

Allen and Beth are users of the system in the context of c_{21}; c_{21} is at some time during January 1st and overlaps the region of Incheon airport.

In the case of agents and objects, taxonomies can be created by summation and intersection. It is important to note here, that we cannot construct arbitrary taxonomies and partonomies of locations and times with the operation of summing locations and times, respectively. For instance, it makes sense to construct a time *morningGMT* as the sum of all intervals between, e.g. 4 and 12 GMT. Likewise, we can define times *afternoonGMT* (12–18), *eveningGMT* (18–22), and *nightGMT* (22–4). The sum of these temporal entities, however, would be the trivial interval covering the whole of time, but not the set of days.

From an AI point of view, the parameter of *how* corresponds to states that hold in a certain context, whereas the parameter of *why* corresponds to events that occur, and actions that are executed in the context. Causal dependencies can be expressed by combining knowledge about states, events, and time. The context term operators, sum, intersection, and complement, can be used to combine states or events. The logical formalism thus provides in-built reasoning about such combinations, for which classical AI approaches based on first-order logic have to describe specific reification mechanisms (Galton, 2006). Our logical language, being designed for representing knowledge on the reactive layer, thus can support generating a description of the current context and triggering actions according to expected situations. However, it does not have the expressiveness and reasoning capabilities necessary to support reasoning about possible time lines, for instance. For this task more expressive formalisms, and thus reasoning on the pro-active layer, would be needed.

In concrete AmI applications, developers might use the *how*-part to represent status information, such as "on vacation," or information about states of a component, such as "playing music" in Fig. 2. Another type of states particularly relevant to AmI applications is qualitative information about environment parameters derived from quantitative sensory data. For instance, the state "cold weather" might be defined as holding whenever a temperature sensor yields a temperature below some threshold, such as "$0°C$." Agents that yield such qualitative statements should themselves take the context of the measurement in account, so that the context-dependency of adjectives, such as "cold," can be reflected: A summer day in Rome, for instance, would have a higher threshold for being called "cold" than a winter night in Moskau. The partial order structure underlying the *why*-component represents causal dependencies of events in

a context as given, for instance, in an action description (Brézillon, 2005) or from a planner. User preferences could also be encoded in such task structures.

In the example of Fig. 2, the time and location sensors activated all nodes that are ancestors of the node *currC1* along the *when* and *where* edges. Also the jukebox works as a sensor, notifying all related nodes about its current state along the *how*-edge. The music selection reacts to the context *currC1* by activating all nodes above *currC2*. In the example of Fig. 2, tasks of the music selection agent are thus handed on to the jukebox actuator along the edges of the *why*-graph.

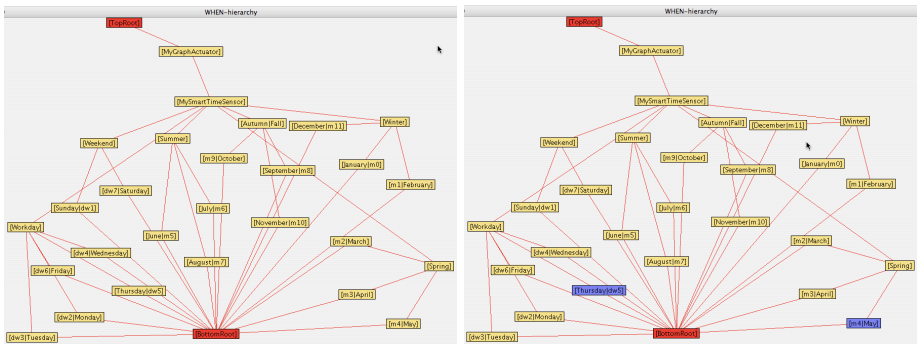
4 Example: Developing an AmI Application

The core components of the framework have been implemented. Classes for implementing the reactive components of the architecture, described in Sect. 2.2, and a knowledge base with a simple reasoning mechanism for the logical language have been realised in Java. Several test applications using this core framework are currently under development. The reasoning mechanism is implemented as a tableau prover supported by six directed acyclic graphs, which represent the partial ordering constraints between atomic context terms. The knowledge base receives input in the form of *profiles*, in which components describe their basic vocabulary and their own relation with respect to this vocabulary and other components.

When a sensor produces a value, a reactive component S wrapped around this sensor translates this value into a possibly complex context logic term c using its vocabulary. It then notifies the knowledge base about c. The intended meaning is that S activates c as a description of the current context as perceived and analysed by S. The reasoning mechanism then determines where the sent context term would be positioned in the directed acyclic graphs of the knowledge base. All related nodes, that is, all nodes that would be ancestors or descendants of a node corresponding to c are then notified about c by the knowledge base. Using this procedure, the description of the current context as generated by S is sent to all components to which it is relevant.

Figure 3 shows a screen shot of a graph drawing application being notified by a time sensor. The application displays the graph of the $\sqsubseteq_{\text{when}}$-relation, as it is represented in the knowledge base. It reacts to notifications by simply highlighting the most specific nodes activated. We developed this application as a tool for debugging. For being notified about terms related to the vocabulary of the time sensor the graph actuator's profile needs only a single line: `[MySmartTimeSensor pwhen MyGraphActuator]`. Figure 4 shows the context knowledge loaded with the time sensor.

Two classes were implemented for realising the time sensor and graph actuator, respectively. The complete code for the time sensor is shown in Fig. 5. It simply constructs a context term from integer values read from a time stamp. The notification sent by the time sensor results in statements, which identifiy the current time as being a time interval that can be described as the intersection



(a) (b)

Fig. 3. The graph actuator at two times: a) in idle state, b) after being notified of the context term `(y2008 sqand (dw6 sqand (dm15 sqand (h2 sqand (ampm1 sqand (hd19 sqand (min49 sqand (sec50 sqand m4))))))))` (here `sqand` is the ASCII-encoding for $\sqcap$).

```
[dw1 eqwhen Sunday]
[dw2 eqwhen Monday]
[dw3 eqwhen Tuesday]
[dw4 eqwhen Wednesday]
[dw5 eqwhen Thursday]
[dw6 eqwhen Friday]
[dw7 eqwhen Saturday]
[m0 eqwhen January]
[m1 eqwhen February]
[m2 eqwhen March]
[m3 eqwhen April]
[m4 eqwhen May]
[m5 eqwhen June]
[m6 eqwhen July]
[m7 eqwhen August]
[m8 eqwhen September]
[m9 eqwhen October]
[m10 eqwhen November]
[m11 eqwhen December]
[Workday pwhen MySmartTimeSensor]
[Weekend pwhen MySmartTimeSensor]
[Spring pwhen MySmartTimeSensor]
[Fall pwhen MySmartTimeSensor]
[Winter pwhen MySmartTimeSensor]
[Summer pwhen MySmartTimeSensor]
[Monday pwhen Workday]
[Tuesday pwhen Workday]
[Wednesday pwhen Workday]
[Thursday pwhen Workday]
[Friday pwhen Workday]
[Saturday pwhen Weekend]
[Sunday pwhen Weekend]
[March pwhen Spring]
[April pwhen Spring]
[May pwhen Spring]
[June pwhen Summer]
[July pwhen Summer]
[August pwhen Summer]
[September pwhen Fall]
[October pwhen Fall]
[November pwhen Fall]
[December pwhen Winter]
[January pwhen Winter]
[February pwhen Winter]
[Fall eqwhen Autumn]
```

Fig. 4. The profile file for the time sensor `MySmartTimeSensor.clf` (`pwhen` is the ASCII encoding for $\sqsubseteq_{\text{when}}$, `eqwhen` stands for $=_{\text{when}}$).

```
/** A sensor that notifies of the current time using Java's
Calendar class. **/
public class SmartTimeSensor extends SmartSensor {
Calendar cal;
   public SmartTimeSensor() {
      super(ContextProcess.TimeFrame.REACTIVE);
      this.cal = Calendar.getInstance();
   }
   /** Notifies about the time using Java's Calendar class. **/
   protected void doStep() {// called every second
      cal.setTime(new Date());
      if (cal.get(Calendar.SECOND)%10 == 0) {
         String year = "(y"+cal.get(Calendar.YEAR)+" sqand ";
         String dayofweek =
            "(dw"+cal.get(Calendar.DAY_OF_WEEK)+" sqand ";
         String dayofmonth =
            "(dm"+cal.get(Calendar.DAY_OF_MONTH)+" sqand ";
         String hour1 = "(h"+cal.get(Calendar.HOUR)+" sqand ";
         String ampm = "(ampm"+cal.get(Calendar.AM_PM)+" sqand ";
         String hourofday =
            "(hd"+cal.get(Calendar.HOUR_OF_DAY)+" sqand ";
         String minute = "(min"+cal.get(Calendar.MINUTE)+" sqand ";
         String seconds = "(sec"+cal.get(Calendar.SECOND)+" sqand ";
         String month = "m"+cal.get(Calendar.MONTH);
         mem.notify(year+dayofweek+dayofmonth+hour1
         +ampm+hourofday+minute+seconds+month+")))))))))");
      }
   }
   /** This sensor does not react to any input. **/
   public void onNotify(String str) {}
}
```

Fig. 5. The implementation of the time sensor class.

```
public static void main (String[] args) {
   SimpleCKB ckb = new SimpleCKB();
   SmartTimeSensor sim = new SmartTimeSensor();
   sim.init(ckb,"profiles/MySmartTimeSensor.clf","MySmartTimeSensor");
   xmpl.caedit.GraphActuator graph =
   new xmpl.caedit.GraphActuator(T5W1H.WHEN);
   graph.init(ckb,"profiles/MyGraphActuator2.clf","MyGraphActuator");
}
```

Fig. 6. Loading and starting the test application.

of several intervals represented in the *when*-hierachy in the knowledge base. The graph actuator class correctly identified the most specific active nodes shown to be `dw5` (temporally equivalent to `Thursday`) and `m4` (temporally equivalent to `May`). It was notified because it is itself an ancestor node of activated nodes.

The main method of the test application (Fig. 6) initialises the knowledge base (class `SimpleCKB`) and creates an instance of the time sensor. The profile of the time sensor is loaded into the knowledge base and the sensor is associated with the node `MySmartTimeSensor`, the top-node of all terms mentioned in the profile. The graph actuator is initialised to show the *when*-domain of the knowledge base. Its simple profile places it above the time sensor's node in the *when*-hierarchy. It can visualise any other class if another profile is loaded.

5 Outlook and Conclusions

We gave an overview of a framework within which AmI applications can be realised. We started from available sensors/actuators producing and consuming numerical data at high frequencies and available logic-based knowledge representation and reasoning systems using expressive logical languages. We argued that a constrained-based partial-order reasoning system is a core component that helps to bridge the gap between sensors and logic-based reasoning. Our work is motivated by, and oriented towards, cognitively adequate representations and processing mechanisms. The current framework is limited in that only partial ordering relations are represented. In future works, we will focus on integrating further types of relations, in particular, granularity relations and perspectival relations.

References

J. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23:123–154, 1984.

M. Benerecetti, P. Bouquet, and C. Ghidini. Contextual reasoning distilled. *Journal of Experimental and Theoretical Artificial Intelligence*, 12(3):279–305, 2000.

T. Bittner, M. Donnelly, and B. Smith. Individuals, universals, collections: On the foundational relations of ontology. In A. Varzi and L. Vieu, editors, *Third Conference on Formal Ontology in Information Systems*. IOS Press, 2004.

P. Brézillon. Task-realization models in contextual graphs. In A. K. Dey, B. N. Kokinov, D. B. Leake, and R. M. Turner, editors, *International Conference on Modeling and Using Context*, volume 3554 of *LNCS*, pages 55–68. Springer, 2005.

R. Casati and A. C. Varzi. *Parts and Places: the Structure of Spatial Representations*. MIT Press, 1999.

M. Donnelly. A formal theory for reasoning about parthood, connection, and location. *Artifcial Intelligence*, 160(1-2):145–172, 2004.

K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J.-C. Burgelman. Scenarios for ambient intelligence in 2010. Technical report, ISTAG, 2001.

C. Eschenbach. How to interweave knowledge about object structure and concepts. In D. Dubois, C. A. Welty, and M.-A. Williams, editors, *Principles of Knowledge Representation and Reasoning*, pages 300–310. AAAI Press, 2004.

G. Forbes. Indexicals. In D. Gabbay and F. Guenther, editors, *Handbook of Philosophical Logic*, volume IV, pages 463–490. D. Reidel, 1989.

A. Galton. Operators vs. arguments: the ins and outs of reification. *Synthese*, 150:415–441, 2006.

P. Gärdenfors. The detachment of thought. In C. Erneling and D. Johnson, editors, *The mind as a scientific subject: between brain and culture*, pages 323–341. Oxford University Press, 2005.

D. Hong, H. R. Schmidtke, and W. Woo. Linking context modelling and contextual reasoning. In A. Kofod-Petersen, J. Cassens, D. B. Leake, and S. Schulz, editors, *4th International Workshop on Modeling and Reasoning in Context (MRC)*, pages 37–48. Roskilde University, 2007.

S. Jang, E.-J. Ko, and W. Woo. Unified user-centric context: Who, where, when, what, how and why. In H. Ko, A. Krüger, S.-G. Lee, and W. Woo, editors, *Personalized Context Modeling and Management for UbiComp Applications*, volume 149 of *CEUR-WS*, pages 26–34, 2005.

B. N. Kokinov. Dynamics and automaticity of context: A cognitive modeling approach. In P. Bouquet, L. Serafini, P. Brézillon, M. Benerecetti, and F. Castellani, editors, *International Conference on Modeling and Using Context*, pages 200–213, 1999.

Y. Lee, H. R. Schmidtke, Y. Suh, and W. Woo. Realizing seamless interaction: a cognitive agent architecture for virtual and smart environments. In D. Hong and S. Jeon, editors, *International Symposium on Ubiquitous Virtual Reality*, volume 260 of *CEUR-WS*, pages 5,6, 2007.

G. Ligozat. Generalized intervals: A guided tour. In *Proceedings of the ECAI-98 Workshop on Spatial and Temporal Reasoning*, Brighton, UK, 1998.

G. Link. The logical analysis of plurals and mass terms: A lattice-theoretical approach. In R. Bäuerle, C. Schwarze, and A. von Stechow, editors, *Meaning, Use and Interpretation of Language*, pages 302–323. De Gruyter, 1983.

H. R. Schmidtke. The house is north of the river: Relative localization of extended objects. In D. Montello, editor, *International Conference on Spatial Information Theory*, volume 2205 of *LNCS*, pages 414–430, Berlin, 2001. Springer.

H. R. Schmidtke and W. Woo. A size-based qualitative approach to the representation of spatial granularity. In M. M. Veloso, editor, *Twentieth International Joint Conference on Artificial Intelligence*, pages 563–568, 2007.

H. R. Schmidtke, D. Hong, and W. Woo. Reasoning about models of context: A context-oriented logical language for knowledge-based context-aware applications. *Revue d'Intelligence Artificielle*, 2008.

M. Wooldridge. Intelligent agents. In G. Weiss, editor, *Multiagent Systems*. MIT Press, 1999.