Linking Context Modelling and Contextual Reasoning*

Dongpyo Hong, Hedda R. Schmidtke, Woontack Woo**

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

Abstract. In this paper, we discuss a novel perspective on ontology-based context modelling that makes it easy to combine context models and contextual reasoning mechanisms. On the context modelling side, we outline our idea of a user-centric context model based on the six fundamental context parameters of *who*, *when*, *where*, *what*, *how*, and *why* (5W1H); on the contextual reasoning side, we introduce syntax and semantics for a simple logical language and sketch a tableau mechanism for reasoning. The model-theoretic semantics for this logical language is like the context model based on the parameters of 5W1H. With the common semantics, it is then easy to show the link between context modelling and contextual reasoning.

1 Introduction

For any research on context-aware, intelligent computer systems, a fundamental question is how to represent context. As perspectives of research and targeted types of context-awareness differ, the specific properties of representations of context also differ (cf. the range of perspectives surveyed in [3]). However, the concept to be represented, that is *context*, is the same; and we can expect that there are interfaces and mappings between different types of *representations of context*. Thus, we present an approach to describe such an interface for the example of a representation of context from the area of *context modelling* and a representation of context from the area of *contextual reasoning*.

A definition of context that has been accepted widely in the area of context-aware applications has been given by Dey and Abowd [6]: "Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." A problem with this definition is that it does not differentiate between context and information about context. A more recent definition by Bardram [1] better reflects the fact that *context* exists outside a representation system: " 'Context'

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

refers to the physical and social situation in which computational devices are embedded." A *context model* can then be conceived of as a data model suitable for storing information about the context of a certain interaction event.

Mobile context-aware computing has to cover issues of sensor reliability, ad hoc network communication, software development support, reasoning and inference, usability, and privacy management. Many of the main differences between approaches can accordingly be traced back to emphasis on one or the other aspect, such as sensor-fusion in [21], networking in [20], and development of context-aware applications in [11]. Most recently, ontology-based approaches [22, 18, 9] have gained importance to answer the demands of heterogeneous application environments. The key idea of ontology-based context modelling is that applications using the context model also have to agree on a common ontology, that is, a set of basic concepts defined in a formal language, which developers can use to specify application specific concepts. Application concepts, being founded upon the same basic concepts, can then be used for communication between different applications.

When human beings reason or communicate about objects and events in the environment, they usually abstract from certain aspects, and reason within a context. When we reason about space, for instance, we may use *to the West of* as a transitive relation (cf. the calculus in [15]). This assumption is valid as long as we suppose a sufficiently small local area of context, as *to the West of* is globally a cyclic relation: Denmark is to the West of Korea, Korea is to the West of Canada, and Canada is to the West of Denmark; within the local context of a city or country, in contrast, *to the West of* can be used in the same manner as *to the North of*, i.e. as if it was a transitive, acyclic relation.

Research in *contextual reasoning* investigates how such locally valid theories can be connected and how inferences can be made across context boundaries: Benerecetti, Bouquet, and Ghidini [2] investigate the basic principles and tasks of contextual reasoning. They use the metaphor of *context as a box* containing the contextualised logical sentences together with a list of parameters and values for these. These parameters establish the link between the contextualised sentences and the surrounding or neighbouring boxes. In [2], three dimensions of context dependence are distinguished: contextualised representations can be *partial*, *approximative*, or *perspectival* representations. The three dimensions of context dependence can be illustrated with respect to the parameter of spatial context: changing from a global to a partial view of space can be identified with spatially focusing on a sufficiently small local area; a change of spatial perspective can be identified with a change of reference frames; and changing the degree of approximativeness can be related to coarsening and refinement of spatial granularity. The focus of this article is on the aspect of *partial* representations.

The key idea of our research was to link a user-centric context model for context-aware applications (Sect. 2), which can provide a system with access to sensory information, and a logic-based contextual reasoning mechanism (Sect. 3) with a common semantics that has its roots in six fundamental parameters of context: both representations of context are understood as describing circum-

stances of a certain *interaction* as a proposition stating that someone (*who*) interacts somehow (*how*) and for a certain reason (*why*) with something (*what*) at a given time (*when*) and place (*where*). We sketch how contextual information retrieved from sensors can be stored and accessed with the context model and how information stored in the context model can be translated into the logical language (Sect. 4).

2 User-centric Context Model

Regarding context model, there have been many research activities from artificial intelligence to mobile, ubiquitous, and pervasive computing. For instance, Chen and Kotz [4] gave a survey of various definitions of context and its applications, particularly in the area of mobile computing. Dey and Abowd [6] explored context from the area of context-aware computing. However, most context models take an application centred perspective. From the perspective of human-computer interactions in contrast, we are more interested in how contextual information is perceived by users rather than by devices, services, or applications. Thus, we define context as user-centric context, that is, described from the perspective of the user: *User-centric context* is represented as a sequence of explicit and implicit information that occurs in a user's interactions with applications.

Each sequence of the proposed context model consists of a set of WHO, WHEN, WHERE, WHAT, HOW, and WHY (for short: 5W1H) which can provide the bridge into the ontology and thus into contextual reasoning (Sect. 3). Sensory information, inferred information, and information entered by a user are sorted accordingly into one of the six categories. Table 1 shows what each category represents in a sequence. Explicit information is the user's direct inputs

Table 1. Context categories

Category	Description	Examples
WHO	Basic user information	name, gender, birthday
WHEN	Time	time stamp, time of day, season
WHERE	Location	coordinate (x,y), place, region
WHAT	Relevant objects	applications, services, commands (application dependent)
HOW	Ongoing processes	signals from sensors, e.g. current activity of the user (sensor dependent)
WHY	Users' intentions	stress, emotion, future events from a schedule

(e.g., from keyboard, mouse, pen) and signals from sensors (e.g., acceleration, temperature, GPS). Implicit information is information inferred from explicit information. In order to turn sensory raw data into information required by services, they have to be accumulated, processed, and integrated [12]. According to

the state of processing, user-centric context can be categorised as *preliminary*, *integrated*, and *final context*.

Preliminary Context (PC) stores primitive data from sensors and primitive features from the data.

Integrated Context (IC) contains accumulated preliminary contexts and inferred information, particularly from sensor-fusion.

Final Context (FC) is the context representation received from and sent to applications. Both a user's expectations (e.g., privacy concerns) and the application's demands for information have to be regarded whenever information is sent to an application.

For example, explicit user input can be the birthday of the user like `birthday` $= x$ in the preliminary context. In the integrated context, it can be expressed as `age` $= f(x, d)$ where $f()$ is a function for counting the number of years between two dates or time stamps.

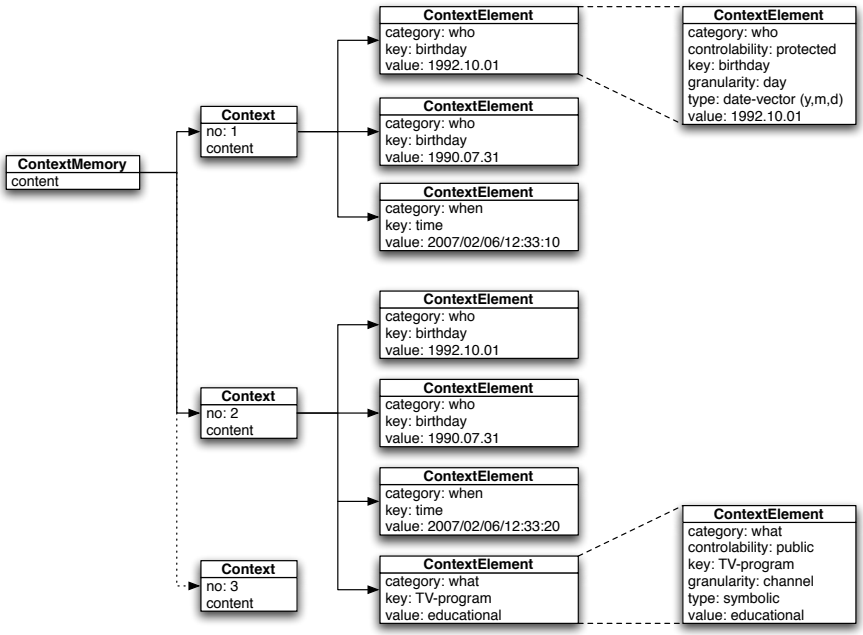

Fig. 1. *Context* objects (whether PC, IC, or FC) consist of *context elements* (here simplified with detailed views for two examples) and are collected into *context memory*.

The proposed context model consists of three parts as follows (Fig. 1).

ContextElement is a basic type in the proposed context model and consists of six attributes: *category* (one of 5W1H), *controlability*, *key*, *granularity*, *type*, and *value*.

Context objects store all information available at a certain time and thus contain a full description of a context as it is retrieved from sensors. The Context class is implemented as a set of contextual elements. For accessing elements in a context object, we can use the category together with the key.

ContextMemory stores context objects as long as they are necessary, because context-aware applications might need to access not only current but also previously collected contextual information. It is implemented as an ordered collection based on time and provides different search facilities.

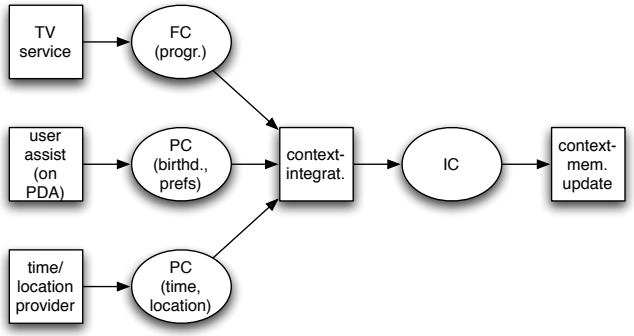

Fig. 2. Context memory updates

Figure 2 illustrates the processing of context in our application framework [12] for a simple example. Context objects (PC and FC) are produced from three sources: a TV service provides information about the currently shown program; the user assistant is a program running on users' PDAs that provides information about users to the environment according to privacy demands; thirdly, a local beacon issues the name of its location and a time stamp signal. Since sensor fusion mechanisms or handling of uncertainty are not necessary in this simple example, context integration can simply unite all PCs it receives between two time stamps into one integrated context object. Finally, the collected information is stored in context memory. Figure 1 shows part of context memory in two consecutive contexts: in context 1, two users are present, each disclosing information about their birthday; in context 2 10s later, the same data about users is broadcast, but additionally the TV service broadcasts that currently an educational program is shown.

If we consider the context objects to be simply collections of pieces of information, the additional information from the category of 5W1H is only an additional sorting criterion. The link into the ontology and thus to contextual reasoning is only established if it makes a semantic difference to which category a certain entry belongs. As an example, compare the *who*-element that contains the birthday of a user with the *when*-element containing the current time in Fig. 1. Above, we mentioned that the *who*-elements provide information about

who are the agents of an interaction event, whereas *when*-elements specify the time of interaction. From a programming perspective, both context elements contain values of date/time data type. Ontologically however, the *birthday*-element must indicate users, and only the *time*-element specifies time. With set theoretical semantics we can realise this demand by stating that each *who*-element describes a set of users – in the example: the set of users whose birthday is on the given date – whereas the *when*-element describes a portion of time, i.e. a set of time points. Given such a mapping of the data structures to standard set-theoretical semantics, we can encode our ontology in a logical language in a straight forward manner.

3 Reasoning about Context Objects

Ontologies support three major issues in the development of context-aware ubiquitous computing applications [18]: discovery and matchmaking, interoperability, and context-awareness. Ontology languages based on description logics (DL), such as OWL (used, e.g., by [22, 9]), particularly support formulation of taxonomic knowledge, which is required for the first two tasks. Reasoning about the classical domains of context, in particular space and time however, requires additional expressive power. Other approaches to ontology-based context modelling use first order logic [18, 9], or F-Logic [22] for additional expressiveness. Ranganathan et. al [18] believe this need for more expressiveness to be caused by space and time being quantitative domains. However, research in qualitative reasoning has shown that space and time can be reasoned about efficiently with qualitative representations [19], and that users are more comfortable with qualitative than with quantitative interfaces to, e.g., spatial knowledge [7].

Qualitative spatial, temporal, and taxonomic knowledge has been handled previously in separate specialised logical languages, and combined languages are only recently being explored [13]. To see that it is not trivial to add, e.g., spatial relations into taxonomic knowledge consider the integration of the RCC-relations [17] in SOUPA [5]: the formal specification[1] expresses, for instance, only that *proper part* and *part* are transitive relations, but not that *part* is a reflexive relation and *proper part* is irreflexive; or that any proper part of a region is also a part of that region. Moreover, the SOUPA specification of space[2] adds a relation *spatiallySubsumes*, which is supposed to provide spatial containment reasoning [5, Sect. 3.1.5]. However, its relation to the RCC-relations, or whether it is reflexive or irreflexive is not specified.

Following a similar approach as [13], we characterise a specialised logical language that combines reasoning about the specific domains. Our main idea for translating knowledge stored in the context model into a format suitable for logical reasoning is to use *context terms* corresponding to the *context objects* from the context modelling side as the atomic units for the logical language. Each context object and therefore also each context term is understood as describing

[1] http://pervasive.semanticweb.org/ont/2004/06/rcc
[2] http://pervasive.semanticweb.org/ont/2004/06/space

circumstances of a certain *interaction* by a proposition stating that someone (*who*) interacts somehow (*how*) and for a certain reason (*why*) with something (*what*) at a given time (*when*) and place (*where*). To simplify the discussion, we only present the logical framework for reasoning about four parameters of an interaction, namely the spatial, temporal, object-related taxonomic, and agent-related taxonomic parameters.

The syntax of the logical language is similar to that of a description logic. In description logics we have two types of expressions: concepts and formulae, where only concepts are recursively defined. Similarly, the logical language defined in this paper consists of the recursively defined context terms, denoting circumstances of an interaction, and formulae. The set of context terms is defined based on a set of atomic context terms as the smallest set that fulfils:

1. All atomic context terms and the special symbols $\top$ (the maximal or trivial context), and $\bot$ (for the empty or impossible context) are context terms.
2. If c and d are context terms then the complement $\neg c$, the sum $(c \sqcup d)$, and the intersection $(c \sqcap d)$ are also context terms.

In comparison to concepts in description logics, we currently do not allow constructions involving quantification. Instead, we encode some of the necessary functionality into different operators for generating formulae out of context terms. A *context formula* is formed from two context terms with one of five operators: if c and d are context terms, then $c \sqsubseteq d, c \sqsubseteq_{\text{who}} d, c \sqsubseteq_{\text{what}} d, c \sqsubseteq_{\text{when}} d, c \sqsubseteq_{\text{where}} d$ are context formulae to be read as summarised in Tab. 2 below. A *contextual knowledge base* (CKB) is defined as a set of context formulae.

We can now specify the semantics of this simple language. For representing the four aspects, the domain of discourse consists of quadruples of subsets of four distinct sub-domains: $U = 2^{U_{\text{A}}} \times 2^{U_{\text{O}}} \times 2^{U_{\text{T}}} \times 2^{U_{\text{S}}}$, where U_{A} is the set of all users, U_{O} is the set of all objects, U_{T} is the set of all temporal points, and U_{S} is the set of all spatial points, i.e. the area of the domain. Any context term c is interpreted as a quadruple $\langle a, o, t, r\rangle \in U$ corresponding to a sentence: "some members of the group of *agents* a interact with some *objects* in o at a *time* in t somewhere in the *region* r." The interpretation function I maps the context terms to elements of U. The special symbols $\top$ and $\bot$ and the context term operators are interpreted in the following way:

$I(\top) = \langle U_{\text{A}}, U_{\text{O}}, U_{\text{T}}, U_{\text{S}}\rangle$ and $I(\bot) = \langle \emptyset, \emptyset, \emptyset, \emptyset\rangle$,

$I(c \sqcap d) = \text{and}(I(c), I(d))$, with *and* being the piecewise intersection:

$$\text{and}(\langle a_1, o_1, t_1, s_1\rangle, \langle a_2, o_2, t_2, s_2\rangle) = \langle a_1 \cap a_2, o_1 \cap o_2, t_1 \cap t_2, s_1 \cap s_2\rangle$$

$I(c \sqcup d) = \text{or}(I(c), I(d))$, with *or* being the piecewise union:

$$\text{or}(\langle a_1, o_1, t_1, s_1\rangle, \langle a_2, o_2, t_2, s_2\rangle) = \langle a_1 \cup a_2, o_1 \cup o_2, t_1 \cup t_2, s_1 \cup s_2\rangle$$

$I(\neg c) = \text{comp}(I(c))$, where *comp* is the piecewise complement:

$$\text{comp}(\langle a, o, t, s\rangle) = \langle (U_{\text{A}} \setminus a), (U_{\text{O}} \setminus o), (U_{\text{T}} \setminus t), (U_{\text{S}} \setminus r),)\rangle$$

The basic relation underlying the semantics of the operators is the relation of *containment* ($\subset$) as shown in Tab. 2. With the interpretation function I and the

Table 2. Syntax and semantics of operators. The semantics is given with respect to two context terms c and d with $I(c) = \langle a_c, o_c, t_c, r_c\rangle$ and $I(d) = \langle a_d, o_d, t_d, r_d\rangle$.

Syntax	Semantics	Reading
$c \sqsubseteq_{\text{who}} d$	is true, iff $a_c \subset a_d$	c is socially a sub-context of d
$c \sqsubseteq_{\text{what}} d$	is true, iff $o_c \subset o_d$	c is conceptually a sub-context of d
$c \sqsubseteq_{\text{when}} d$	is true, iff $t_c \subset t_d$	c is temporally a sub-context of d
$c \sqsubseteq_{\text{where}} d$	is true, iff $r_c \subset r_d$	c is spatially a sub-context of d
$c \sqsubseteq d$	is true, iff $a_c \subset a_d$, $o_c \subset o_d$, $t_c \subset t_d$, and $r_c \subset r_d$	c is a sub-context of d

domain U defined, entailment from CKBs can be derived in the standard way. A structure $\langle I, U\rangle$ is a model for a formula ϕ, iff ϕ is true in $\langle I, U\rangle$. We obtain five variants for the semantic concept of satisfiability: we call a context term c *spatially (temporally, conceptually, socially) satisfiable* iff a structure $\langle I, U\rangle$ exists in which $c \sqsubseteq_{\text{where}} \bot$ ($c \sqsubseteq_{\text{when}} \bot$, $c \sqsubseteq_{\text{what}} \bot$, $c \sqsubseteq_{\text{who}} \bot$) does not hold; c is *strongly satisfiable* iff it is spatially, temporally, conceptually, and socially satisfiable.

We suggest a simple tableau mechanism (cf. [8] for an introduction and overview) for reasoning with CKBs. In order to ask a question, such as whether $c \sqsubseteq_{\text{where}} d$ holds in a CKB, we ask whether the context term $(c \sqcap \neg d)$ is spatially unsatisfiable. More exactly, we ask whether $q \sqsubseteq_{\text{where}} (c \sqcap \neg d)$ entails $q \sqsubseteq_{\text{where}} \bot$, for an arbitrary new context term q (for *query*). We can then start the tableau algorithm with the CKB and a set of query formulae as given in Tab. 3.

Table 3. Examples for questions to the CKB. The query context term q does not appear anywhere else in the CKB. The formula $c \sqsubseteq d$ is expanded to $\{c \sqsubseteq_{\text{where}} d, c \sqsubseteq_{\text{when}} d, c \sqsubseteq_{\text{what}} d, c \sqsubseteq_{\text{who}} d\}$. Note that the semantic concept of satisfiability of a context term cannot be expressed within the logical language itself, since negation of a formula cannot be expressed.

Question	Formula	Negated Query Set Q
Is a context as described by the term c satisfiable?	no positive form	$\{q \sqsubseteq c\}$ expanded: $\{q \sqsubseteq_{\text{where}} c, q \sqsubseteq_{\text{when}} c, q \sqsubseteq_{\text{what}} c, q \sqsubseteq_{\text{who}} c\}$
Do the interactions of c take place in the region of d?	$c \sqsubseteq_{\text{where}} d$	$\{q \sqsubseteq_{\text{where}} (c \sqcap \neg d)\}$
Do the interactions of c involve objects of d?	$c \sqsubseteq_{\text{what}} d$	$\{q \sqsubseteq_{\text{what}} (c \sqcap \neg d)\}$

The algorithm starts with the set $T = \{CKB \cup Q\}$ containing as the only branch $CKB \cup Q$. In every step, we expand a branch $S \in T$: we first check whether S is a closed branch, that is, whether it contains for some operator $\sqsubseteq_m$, either $q \sqsubseteq_m \bot$, or both $q \sqsubseteq_m c$ and $q \sqsubseteq_m \neg c$ for some context term c. If S is closed, it is removed from T. If $T = \emptyset$, the tableau is closed, and the query has been proved. As long as there is still an open branch $S \in T$, we select a formula ϕ from S and modify S according to the rules given in Tab. 4. In case of the β-rules, we replace S with two branches. If a branch in T cannot be closed and no rule is applicable, the query has been disproved.

Table 4. Simple rules for basic contextual reasoning. For each rule $\sqsubseteq_m$ signifies one of $\sqsubseteq_{\text{who}}, \sqsubseteq_{\text{where}}, \sqsubseteq_{\text{when}}, \sqsubseteq_{\text{what}}$, so that we obtain a total of $4 * 10 = 40$ rules.

Name	Input (ϕ in branch S)	Output ($S' = S \setminus \{\phi\}$)
q-intro	$c \sqsubseteq_m d$ (with $c \neq q$)	$S' \cup \{q \sqsubseteq_m (\neg c \sqcup d)\}$
$\bot$-elim	$q \sqsubseteq_m c \sqcup \bot$	$S' \cup \{q \sqsubseteq_m c\}$
$\bot$-abs	$q \sqsubseteq_m c \sqcap \bot$	$S' \cup \{q \sqsubseteq_m \bot\}$
$\top$-elim	$q \sqsubseteq_m c \sqcap \top$	$S' \cup \{q \sqsubseteq_m c\}$
$\top$-abs	$q \sqsubseteq_m c \sqcup \top$	$S' \cup \{q \sqsubseteq_m \top\}$
$\neg$-elim	$q \sqsubseteq_m \neg\neg c$	$S' \cup \{q \sqsubseteq_m c\}$
α-rule 1	$q \sqsubseteq_m (c \sqcap d)$	$S' \cup \{q \sqsubseteq_m c, q \sqsubseteq_m d\}$
α-rule 2	$q \sqsubseteq_m \neg(c \sqcup d)$	$S' \cup \{q \sqsubseteq_m \neg c, q \sqsubseteq_m \neg d\}$
β-rule 1	$q \sqsubseteq_m (c \sqcup d)$	two branches $S' \cup \{q \sqsubseteq_m c\}$ and $S' \cup \{q \sqsubseteq_m d\}$
β-rule 2	$q \sqsubseteq_m \neg(c \sqcap d)$	two branches $S' \cup \{q \sqsubseteq_m \neg c\}$ and $S' \cup \{q \sqsubseteq_m \neg d\}$

4 Discussion

With both the context model and the context reasoning mechanism described, we can sketch how the two components can be linked. The proposed context model provides us with the necessary data model to store and process sensory data in a common format, in our approach the context elements. All data collected about the same situation together yield the context object. We therefore can state that the context object represents the situation as it is *perceived* by the system. Contextual reasoning comes in, as these perceptions are assigned a *meaning* by sorting them into the conceptual scheme provided by application ontologies: we might, for instance, classify a certain range of sensory values from a physiological sensor as signalling a critical health condition. With this classification we link the perception into our representation of the world. When the percept is classified, it becomes a new fact in the CKB.

Thus, we can compare the context model with a representation of perceptions, whereas contextual reasoning handles representations of knowledge. The gap between context modelling and contextual reasoning can then be identified as the well-known grounding problem of how knowledge is anchored in perception [10].

Context modelling, designed to support the generation of more and more abstract classifications – from *preliminary context* retrieved from sensors to *integrated context* required for triggering services –, can be understood as an effort to bridge this gap from the sensory or perceptual side.

Application ontologies can be encoded in our framework as consistent sets of sentences, i.e., as axiomatic systems formulated in the logical language. They form the static basis of the CKB in a running context-aware system. Additionally, the context-aware system can dynamically expand the CKB with information from the context objects. Every newly constructed context object, the `currentContext` object, can be translated into a context term that can be sorted into the (taxonomic, spatial, and temporal) hierarchies encoded in the CKB, in order to be available for reasoning. Classification of context terms requires two steps, first, *algorithmic classification*, a step of abstraction in context processing, and second, *ontology-based classification*, using the reasoning mechanism. Consider, for instance, a context-aware media centre in a smart home environment, such as the ubiTV application [16].

Algorithmic classification In the first step, an application developer has to provide methods to convert collections of context elements into context terms, thus assigning a meaning to them with respect to the semantics of the logical language. What does it mean, for instance, if there are two context elements with the *birthday*-key and different values in a context, as in Fig. 1 (p. 4)? Since the elements are in the *who*-domain they have to be interpreted as denoting groups of users. An interprperspicuousetation of this context object that makes sense is to assume that the group of users in this context is contained in the set of all persons whose birthday is on one of the two dates, i.e., as the union of the groups of users described by each context element. The result of algorithmic classification is a description of the current context in terms of the application ontology, in the example, a classification of users according to age and of TV-programs according to content might be relevant for the TV-application:

$$\begin{aligned} \texttt{currContext} &\sqsubseteq_{\text{who}} \text{Teenager} \\ \texttt{currContext} &\sqsubseteq_{\text{what}} \text{EducationalTVProgram} \end{aligned}$$

Ontology-based classification After algorithmic classification has been used to generate a description of the situation in terms of the application ontology, we can further reason about the situation within the logical framework. With the following statements from an application ontology, for instance,

$$\begin{aligned} \text{EducationalTVProgram} &\sqsubseteq_{\text{what}} \text{TVProgram}, \\ \text{TVProgram} &\sqsubseteq_{\text{what}} \text{NeedsSoundResource} \sqcap \text{NeedsPictureResource} \end{aligned}$$

the media centre can conclude that the current context requires sound resources

$$\texttt{currContext} \sqsubseteq_{\text{what}} \text{NeedsSoundResource}.$$

5 Conclusion and Future work

We discussed a novel perspective on ontology-based context modelling that makes it easy to combine context models and contextual reasoning mechanisms. On the context modelling side, we outlined our idea of user-centric context modelling; on the contextual reasoning side, we introduced syntax and semantics for a simple logical language. The expressiveness of this first, simple language does not go beyond propositional logics and further extensions are a focus of ongoing works. However, even this simple language already supports reasoning about main parameters of context (who, when, where, what) in a unifying and perspicuous way, and is sufficient to encode, e.g., the location hierarchies proposed by Leonhardt [14].

From the context modelling perspective, this paper illustrates how a logical query language with a clear semantics can be used to provide contextual reasoning capabilities as well as model-theoretic semantics to a context model. From the contextual reasoning side, the context model provides a way for grounding knowledge-based reasoning about context in sensory or perceptual data about the real world.

The paper provided only an overview of our approach. Details, e.g., regarding how context integration is performed on the context modelling side, or the proof of formal properties of the logical language have not been shown. However, one of the key benefits of our approach to ontology-based context modelling is that questions regarding completeness and soundness of the logical framework can be asked and answered much easier and clearer than in the conventional approach to ontology design based on DL/OWL; and even more so, since several prominent approaches to ontology-based context modelling are formulated using a combination of DL with more expressive logical languages. We conclude that context-aware applications can benefit from specific context logics, and a focus of ongoing works is the detailed investigation of more expressive logical languages for specifying context ontologies. Among the desiderata are, particularly, extensions for representing linear orders, such as the temporal *before*, in order to move towards a fully-fledged contextual reasoning mechanism [2].

References

1. J. E. Bardram. The Java context awareness framework (JCAF) - a service infrastructure and programming framework for context-aware applications. In H.-W. Gellersen, R. Want, and A. Schmidt, editors, *Pervasive Computing, Third International Conference*, pages 98–115, 2005.
2. M. Benerecetti, P. Bouquet, and C. Ghidini. Contextual reasoning distilled. *Journal of Experimental and Theoretical Artificial Intelligence*, 12(3):279–305, 2000.
3. P. Brézillon. Context in problem solving: A survey. *The Knowledge Engineering Review*, 14(1):1–34, 1999.
4. G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, November 2000.

5. H. Chen, F. Perich, T. Finin, and A. Joshi. SOUPA: Standard ontology for ubiquitous and pervasive applications. In *First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, 2004.
6. A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Workshop on The What, Who, Where, When, and How of Context-Awareness*, 2000.
7. M. J. Egenhofer. Spatial SQL: A query and presentation language. *IEEE Transactions on Knowledge and Data Engineering*, 6(1):86–95, 1994.
8. M. Fitting. *First-Order Logic and Automated Theorem Proving*. Springer, 1996.
9. T. Gu, H. K. Pung, and D. Q. Zhang. A service-oriented middleware for building context-aware services. *Journal of Network and Computer Applications*, 28(1):1–18, 2005.
10. S. Harnad. The symbol grounding problem. In *Encyclopedia of Cognitive Science*. Macmillan and Nature Publishing Group, 2003.
11. K. Henricksen and J. Indulska. Developing context-aware pervasive computing applications: Models and approach. *Pervasive and Mobile Computing*, 2:37–64, 2006.
12. D. Hong, Y. Suh, A. Choi, U. Rashid, and W. Woo. wear-UCAM: A toolkit for mobile user interactions in smart environments. In *Embedded and Ubiquitous Computing*, pages 1047–1057. Springer, 2006.
13. O. Kutz, C. Lutz, F. Wolter, and M. Zakharyaschev. $\mathcal{E}$-connections of abstract description systems. *Artificial Intelligence*, 156(1):1–73, 2004.
14. U. Leonhardt. *Supporting Location Awareness in Open Distributed Systems*. PhD thesis, Imperial College, London, UK, 1998.
15. G. Ligozat. Reasoning about cardinal directions. *Journal of Visual Languages and Computing*, 9:23–44, 1998.
16. Y. Oh, C. Shin, S. Jang, and W. Woo. ubi-UCAM 2.0: A unified context-aware application model for ubiquitous computing environments. In *UbiCNS*, 2005.
17. D. Randell, Z. Cui, and A. Cohn. A spatial logic based on region and connection. In *Third International Conference on Knowledge Representation and Reasoning*, pages 165–176. Morgan Kaufmann, 1992.
18. A. Ranganathan, R. E. McGrath, R. H. Campbell, and M. D. Mickunas. Ontologies in a pervasive computing environment. In *Workshop on Ontologies and Distributed Systems (part of IJCAI)*, 2003.
19. J. Renz. Qualitative spatial and temporal reasoning: Efficient algorithms for everyone. In *Twentieth International Joint Conference on Artificial Intelligence*, pages 526–531, 2007.
20. B. N. Schilit, N. I. Adams, and R. Want. Context-aware computing applications. In *Workshop on Mobile Computing Systems and Applications*, pages 85–90. IEEE Computer Society, 1994.
21. A. Schmidt, M. Beigl, and H.-W. Gellersen. There is more to context than location. *Computers and Graphics*, 23(6):893–901, 1999.
22. T. Strang, C. Linnhoff-Popien, and K. Frank. CoOL: A context ontology language to enable contextual interoperability. In J.-B. Stefani, I. M. Demeure, and D. Hagimont, editors, *4th International Conference on Distributed Applications and Interoperable Systems*, pages 236–247. Springer, 2003.