A Survey of Context Modelling and Reasoning Techniques

Claudio Bettini [a,*] Oliver Brdiczka [b] Karen Henricksen [c]
Jadwiga Indulska [d] Daniela Nicklas [e] Anand Ranganathan [f]
Daniele Riboni [a]

[a] *DICo, Università di Milano, Italy*

[b] *Palo Alto Research Center (PARC), Palo Alto, CA, USA*

[c] *NICTA, Australia*

[d] *The University of Queensland, School of Information Technology and Electrical Engineering, Australia*

[e] *Department für Informatik, Carl von Ossietzky Universität Oldenburg, Germany*

[f] *IBM TJ Watson Research Center, Hawthorne, NY, USA*

Abstract

Development of context-aware applications is inherently complex. These applications adapt to changing context information: physical context, computational context, and user context/tasks. Context information is gathered from a variety of sources that differ in the quality of information they produce and that are often failure prone. The pervasive computing community increasingly understands that developing context-aware applications should be supported by adequate context information modelling and reasoning techniques. These techniques reduce the complexity of context-aware applications and improve their maintainability and evolvability. In this paper we discuss the requirements that context modelling and reasoning techniques should meet, including the modelling of a variety of context information types and their relationships, of high-level context abstractions describing real world situations using context information facts, of histories of context information, and of uncertainty of context information. This discussion is followed by a description and comparison of current context modelling and reasoning techniques and a lesson learned from this comparison.

Key words: context modelling, context reasoning, context management, quality of context, situation modelling

* corresponding author
Email address: bettini@dico.unimi.it (Claudio Bettini).

1 Introduction

There is a growing body of research on the use of context-awareness as a technique for developing pervasive computing applications that are flexible, adaptable, and capable of acting autonomously on behalf of users. A large part of this research investigates approaches to modelling context information used by context-aware applications and reasoning techniques for context information. The pervasive computing community increasingly understands benefits of formal context information modelling. First of all, due to the inherent complexity of context-aware applications, the development should be supported by adequate software engineering methods. The overall goal is to develop evolvable context-aware applications. Therefore the design of the general functions of such applications should not be intertwined with the definition and evaluation of context information, which is often subject to change. A good context information modelling formalism reduces the complexity of context-aware applications and improves their maintainability and evolvability. In addition, since gathering, evaluating and maintaining context information is expensive, re-use and sharing of context information between context-aware applications should be considered from the beginning. The existence of well-designed context information models eases the development and deployment of future applications. Moreover, a formal representation of context data within a model is necessary for consistency checking, as well as to ensure that sound reasoning is performed on context data.

Existing approaches to context information modelling—or context modelling as they are often referred to—differ in the ease with which real world concepts can be captured by software engineers, in the expressive power of the context information models, in the support they can provide for reasoning about context information, in the computational performance of the reasoning, and in the scalability of the context information management. The goal of this paper is to show the state-of-the-art in context modelling, management and reasoning in pervasive computing. We discuss the current approaches and show the lessons learned from the context models and their context management and reasoning systems.

The paper structure is as follows. Section 2 shows the requirements that need to be taken into account when modelling context information and provides a brief overview of the evolution of context models. The section also provides our justification for the selection of three prominent approaches to context modelling (object-role based, spatial models, ontology based) for a detailed description and evaluation. Sections 3 – 5 describe these three approaches to context modelling and reasoning showing how they meet the context modelling requirements. Section 6 discusses high level context abstractions that can model real world situations. Modelling real situations may require processing of context facts and reasoning upon them to attain a form of context information that is appropriate for use by context-aware applications. While the models presented in Sections 3 – 5 may have their own modelling approach to such high-level context abstractions the variety of ways these abstractions can be modelled warrants a thorough discussion of the topic. Section 7 addresses the issue of context information uncertainty. Adaptation decisions in context-aware applications are made based on evaluation of context information that can be erroneous, imprecise or conflicting. Therefore modelling of quality of context information and reasoning on context uncertainty is a very important feature of context modelling and reasoning and also warrants a separate thorough discussion. The three selected context modelling and reasoning approaches address many of the context modelling requirements; however none of them fulfils all the requirements for a generic context information modelling and reasoning approach. Section 8 presents the research on hybrid context models as a lesson learned from the context modelling, management and reasoning approaches. Section 9 concludes the paper.

2 Evolution of context modelling and reasoning

A number of context modelling and reasoning approaches have been developed over the last decade ranging from very simple early models to the current state-of-the-art context models. The research on the models was accompanied by development of context management systems that were able to gather, manage, evaluate and disseminate context information. A large number of context-aware applications based on various context models have been developed over the years for a variety of application domains. These experiences with the variety of applications influenced the set of the requirements defined for context modelling and reasoning, and therefore also influenced the research on context information models that have high expressive power, can support reasoning about context, and have good computational performance of the reasoning. In this paper we aim to describe and evaluate the state-of-the-art context models that are generic (i.e., suitable for any kind of application) and are able to meet most of the requirements set for the context modelling, management and reasoning. In this section we describe the requirements defined for context modelling, management and reasoning and show various stages of evolution of context modelling.

2.1 Requirements

We start with the description of the requirements set for context models and their context management systems.

Heterogeneity and mobility: Context information models have to deal with a large variety of context information sources that differ in their update rate and their semantic level. Some context information is sensed. Sensors can observe certain states of the physical world and provide fast and near real-time access, while providing rather raw data (like a GPS position or a camera stream) that has to be interpreted before being usable by applications. Information provided by the user—like user profiles —is updated more rarely and in general does not need additional interpretation. Context data can also be derived from existing context information. Context data obtained from databases or digital libraries—like geographic map data—is often static. A context model should be able to express those different types of context information and the context management system should provide management of the information depending on its type. Many context-aware applications are also mobile (i.e., running on a mobile device) or depend on mobile context information sources (e.g., mobile sensors). This adds to the problem of heterogeneity, as the context information provisioning must be adaptable to the changing environment. In addition, location and spatial layout of the context information play important roles due to this requirement.

Relationships and dependencies: There exist various relationships between types of context information that have to be captured to ensure correct behaviour of the applications. One such relationship is dependency whereby context information entities/facts may depend on other context information entities: for example, a change to the value of one property (e.g., network bandwidth) may affect the values of other properties (e.g., remaining battery power).

Timeliness: Context-aware applications may need access to past states and future states (prognosis). Therefore, timeliness (context histories) is another feature of context information that needs to be captured by context models and managed by the context management system. The management of context histories is difficult if the number of updates is very high. It may not be feasible to store every value for future access, and therefore summarisation techniques (e.g., the aggregation of position updates to a movement function using interpolation techniques, or the use of historical synopses of data) need to be applied.

Imperfection: Due to its dynamic and heterogeneous nature, context information may be of variable quality. In fact, it may even be incorrect. Most sensors feature an inherent inaccuracy (e.g., a few metres for GPS positions), and the sensed values age if the physical world changes, so that this inaccuracy increases over time. In addition, the context information may be incomplete or conflicting with other context information. Thus, a good context modelling approach must include modelling of context information quality to support reasoning about context.

Reasoning: Context-aware applications use context information to evaluate whether there is a change to the user and/or computing environment context; taking a decision whether any adaptation to that change is necessary often requires reasoning capabilities. It is therefore important that the context modelling techniques are able to support both consistency verification of the model and context reasoning techniques. The later can be used to derive new context facts from existing context facts and/or reason about high level context abstractions that model real world situations. The reasoning techniques should be computationally efficient.

Usability of modelling formalisms: Context information models are created by designers of context-aware applications and are also used by the context management systems and context-aware applications to manipulate context information. Therefore the important features of modelling formalisms are the ease with which designers can translate real world concepts to the modelling constructs and the ease with which the applications can at runtime use and manipulate context information.

Efficient context provisioning: Efficient access to context information is needed which can be a difficult requirement to meet in the presence of large models and numerous data objects. To select the relevant objects, attributes for suitable access paths have to be represented in the context modelling. These access paths represent dimensions along which applications often select context information, typically supported by indexes. These dimensions are often referred to as primary context, in contrast to secondary context, which is accessed using the primary context. Commonly used primary context attributes are the identity of context objects, location, object type, time, or activity of user. Since the choice of primary context attributes is application-dependent, given an application domain, a certain set of primary context attributes is used to build up efficient access paths (e.g., spatial indexes if location is a primary context).

2.2 Early approaches: key-value and markup models

Key-value models use simple key-value pairs to define the list of attributes and their values describing context information used by context-aware applications. Markup based context information models use a variety of markup languages including XML. The W3C standard for description of mobile devices, *Composite Capabilities / Preference Profile (CC/PP)* [42], is probably the first context modelling approach to use Resource Description Framework (RDF) and to include elementary constraints and relationships between context types. CC/PP can be considered a representative both of the class of key-value models and of markup models, since it is based on RDF syntax to store key-value pairs under appropriate tags. Simple kinds of reasoning over the elementary constraints and relationships of CC/PP can be performed with special purpose reasoners. CC/PP as well as other key-value and markup based context information models have been already described and evaluated in literature surveys and their limitations have been shown in [41,73,47]. The main critics of these approaches concern their limited capabilities in: (i) capturing a variety of context types, (ii) capturing relationships, dependencies, timeliness, and quality of context information, (iii) allowing consistency checking, and (iv) supporting reasoning on context, on context uncertainty and on higher context abstractions.

2.3 Domain-focused modelling

There is a body of work in various application domains on types of context information that can significantly enhance the functionalities of domain-specific context-aware applications. A relevant example of such work is the W4 context model, and its supporting infrastructure developed for context-aware browsing [11]. This model supports the representation of context as (Who, What, Where, When) Linda-like tuples and provides an interface to store and query such tuples. This and similar approaches are important for particular application domains. However, this survey focuses on generic context modelling techniques that address the issue of "how" context should be modelled but do not stipulate "which" context information should be modelled.

2.4 Towards more expressive modelling tools

Early approaches to context modelling, represented by CC/PP and similar approaches, do not meet many of the requirements listed in Section 2.1. Other approaches, characterized by more expressive context modelling tools, provide better solutions for some of the identified requirements. The object-role based modelling approach presented in Section 3 originated from information systems modelling to provide an easy mapping from real world context concepts to modelling constructs. The approach also uses a novel form of predicate logic to reason about high-level context abstractions and aims, in particular, to satisfy the *heterogeneity*, *timeliness*, *reasoning* and *usability* requirements. The *mobility*, *timeliness*, and *efficiency* requirements are addressed in particular by spatial context models, which are reviewed in Section 4. Ontological approaches to context modelling, reviewed in Section 5, can be considered a natural extension of CC/PP and RDF based approaches to satisfy the requirements of *heterogeneity*, *relationship*, and *reasoning*. While approaches that use full RDF to represent context can be considered in this category, in this survey we concentrate on the use of the OWL language instead, since it better supports automated reasoning. Hybrid context models, reviewed in Section 8, aim at integrating different approaches to obtain more comprehensive solutions.

3 Object-role based models of context information

Fact-based context modelling approaches, including the object-role modelling approach described in this section, originated from attempts to create sufficiently formal models of context to support query processing and reasoning, as well as to provide modelling constructs suitable for use in software engineering tasks such as analysis and design. Early context modelling approaches, such as attribute-value pairs, could not satisfy these requirements, particularly as the types of context information used by applications grew more sophisticated.

This section is concerned with context modelling approaches that have their early roots in database modelling techniques. In particular, it focuses on the Context Modelling Language (CML), which was described in a preliminary form by Henricksen et al. in 2002 [35] and refined in later publications [33,34].

3.1 CML overview

CML is based on Object-Role Modeling (ORM) [31], which was developed for conceptual modelling of databases. CML provides a graphical notation designed to support the software engineer in analysing and formally specifying the context requirements of a context-aware application. It extends ORM with modelling constructs for:

- capturing the different classes and sources of context facts discussed in section 2.1: specifically, static, sensed, derived, and user-supplied ("profiled") information;
- capturing imperfect information using quality metadata and the concept of "alternatives" for capturing conflicting assertions (such as conflicting location reports from multiple sensors) [33];
- capturing dependencies between context fact types; and
- capturing histories for certain fact types and constraints on those histories.

The formality of ORM and the CML extensions makes it possible to support a straightforward mapping from a CML-based context model to a runtime context management system that can be populated with context facts and queried by context-aware applications. Halpin [30] describes the *Rmap* procedure for transforming a conceptual schema to a relational schema, and Henricksen [32]

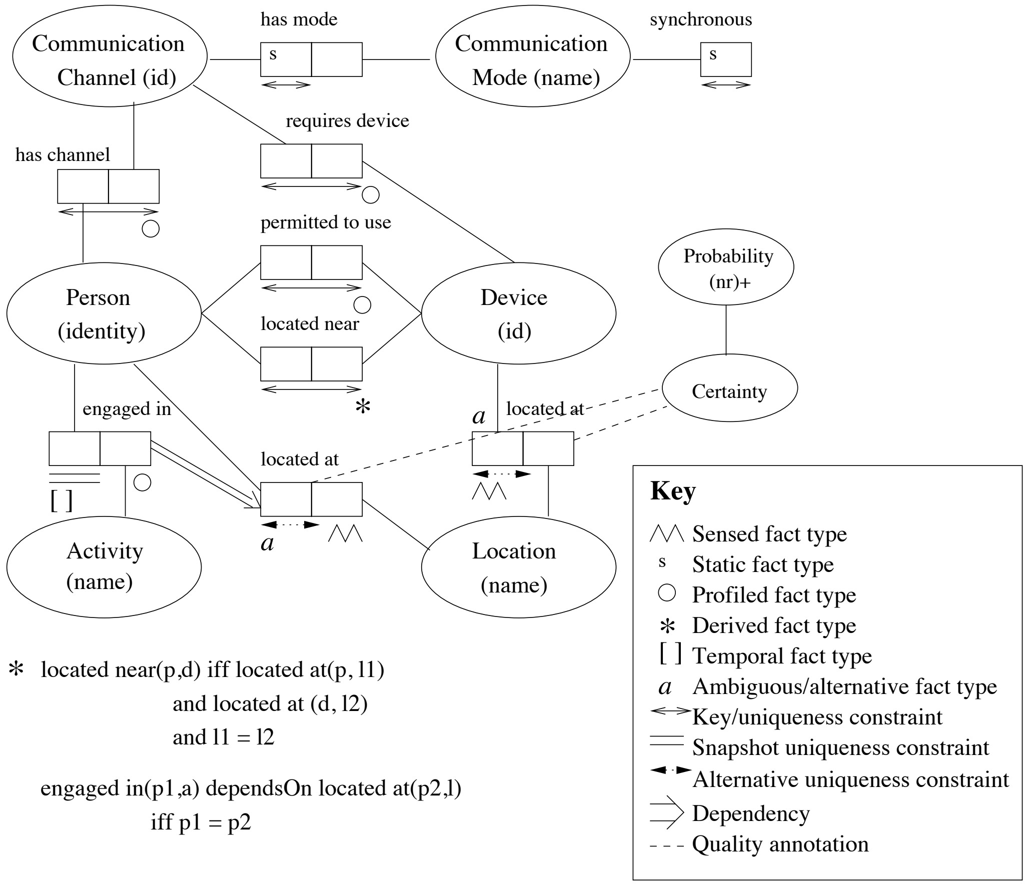

Fig. 1. An example CML model.

has developed an extension of Rmap that can be used to map a CML-based context model to a relational database. However, the formal semantics of ORM and CML can be leveraged to provide integration with other implementations such as fact-based reasoners (though it should be noted that some features of CML—particularly the constructs related to imperfect information—may not be supported).

Figure 1 illustrates the graphical notation using an example context model. This example model is designed for use by context-aware communication applications such as the one described in [34]. The model captures user activities in the form of a temporal fact type that covers past, present and future activities; associations between users, communication channels, and devices; and locations of users and devices (both absolute and relative, where the latter is represented as a derived fact type).

Each ellipsis in the figure depicts an object type—with the value in parentheses describing the representation scheme used for the object type—while each box denotes a role played by an object type within a fact type. The key summarises the remainder of the notation used in the figure. A detailed discussion of both

Table 1
Example instantiation of the "**located at**" fact type without alternatives.

Person	**Location**
Fitzwilliam Darcy	Kitchen
Elizabeth Bennet	Study

Table 2
Example instantiation of the "**located at**" fact type with alternatives.

Person	**Location**
Fitzwilliam Darcy	Kitchen
Fitzwilliam Darcy	Dining
Elizabeth Bennet	Study

the model and the software engineering process used in conjunction with CML can be found in [34].

3.2 Support for reasoning

CML leverages the formality of ORM to support the evaluation of simple assertions as well as SQL-like queries. One of the novel features of CML is its ability to support querying over uncertain information (specifically, ambiguous information represented using the "alternatives" construct) using a three-valued logic. This can be illustrated using the "**located at**" fact type from the model in Figure 1 as an example. Two possible instantiations of this fact type are shown in Tables 1 and 2. Using the three-valued logic, the assertion "Fitzwilliam Darcy **located at** Kitchen" evaluates to *true* with respect to the first instantiation and *possibly true* with respect to the second.

To evaluate more complex conditions than can be captured by assertions, Henricksen et al. define a grammar for formulating high level abstractions of context (called 'situations' in this approach), that model real world situations. These high level context abstractions are expressed using a novel form of predicate logic that balances efficient evaluation against expressive power. They are defined as named logical expressions of the form $S(v_1, ..., v_n) : \varphi$, where S is the name of the high level context abstraction, v_1 to v_n are variables, and φ is a logical expression in which the free variables correspond to the set $\{v_1, ..., v_n\}$. The logical expression combines any number of basic expressions using the logical connectives, *and*, *or* and *not*, and special forms of the universal and existential quantifiers. The permitted basic expressions are either equalities/inequalities or assertions. High level context abstractions can be incrementally combined to form more complex logical expressions. Examples and further information can be found in [34].

3.3 Evaluation

One of the main strengths of CML is its support for various stages of the software engineering process. Its graphical notation supports analysis and design of the context requirements of a context-aware application; the relational representation and grammar for high level context abstractions support run-time representation and querying. CML also provides more comprehensive support for capturing and evaluating imperfect and historical information than many of the other context modelling approaches that are currently in use.

However, CML has several weaknesses. It has a "flat" information model, in that all context types are uniformly represented as atomic facts. If a hierarchical structure is needed, or one particular dimension of context is dominant from the perspective of querying (as in spatial models, which place greater importance on location than on other types of information), then other representations may be more appropriate. CML also emphasises the development of context models for particular applications or application domains, and does not provide the support for interoperability that is found in models such as Strang et al.'s ontology-based Aspect-Scale-Context model [74]. An attempt to create a hybrid model that combines the respective advantages of CML and ontology-based approaches (including support for hierarchical structures and interoperability) is described by Henricksen et al. in [36]. The development of hybrid models is also discussed further in Section 8.

4 Spatial models of context information

Space is an important context in many context-aware applications. Most context definitions mention space as a vital factor: e.g., Schilit, Adams and Want define three important aspects of context as *"Where you are, who you are with and what resources are nearby"* [68]. Also, in the most frequently used context definition by Dey et al. [19], space can be seen as a central aspect of context entities: *"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"*—places are spatial entities, and interaction typically requires some vicinity. Thus, some context modelling approaches give space and location a preferential treatment. As we will see, space is well suited to organise and efficiently access context information. Spatial existence also serves well as an intuitive metaphor for non-physical context information (e.g., virtual information towers [44] for context-tagged web pages or Pascoe's Stick-E-Notes [58]). How people associate certain situations with location can be also seen in the most common question they ask on a mobile phone: "Where are you?" What they typically are interested in is not the exact location but the general situation of the person they are talking to.

4.1 Context information model

Most spatial context models are fact-based models (see Section 3) that organise their context information by physical location. This could be the location of the real-world entities which is described in the context information (e.g., the boundaries of a room), the location of the sensor that measures the context information, or, for non-physical context information, an associated location as metaphor (e.g., Stick-E-Notes or virtual information towers).

This location information is either pre-defined (if the entities are static), or it is obtained by positioning systems which track mobile objects and report their position to a location management system. Basically, two kinds of coordinate systems are supported by positioning systems:

Geometric coordinates: Represent points or areas in a metric space, such as the WGS84 coordinates of GPS (latitude, longitude, and elevation above sea level). Using geometric functions such as the Euclidian distance allows distance calculation and allows nearest neighbour queries. Overlaps of geometric figures can be used to specify ranges by their geometric extension and determine whether ranges are included in each other, which allows range queries.

Symbolic coordinates: Symbolic coordinates are represented by an identifier, such as a room number or the ID of a cell or access point in wireless telephone or local area networks. In contrast to geometric coordinates there is no spatial relation offered by symbolic coordinates. In order to allow spatial reasoning about inclusion (for ranges) and distances (for nearest neighbours) explicit information about the spatial relations between pairs of symbolic coordinates has to be provided. Note that this location model also is applicable if there is no explicit modelling of space but only relations between objects (as in [59]).

The semantic level of spatial context models can be discussed along the tiers of spatial ontologies proposed by A. Frank [24]:

- Tier 0 is the ontology of the physical reality. It is based on the assumption that there is exactly one real world; hence, for every property in the world and for a given point in time-space there is a single value.
- Tier 1 includes observations of reality and is the first tier that can be accessed in context models. Here, a value can be derived at a location with a given observation type. The type determines the measurement scale of the value (e.g., nominal or rational) and the measurement unit (e.g., metres or

seconds). For spatial values, a coordinate system must be given. Values normally have a limited accuracy due to observation errors. Fact-based context models are typically situated on that tier.

- In tier 2, single observations are grouped with individual objects that are defined by uniform properties. Now, the value of an observation is the state of a whole object, given by an identifier. Frank only considers physical objects in this tier, i.e., “things which exist in the physical world and can be observed by observation methods”. They have a geometric boundary in the world, but it can change over time (like dunes or fluids). Up to tier 2, the ontology tiers cover data that can be seen as objective reality—you can send out a team of geographers or students to model physical objects and they will come to an agreement about their observations. Thus, this kind of information can be easily shared between different context-aware applications.
- In tier 3, the socially constructed reality is represented. Social reality includes all objects and relations that are created by social interactions. Such properties are classified and named within the context of administrative, legal or institutional rules. Object names belong to this tier since they are assigned by culture; for many important things (but not all) there are functions to determine the name and to find the object by name in a certain environment.
- Finally, in tier 4 the rules are modelled that are used by cognitive agents (both human and software) for deduction. This tier is normally built into database query languages, applications or reasoning engines of knowledge based systems. Ontology based models of context information (see Section 5) typically cover all tiers up to this level.

Although the tiered model of Frank is just an abstract conceptualization of different (spatial) representations of the world, it is useful to distinguish between various implementations of spatial context models (as can be seen in [7]). For example, fact-based models like the Context Modelling Language of Henricksen et al. (Section 3 and [34]) cover tier 1–3, and the grammar used to define situations (as higher-level context abstractions) is located at tier 4.

The spatial context model developed in the Nexus project (called Augmented World Model [57]) is an object-based class hierarchy of context information that supports multi-inheritance (a camera can be both a **MobileObject** and a **Sensor**), multi-attributes (a **MobileObject** can have multiple instances of its position attribute that differ in their meta data, e.g., the measurement time), and both a geometric coordinate system (that supports multiple spatial reference systems) and a simple symbolic location system (based on spatial relationships). Most object classes inherit from the class **SpatialObject**, which makes the Augmented World model inherently spatial: almost all objects (real and virtual) are modelled with a location, either by their physical location or by a meaningful association metaphor (like the location of a virtual information tower for web sites). The Nexus context model was designed to be sharable between different context-aware applications in a potentially global scope and thus to be scalable to a high amount of context data [28]. In its current implementation, the managed context information only represents information of tiers 1–3. Higher-level context information like situations is managed by the applications.

In contrast to the Nexus model, the Equator project context model [52] is a typical contextual ontology that represents all tiers by an OWL class model. Its location model is a hierarchical notion of inter-connected symbolic spaces, such as **Buildings**, **Floors** and **Rooms**. Properties define spatial relations between these spaces. Although the ontology also offers coordinate features (properties that represent, e.g., a GPS location), Millard et al. states that it is very hard to perform any inference over them using a normal reasoner, as they are usually not spatially aware.

4.2 Support for reasoning

Spatial context models allow reasoning about the location and the spatial relationships of objects. Such relations cover the inclusion in a distinct area or range and the distance to other entities. According to [6], there are three typical spatial queries on spatial context information: (i) Position: retrieve the position of an object; (ii) Range: retrieve objects that are located in a spatial range; and (iii) Nearest Neighbour: retrieve a list of one or more objects which are closest to the position of an object. These queries become more challenging when the position of the object is imprecise, and given as an area.

Although these queries at first seem simple and obviously necessary for a variety of context-aware applications, their efficient processing highly depends on the underlying context information management system, which may use spatial database support or other specialised modules. Grossmann et al. [28] show how the characteristics of different types of context information can be used to design efficient management systems. The two main factors are update rate (how often a certain context information is updated—by sensors, by humans, or almost never) and usage for selection (how often a certain context information is used to restrict the set of relevant context information). The latter is often referred to as primary context, which is used to retrieve secondary context. Since many context-aware applications use space as a primary context, it is reasonable to design context management systems to efficiently support spatial queries, e.g., by managing spatial indexes.

In addition, if the amount of context information gets very large, it can be partitioned along the spatial dimension (e.g., by introducing context servers with spatial service areas).

4.3 Evaluation

Obviously, spatial context models are well suited for context-aware applications that are mainly location-based, like many mobile information systems. However, even if location is not a primary context for a context-aware application, a spatial organisation of the context information may be beneficial: if the amount of managed context information is large, spatial partitioning can be used to cope with the complexity. In particular, mobile systems can benefit from spatial context models: due to their inherent (potentially global) mobility, they are likely to need large amounts of context information in total, which can be easily preselected to relevant context information in the vicinity by using a spatial predicate. As we see later, hybrid context modelling approaches separate the fact-based context management (tier 1–3 in Frank's hierarchy) from higher level reasoning (tier 4) functions. Thus, a spatial pre-selection of relevant context information could be reasonable to speed up the reasoning process by reducing the size of the knowledge base [7,56].

A main consideration for spatial context models is the choice of the underlying location model. Geometric and geographic location models offer simple mapping to map data and GPS sensor data, while symbolic and relational location models are easier to build up and represent a simple perception of space (with relations like part-of and located-near). This choice also determines how the context information should be managed (e.g., by a spatial database), what reasoning methods and queries are available and what access paths have to be built up.

A drawback of spatial context model is the effort it takes to gather the location data of the context information and to keep it up to date. Thus, if the spatial dimension is of no importance (or only including simple spatial relationships like the meeting of two users), this effort could be saved.

5 Ontology based models of context information

Context, as intended in this paper, can be considered as a specific kind of knowledge. Thus, it is quite natural to investigate if any known framework for knowledge representation and reasoning may be appropriate for handling context. The trade-off between expressiveness and complexity of reasoning has driven most of the research in symbolic knowledge representation in the last two decades, and description logics [3] have emerged among other logic-based formalisms, mostly because they provide complete reasoning supported by optimised automatic tools. Since ontologies are essentially descriptions of concepts and their relationships, it is not surprising that the subset of the OWL language admitting automatic reasoning (i.e., OWL-DL) is indeed a description logic. Ontology-based models of context information exploit the representation and reasoning power of these logics for multiple purposes: a) the expressiveness of the language is used to describe complex context data that cannot be represented, for example, by simple languages like CC/PP [42]; b) by providing a formal semantics to context data, it becomes possible to share and/or integrate context among different sources; c) the available reasoning tools can be used both to check for consistency of the set of relationships describing a context scenario, and, more importantly, to recognise that a particular set of instances of basic context data and their relationships actually reveals the presence of a more abstract context characterisation (e.g., the user's activity can be automatically recognised). In this section, we briefly illustrate the main ontology-based context models that have been proposed, we show how reasoning is performed, and we identify current critical issues.

5.1 Context information model

The formalism of choice in ontology-based models of context information is typically OWL-DL [40] or some of its variations, since it is becoming a de-facto standard in various application domains, and it is supported by a number of reasoning services. By means of OWL-DL it is possible to model a particular domain by defining *classes*, *individuals*, characteristics of individuals (*datatype properties*), and relations between individuals (*object properties*). Complex descriptions of classes and properties can be built by composing elementary descriptions through specific operators provided by the language. For instance, given two atomic classes `Person` and `Female`, the class `Male` can be defined as:

$$\texttt{Male} \equiv \texttt{Person} \sqcap \neg\texttt{Female}.$$

More complex definitions can be obtained by using operators such as *property restrictions* that can force all/some values of a certain property to belong to a given class, or can force a property to have at least k values.

Hence, complex context data, intended as those data that can be inferred by means of reasoning tasks on the basis of raw data directly acquired from sensors, and other complex context data, can be represented by structured OWL-DL expressions. These data typically include information regarding the sociocultural environment of users, complex user preferences regarding the adaptation of services, and activities. For example, the following definition (taken by the ontology used within the *CARE* framework [1]) is used to describe `BusinessMeeting` as including any activity performed in a conference room within a company building, and having at least two actors, each of which is an employee: [1]

$$\begin{aligned}\mathtt{BusinessMeeting} \sqsubseteq \mathtt{Activity} \sqcap \geq 2\,\mathtt{hasActor} \sqcap \\ \forall\,\mathtt{hasActor.Employee} \sqcap \\ \exists\,\mathtt{hasLocation.(ConfRoom} \sqcap \mathtt{CompanyBuilding)}\end{aligned}$$

In addition to providing an expressive formalism for representing complex context data, ontologies are well-suited for knowledge sharing since they provide a formal specification of the semantics of context data. We point out that this feature is particularly important in mobile and pervasive environments, in which different heterogeneous and distributed entities must interact for exchanging users' context information. To this end, various OWL ontologies have been proposed for representing shared descriptions of context data. Among the most prominent proposals are the *SOUPA* [15] ontology for modelling context in pervasive environments, and the *CONON* [79] ontology for smart home environments. Those shared ontologies can be integrated with application-specific models of context by means of extensions of the OWL language, such as the one proposed in [9].

OWL-DL ontological models of context have been adopted in several architectures for context-awareness; among the others, we recall the Context Broker Architecture (CoBrA) [14] and the SOCAM [29] middleware, that adopt the SOUPA and CONON ontologies, respectively. The DAML+OIL ontology language (a predecessor of OWL) is the basis of the context model of the GAIA [62] middleware for active spaces. In GAIA, reasoning for deriving new context data is performed by means of rule-based inferencing and statistical learning; ontologies are used to provide a clear semantics to data derived through different reasoning techniques. Finally, some architectures for context-awareness (e.g., the *semantic eWallet* [25]) have adopted more expressive ontology languages obtained by extending OWL-DL with rules.

5.2 Support for reasoning

A further benefit of ontologies with respect to simpler representation formalisms consists in the support of reasoning tasks. Indeed, on the basis of the asserted knowledge it is possible to *i)* automatically derive new knowledge

[1] Here we use DL syntax, but an equivalent OWL description can be easily obtained.

about the current context, and *ii)* detect possible inconsistencies in the context information. With respect to *i)*, ontological reasoning can be executed for inferring new context information based on the defined classes and properties, and on the individual objects retrieved from sensors and other context sources. For instance, it is possible to derive the set of individual objects that are related to a given one by a particular property (e.g., the set of activities taking place in a specific location), or to calculate the most specific class an individual object belongs to (e.g., the fact that the activity performed by a given employee is a business meeting). With respect to *ii)*, we point out that consistency checking is crucial in the definition of an ontology, as well as in its population by new instances. Hence, automatic consistency checking can be performed to capture possible inconsistencies in the definition of the classes and properties of the ontology (e.g., a class being a subclass of two disjoint classes), or in its population (e.g., a person being in different rooms at the same time).

5.3 Evaluation

With respect to simpler approaches (e.g., key-value and markup models), ontological models of context provide clear advantages both in terms of heterogeneity and interoperability. Considering usability issues, we point out that user-friendly graphical tools exist (e.g., *Protégé*[2]) that make the design of ontological context models viable also to developers that are not particularly familiar with description logics. However, with respect to timeliness we note that at the time of writing there is very little support for modelling temporal aspects in ontologies. Moreover, despite the ability to express relations and dependencies among context data makes the ontological model a satisfactory solution for a wide range of context-aware applications, experiences with the development of context ontologies show that the operators provided by OWL-DL are sometimes inadequate to define complex context descriptions (see, e.g., [2]). This problem is due to the fact that the constructors included in the OWL-DL language were chosen in order to guarantee decidable reasoning procedures. For this reason, OWL-DL does not include very expressive constructors that would be helpful for modelling complex domains, such as users' activities.

Consider for example the `isColleagueOf` property, which is very useful for modelling activities performed within an organisation. A straightforward definition of that property can be given by composing the atomic properties `isEmployedBy` and `isEmployerOf`:

$$\texttt{isColleagueOf} \equiv \texttt{isEmployedBy} \circ \texttt{isEmployerOf}$$

[2] `http://protege.stanford.edu/`

Indeed, if a person a is employed by a person b that is the employer of c, then a is colleague of c. Unfortunately, this definition cannot be expressed in OWL-DL. In fact, the language – in order to preserve its decidability – does not include a constructor for composing relations. Similarly, OWL-DL does not include some expressive class constructors, such as the ones that restrict the membership to a class only to those individual objects that are fillers of two or more properties (these constructors are called *role-value-maps* in the literature). Formally, a role-value map $R \subseteq S$ defines the class of individuals i such that the individuals related to i by property R are related to i also by property S (R and S can be composed properties). For example, given a property `isCoactorOf` that relates individuals performing an activity together, the role-value-map (`isCoactorOf` $\subseteq$ `isColleagueOf`) defines the class of individuals having as coactors only persons that are their colleagues. If for the sake of simplicity one assumes than an individual cannot perform more than one activity at a time, a more precise definition of *BusinessMeeting* could be given by substituting `Employee` in that definition with (`isCoactorOf` $\subseteq$ `isColleagueOf`).

Even if proposed extensions of OWL promise to overcome part of these limitations (in particular, by including a restricted form of property composition [54]), at the time of writing the definition of some context domains with OWL-DL can be problematic. Hence, the possibility of augmenting the expressivity of ontological languages through an extension with rules has been recently investigated by the Semantic Web community, and brought to the definition of logic languages such as SWRL [39], adopted for example in [13]. These rule extensions are not really hybrid approaches since rules are fully integrated in ontological reasoning. The main problem with this approach is that reasoning in OWL-DL is already computationally expensive, as described in the following paragraph, and the proper integration of rules makes the resulting language undecidable. A further research issue consists in extending existing ontological languages to support fuzziness and uncertainty while retaining decidability (see, e.g., [72,21]).

In addition to the above mentioned expressiveness limitations, ontological reasoning with OWL-DL also poses serious performance issues. Indeed, a natural solution for deriving complex context data through ontological reasoning is to perform the *realisation* of an individual of interest (e.g., the individual representing the user's *Activity*) in order to find the most specific class the individual belongs to (e.g., a *BusinessMeeting*). Unfortunately, the realisation problem has `NExpTime` complexity. One could argue that this is a worst-case complexity, and that current optimised reasoners can still be practical for many applications. However, performance issues when reasoning with OWL-DL are confirmed by experimental evaluations with different ontology-based context reasoning architectures (see, e.g., [75,1]). Hence, online execution of ontological reasoning poses scalability issues, especially when the ontology is populated by a large number of individuals. In order to improve the efficiency of reasoning with OWL-DL, various optimisations based on the use of relational database techniques have been recently proposed. A well-known proposal in this sense is the *InstanceStore* system [38]. However, at the time of writing, InstanceStore has some limitations that are critical for reasoning with context data. Indeed, it does not allow the instantiation of relations between individuals. In some cases, efficiency problems can be avoided by executing particularly onerous tasks asynchronously with respect to the service requests. Details about these optimisations are reported in [1].

6 High-level context abstractions

Information from physical sensors, called low-level context and acquired without any further interpretation, can be meaningless, trivial, vulnerable to small changes, or uncertain [77]. Schilit et al. [68] observed hence that context encompasses more than just the user's location, because other things of interest, including the user's social situation, are also changing. The limitation of low-level contextual cues when modelling human interactions and behaviour risks reducing the usefulness of context-aware applications. A way to alleviate this problem is the derivation of higher-level context information from raw sensor values, called context reasoning and interpretation. The idea is to abstract from low-level context by creating a new model layer that gets the sensor perceptions as input and generates or triggers system actions. In the literature, different notions have been employed to refer to this higher-level context layer. Situational context [26] and situation [19] [22] are the most common ones. The notion of situation is used as a higher-level concept for a state representation. Initially, the term "situation" was used in linguistics and natural language semantics. In 1980, Barwise and Perry wrote in their paper The Situation Underground [4] of situations:

> "The world consists not just of objects, or of objects, properties and relations, but of objects having properties and standing in relations to one another. And there are parts of the world, clearly recognised (although not precisely individuated) in common sense and human language. These parts of the world are called situations. Events and episodes are situations in time, scenes are visually perceived situations, changes are sequences of situations, and facts are situations enriched (or polluted) by language."

In context-aware applications, situations are external semantic interpretations of low-level context [22], permitting a higher-level specification of human behaviour in the scene and the corresponding system services. Situations inject meaning into the application and are more stable, and easier to define and maintain than basic contextual cues. Adaptations in context-aware applications are then caused by the change of situations (i.e., a change of a context value triggers adaptation if the context update changes the situation). Design and implementation of the applications become much easier with situations because the designer/programmer can operate at a high level of abstraction (situation) not on all context cues that create the situation. For example, [46] describes six different ways to specify the situation *in_meeting_now* based on contextual cues:

- co-location of people and agenda information
- co-location of filled coffee cups in a room
- weight sensors on the floor
- devices in the room (lights, projector, PowerPoint on PC)
- sounds and noises
- cameras ("watch" meeting room for activity)

In each case, the situation *in_meeting_now* remains stable and appropriate system actions can simply be associated to this situation, while the contextual cues regarding this situation may change. Additional contextual cues relevant to this situation can be added or obsolete ones can be removed without changing the situation itself, but by modifying only its specification.

Figure 2 summarizes the basic ideas of this section. Sensor-based low-level context information is semantically interpreted by the high-level context layer. Situations abstract from low-level data and are reusable in different environments and applications. Relationships defined between situations can provide for a further abstraction and limitation of complexity.

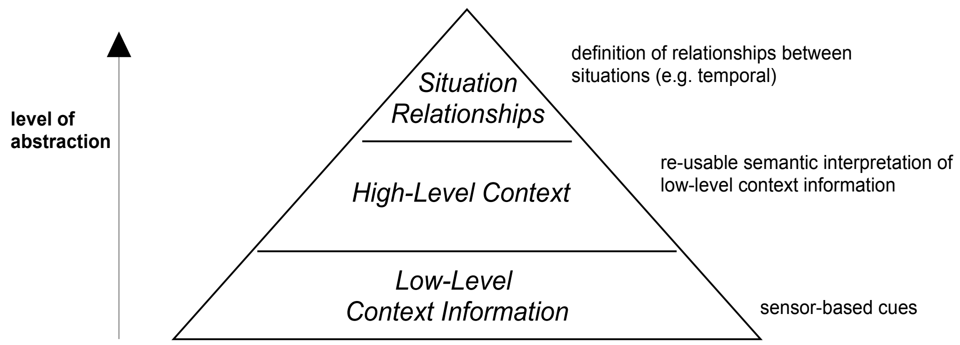

Fig. 2. Overview of the different layers of semantic context interpretation and abstraction

As situations are semantic abstractions from low-level contextual cues, human knowledge and interpretation of the world must be integrated into a model or situation representation. This can either be done during a specification process, i.e., a human defines the situations and their relationship based on his knowledge, or situations are recognised and learned automatically, i.e., sensor perceptions are aggregated and associated to a human-defined situation label using machine learning techniques.

The latter relates to the domain of human activity recognition. Most approaches in this area focus on the classification of basic human activities or scenarios without considering a richer contextual description (e.g., [55]). Some recent work, however, attempts to acquire high-level contextual models involving situations. Clarkson [16] proposes a wearable system capable of distinguishing coarse locations and user situations. Different locations and situations of an individual user like “home”, “at work”, “bathroom” or “restaurant” are isolated and then recognised based on a clustering of video and audio data recordings. McCowan et al. [51] go further by proposing a two-layered framework for modelling and recognising individual and group actions in meetings. A first layer detects individual actions like “writing” or “speaking” from individual audio and video recordings. The second group layer fuses the individual output of the first layer as well as group audio and video features (coming from projector screen and white-board). The output of the second layer are group situations like “discussion”, “monologue”, “note-taking”, or “presentation”. Brdiczka [10] finally proposes a four-layered situation learning framework. This framework acquires different parts of a situation model, namely situations and roles, with different levels of supervision. Situations like “presentation”, or “aperitif” and roles played by individuals like “lying down” or “sitting and gesturing” are learned from individual audio and video data streams. Depending on number and kind of observations provided for recognition and situations to be recognised, these learning-based approaches have a correct recognition rate of situations between 85% and 100% (95% in [16], 88.8% in [51], and 100% (with presegmentation) in [10]).

However, these approaches require an important training period during which several examples of each situation and related concepts are collected and analysed. This training phase often needs as much human intervention (e.g., for semantic labelling) as a manual situation specification phase would require. The granularity of the learned concepts is further influenced by the character and availability of the low-level sensor data. For example, if an application needs a finer granularity of the situation “meeting”, e.g., a “conference meeting” situation, we would need at least a recording of one or two conferences (which may only take place once a year). Finally, machine learning methods choose a tradeoff between generalisation and specification when acquiring concepts from sensor data recordings, which does not always meet the correct semantics, hence resulting in wrong detections of situations.

When contextual cues and application needs of situations perception are known in advance, a human can specify the situations manually. In context-aware computing, most approaches for manual situation specification refer to Dey's context definition [19] as "any information that can be used to characterise the situation of an entity". An entity can be a person, place or object considered relevant to user and application, including the user and the application themselves. Dey defines as situation further as the "description of the states of relevant entities". A situation is thus a temporal state within context. Early approaches use formal logics to describe and represent these states. A first representative of this kind is the Situation Theory proposed by Barwise and Perry [5]. Situation theory tries to cover model-theoretic semantics of natural language in a formal logic system. The situation calculus [64] further provides a logical language for reasoning about action and change. Changing scenarios are represented as a set of second-order logic formulae.

Even though approaches based on formal logics provide a high level of abstraction and formality for specifying the situations, they are error-prone in the domain of context-aware computing due to the incompleteness and ambiguity of contextual cues and information. Limited reasoning performance further reduces the scalability of these approaches in real-world applications. To cope with this, some approaches try to balance efficient evaluation and expressive power (e.g., the grammar for formulating situations described in section 3). Assertions that are interpreted under a closed-world assumption are used to reduce the values in quantified expressions describing situations. Crowley et al. [17] introduce the concepts of role and relation in order to characterise a situation. Roles involve only one entity, describing its activity. An entity is observed to "play" a role. Relations are defined as predicate functions on several entities, describing the relationship or interaction between entities playing roles. This model is less formal (even though a formal definition of the concepts is provided [65]) and highlights the application viewpoint by proposing different implementations for the situations [63].

6.2 Relationships between situations

While many approaches only focus on defining and recognising situations, some approaches also specify and model situation relationships. One motivation is to considerably reduce the search space for potential situations to be recognised, once the actual situation is known and knowing possible relationships (e.g., knowing possible successor situations of the current situation).

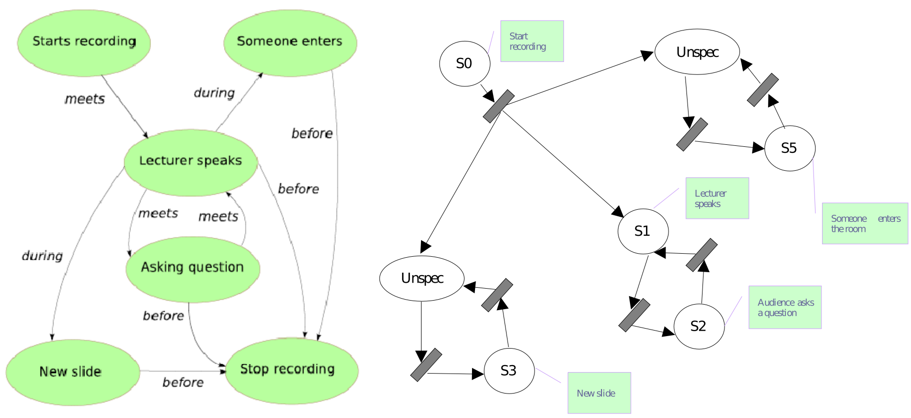

Fig. 3. Temporal Situation Model of the Automatic Cameraman system [63] that proactively chooses the viewpoint for recording a lecture (left) and compiled Petri net (right) (pictures from [63])

The state aspect of situations is often emphasised by the constraint that at least one situation must be active at a time. This can provide more stability and better performance, but requires a complete (exhaustive) situation model for the context-aware application. All potential situations, their relationships and transitions must be included in this model, which is not always possible, in particular in informal settings and scenarios. In [63], a situation model is defined in order to provide an automatic cameraman service that proactively chooses the viewpoint for recording a lecture (Figure 3 left). Situations relationships are represented by Allen's temporal logic. For execution, the temporal relations are automatically compiled into a synchronised Petri net that takes contextual changes as input to trigger the situation transitions, while one or more places of the Petri net represent the intrinsic situations (Figure 3 right). Even though the temporal relations implemented by a Petri net provide high stability and good performance of the context-aware application, only a limited number of situations in a rather formal setting (e.g., lecture) can be covered.

7 Uncertainty of context information

Both the physical world itself, and our measurements of it are prone to uncertainty. Hence, one of the key requirements of context-awareness is capturing and making sense of imprecise, and sometimes conflicting data, about the physical world.

Different types of entities (or software objects) in the environment must be able to reason about uncertainty. These include entities that sense uncertain contexts, entities that infer other uncertain contexts from these basic, sensed contexts, and applications that adapt how they behave on the basis of uncertain contexts. Having a common model of uncertainty that is used by all entities in the environment makes it easier for developers to build new services and applications in such environments and to reuse various ways of handling uncertainty.

7.1 Models for uncertainty

There has been work in addressing the problem of representing, reasoning about and overcoming uncertainty in context information. Hui Lei et al. describe a context service that allows context information to be associated with quality metrics, such as freshness and confidence [43]. Castro et al. use Bayesian networks for sensor fusion [12], in particular considering location information. Schmidt et al. associate each of their context values with a certainty measure that captures the likelihood that the value accurately reflects reality [70]. Gray and Salber include information quality as a type of meta-information in their context model, and describe six quality attributes: coverage, resolution, accuracy, repeatability, frequency and timeliness [27]. The model described by Henricksen et al. [35] supports quality by allowing associations between objects to be annotated with a number of quality parameters, which capture the dimensions of quality considered relevant to that association. Dey et al. suggest a mechanism for overcoming uncertainty whereby ambiguous information can be resolved by a mediation process involving the user [20]. This solution is particularly viable when the context information under consideration is small in volume and doesn't change rapidly, so that the user is not unreasonably burdened. Rene Mayrhofer [50] and Albrecht Schmidt [69] propose approaches for inferring and predicting context information from sensor data in a bottom-up manner. Hightower et al [37] proposed the Location Stack, where they argued that uncertainty must be preserved. The Location Stack has different layers for helping manage uncertainty in location. For example, a "Fusion" layer is in charge of continually merging streams of measurements into a time-stamped probabilistic representation of the positions and orientations of objects. The "Arrangement" layer contains an engine for probabilistically reasoning about the relationships (e.g. proximity, containment, geometric formations) between two or more objects.

Ranganathan et al. [61] provide a categorisation of different kinds of quality metrics that can specifically be associated with location information obtained from different kinds of sensors. These metrics are:

(1) *Resolution*, which is the region that the sensor says the mobile object is in. Resolution can be expressed either as a distance or as a symbolic location, depending on the kind of sensor. Sensors like RF badges or GPS devices give resolution in terms of distance. For example, some GPS devices have

a resolution of 50 feet, which means that the object lies within a circle of 50 feet from the location given. Other sensors such as card-readers give resolution in terms of a symbolic location, like a room. For example, a card reader says that a person is somewhere inside a room.

(2) *Confidence*, which is measured as the probability that the person is actually within a certain area returned by the sensor. This probability is calculated based on which sensors can detect the person in the area of interest.

(3) *Freshness*, which is measured based on the time that has elapsed since the sensor reading. All sensor readings have an expiry time, beyond which the reading is no longer valid.

Ranganathan et al. [60] also developed an uncertainty model based on a predicate representation of contexts, where each context predicate is associated with a confidence value. The confidence value associated with a predicate measures the probability (in the case of probabilistic approaches) or the membership value (in the case of fuzzy logic) of the event corresponding to the context predicate being true. For example, prob(location(carol, in, room 3233)) =0.5 means that the probability that Carol is in Room 3233 is 0.5. This model forms the basis for reasoning about uncertainty using various mechanisms such as probabilistic logic, fuzzy logic and Bayesian networks. They incorporated these reasoning mechanisms in Gaia [67], their distributed middleware system for enabling Active Spaces.

7.2 Reasoning on uncertainty

A number of mechanisms have been proposed in the literature for reasoning on uncertainty. Broadly, there are two main purposes for reasoning on uncertainty : improving the quality of context information, and inferring new kinds of context information. Reasoning to improve the quality of context information typically takes the form of multi-sensor fusion where data from different sensors are used to increase confidence, resolution or any other context quality metrics. Reasoning for the purpose of inferring new context information typically takes the form of deducing higher level contexts or situations (like the activity of a user) from lower level contexts (like the location or instant messaging status of the user). Since we cannot directly sense the higher level contexts, these contexts may be associated with a certain level of uncertainty, depending on both the accuracy of the sensed information and precision of the deduction process.

Different approaches have been used for reasoning on uncertain context information. In this paper, we describe some of these approaches: fuzzy logic, probabilistic logic, Bayesian networks, Hidden Markov models, and Dempster-Shafer theory of evidence.

Fuzzy Logic. In fuzzy logic [78], confidence values represent degrees of membership rather than probability. Fuzzy logic is useful in capturing and representing imprecise notions such as “tall”, “trustworthy”, and “confidence” and reasoning about them. The elements of two or more fuzzy sets can be combined (fused) to create a new fuzzy set with its own membership function. Examples of fusion operations are intersection, union, complement, and modification. Fuzzy logic is well suited for describing subjective contexts, performing multi-sensor fusion of these subjective contexts and resolving potential conflicts between different contexts.

Probabilistic Logic. Probabilistic logic allows making logical assertions that are associated with a probability. One such logic, based on proposition-logic, was proposed by Fagin et al [23]. They specify a complete axiomatisation, and also show that the complexity of deciding satisfiability in their logic is no worse than that of propositional logic. This logic lets us make statements such as “the probability of E is less than 1/3” and “the probability of E is at least twice the probability of F,” where E and F are arbitrary events. Probabilistic logic lets us write rules that reason about events’ probabilities in terms of the probabilities of other related events. These rules can be used both for improving the quality of context information through multi-sensor fusion as well as for deriving higher level probabilistic contexts. The rules can also be used for resolving conflicts between context information obtained from different sources (such as when different location sensing modalities give different locations for the same entity). Various rule engines like Prolog can then be used to reason on these rules. Ranganathan et al [60] have used such rules for encoding access control policies.

Bayesian Networks. Bayesian networks are directed acyclic graphs, where the nodes are random variables representing various events and the arcs between nodes represent causal relationships. The main property of a Bayesian network is that the joint distribution of a set of variables can be written as the product of the local distributions of the corresponding nodes and their parents. Bayesian networks are particularly efficient in representing and storing conditional probabilities, if the dependencies in the joint distribution are sparse. Examples of use of Bayesian networks are in location sensor fusion [12] and in diagnosing the source of faults in pervasive computing environments [60]. In general, Bayesian networks are well suited for combining uncertain information from a large number of sources and deducing higher-level contexts.

Hidden Markov Models. A Hidden Markov Model (HMM) represents stochastic sequences as Markov chains; the states are not directly observed, but are associated with observable evidences, called emissions, and their occurrence probabilities depend on the hidden states. These models have been used for location prediction. For example, [45] use a hierarchical Markov model that can learn and infer a user's daily movements through an urban community. The model uses multiple levels of abstraction in order to bridge the gap between raw GPS sensor measurements and high level information such as a user's destination and mode of transportation.

Dempster Shafer Theory. The Dempster-Shafer theory is a mathematical theory of evidence [71] based on belief functions and plausible reasoning, which is used to combine separate pieces of information (evidence) to calculate the probability of an event. It is often used as a method of sensor fusion, by obtaining degrees of belief for one question from subjective probabilities for a related question, and then combining such degrees of belief when they are based on independent items of evidence. The belief in a hypothesis is constituted by the sum of the masses of all sets enclosed by it (i.e., the sum of the masses of all subsets of the hypothesis). This reasoning approach has been used by Wu to deal with uncertainty associated with context sensing [76]. In his implementation, an Aggregator receives video and audio features from a camera and a set of microphone widgets to determine the likelihood of a participant's focus of attention in a meeting.

Dargie [18] has proposed a conceptual architecture where different reasoning mechanisms can be incorporated in a unified manner for acquiring, aggregating and reasoning about context information. In this architecture, different reasoning mechanisms are used in different scopes: fuzzy logic may be used for defining the conceptual states of a primitive context to enable human-like reasoning; Dempster-Shafer Theory for combining the independent observations of multiple sensors each of which observes one and the same phenomenon; and Hidden Markov Models and Bayesian Networks for actually computing a higher-level context.

8 Hybrid context models

In this section we investigate context modelling approaches that try to integrate different models and different types of reasoning in order to obtain more flexible and general systems. We first discuss some limitations of previously presented models, arguing that they may benefit from the integration with others. Then, we illustrate some existing approaches in this direction, and finally we provide general ideas on how a more comprehensive hybrid model may be designed.

	Object-role	Spatial	Ontological
Heterogeneity	+	∼	+
Mobility	∼	+	−
Relationships	∼	∼	+
Timeliness	+	+	−
Imperfection	∼	∼	−
Reasoning	∼	−	+
Usability	+	∼	∼
Efficiency	∼	+	−

Fig. 4. A comparison of context modelling approaches

8.1 Why hybrid models are needed

The previous sections have illustrated the main approaches for context modelling and reasoning that can be found in the literature. As shown in Figure 4, none of them can satisfy all the requirements described in the introduction. Partial satisfaction of the requirements is shown as '∼' in the figure.

Spatial models provide efficient procedures for the execution of typical spatial queries; however, they do not always cope with the uncertainty of actual location readings. Moreover, interoperability among different spatial models can be easily achieved when the location information is confined to very simple spatial data (e.g., points in the space represented by their coordinates in the WGS 84 standard); if more complex spatial domains are to be modelled, interoperability can be obtained only by adopting expressive languages (e.g., coupling the different models with a shared ontology of location).

With regard to fact-based models, the CML language has advantages in its support for software engineering. It captures the heterogeneity of the context information sources, histories (timeliness) of context information and provides an easy mapping from real world concepts into modelling constructs. It also provides a good balance between expressive power and efficient reasoning procedures for evaluation of simple assertions about context and for reasoning about high-level context abstractions (called 'situations' in that approach) expressed as a form of predicate logic. Indeed, the predicate logic supported by CML is well suited for expressing dynamic context abstractions. However, in order to preserve efficiency, that language is less expressive than ontological languages like OWL-DL. A possible shortcoming of CML with respect to more expressive languages is the lack of support for hierarchical context descriptions. Moreover, even if CML supports queries over uncertain information through a three-valued logic, a deeper support for modelling and reasoning about uncertainty is desirable.

Finally, ontological models have clear advantages regarding support for interoperability and heterogeneity. Moreover, since they support the representation of complex relationships and dependencies among context data, they are particularly well-suited to the recognition of high-level context abstractions. However, ontological models alone are generally unsuited to the recognition of simpler context data, like, e.g., basic physical activities; in order to recognize such context data, ontological models should be at least integrated with statistical machine learning methods (see, e.g., [66]). Furthermore, when considering the tradeoff between expressiveness and complexity the choice of ontological models may not always be satisfactory. In particular, in addition to the expressivity and complexity issues illustrated in Section 5, we argue that ontologies are not well suited to represent some dynamic context data such as users' adaptation preferences; these data can be more profitably modelled by lower-complexity, restricted logics (e.g., those proposed in [34] and [8]). Moreover, even if some preliminary proposals to extend OWL-DL to represent and reason about fuzziness and uncertainty exist (see, e.g., [72,21]), at the time of writing ontology languages and related reasoning tools do not properly support uncertainty in context data.

The above considerations seem to suggest that different models and reasoning tools need to be integrated with each other. Though a single expressive representation language fulfilling most of the identified requirements could probably be defined, there are strong indications that the resulting complexity of reasoning would make it useless in real-world scenarios. In the area of knowledge representation, an alternative approach to the use of a single very expressive formalism has been identified in *hybrid knowledge representation formalisms*; i.e., formalisms composed by different sublanguages to represent different kinds of knowledge, and loosely coupled reasoning procedures. One of the advantages of such formalisms is that the complexity of hybrid reasoning is generally no worse than the complexity of reasoning with the single sublanguages. In the next section we report two early hybrid approaches proposed for context representation and reasoning.

8.2 Existing hybrid approaches to context modelling

Hybrid fact-based/ontological model. Henricksen et al. [36] propose a hybrid approach to context modelling, combining ontologies with the fact-based approach provided by the CML language. The goal is to combine the particular advantages of CML models (especially the handling of ambiguous and imperfect context information) with interoperability support and various types of reasoning provided by ontological models. The hybrid approach is based on a mapping from CML modelling constructs to OWL-DL classes and relationships. It is worth noting that, because of some expressivity limitations of OWL-DL, a complete mapping between CML and OWL-DL cannot be obtained. With respect to interoperability issues, the advantages gained by an ontological representation of the context model are clearly recognizable. However, with respect to the derivation of new context data, experiences with the proposed hybrid model showed that ontological reasoning with OWL-DL and its SWRL extension did not bring any advantage with respect to reasoning with the CML fact-based model. For this reason, ontological reasoning is performed only for automatically checking the consistency of the context model, and for semantic mapping of different context models.

Loosely coupled markup-based/ontological model. The *CARE* [1] framework for context-awareness adopts a context modelling approach that is based on a loose interaction between a markup model – extended with policy rules expressed in a restricted logic programming language – and an ontological model. The interaction between these models is realized through the representation of context data by means of CC/PP profiles which contain a reference to OWL-DL classes and relations. In order to preserve efficiency, ontological reasoning is mainly performed in advance with respect to the service provision. Whenever relevant new context data is acquired, ontological reasoning is started, and derived information is used, if still valid, at the time of service provisioning together with efficient rule evaluation. Complex context data (e.g., the user's current activity) derived through ontological reasoning can be used in rule preconditions in order to derive new context data such as user preferences. As an example, consider the following rule:

$$\texttt{hasCurrActivity}^*(x,\texttt{BusinessMeeting}) \rightarrow \texttt{hasAvailState}(x,\texttt{Busy}).$$

The rule precondition involves complex context data – identified by a *star* symbol – that represents the current activity of an individual instance x (in this case, the current user). As in [36], ontological reasoning is also performed to check the consistency of the context model.

8.3 Towards a hierarchical hybrid model: gains and open issues

We now illustrate how existing hybrid approaches may be further extended to design a hierarchical hybrid context model that may satisfactorily address a larger number of the identified prerequisites.

A preliminary proposal for a hierarchical model has been presented in [7] focusing on the spatial/ontological component. The model presented here is

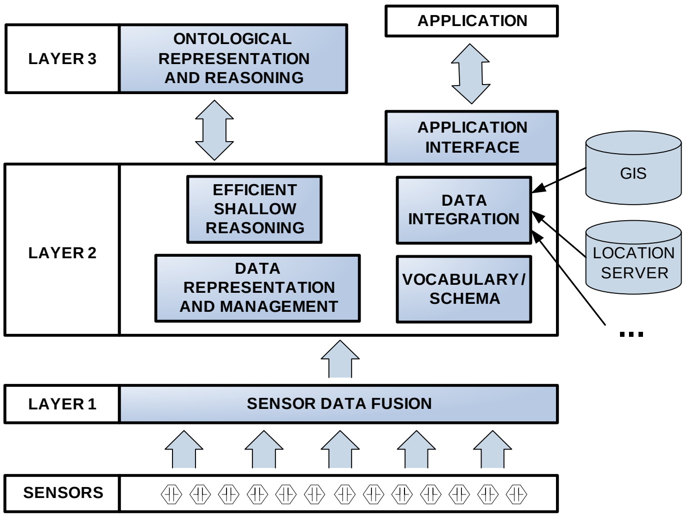

Fig. 5. Multilayer framework

intended to provide a more comprehensive solution, both in terms of integration of different forms of reasoning, and in terms of expressiveness. The proposed model includes a representation formalism to represent data directly acquired from sensors (or retrieved from a module executing some sensor data fusion technique). In order to support the scalability requirements of pervasive computing services, this representation formalism should make possible the execution of efficient reasoning techniques to derive high-level context data on the basis of raw ones (e.g., by executing rule-based reasoning in a restricted logic programming language, or statistical inferencing). Since such a representation formalism inevitably does not support a formal definition of the semantics of context descriptions, a more expressive, ontology-based context model is desirable on top of it. In addition to providing a formal semantics of the data, an ontological context model also supports the execution of reasoning tasks such as consistency checking and derivation of new context information. Clearly, there must be a mapping between terms used in context descriptions for efficient reasoning and ontological classes and relations. The corresponding framework is shown in Figure 5; it is composed of the following layers:

- Layer 1: Techniques for sensor data fusion (like, e.g., in [49]). This layer can also be organized as a peer-to-peer network of software entities devoted to acquire, process and propagate raw context data in the pervasive space in order to support cooperation and adaptation of services (see, e.g., [48]).
- Layer 2: This layer is devoted to shallow context data representation, integration with external sources, and efficient context reasoning. In particular, it includes the following modules:
  - · module for efficient markup-based, RDF-based, or DB-based representation and management of context data. This includes the definition of shared vocabularies (CC/PP vocabularies are an example for RDF-based

representation, and annotated DB schemas are an example for DB-based management);
  · modules for efficient shallow reasoning (logics and/or statistics-based, uncertainty reasoning may be supported);
  · data integration techniques for acquiring data from external sources (e.g., GIS, location servers, user modelling systems) and for conflict resolution (even due to conflicting rules).
- Layer 3: Realization/abstraction process to apply ontological representation and reasoning. This layer has the following main goals:
  · to specify the semantics of context terms (important for sharing and integration);
  · to check consistency;
  · to provide an automatic procedure to classify sets of context data as more abstract context abstractions.

The interface exposed to applications is provided at Layer 2. This is mainly due to efficiency concerns and to the fact that ontological reasoning is mainly based on relationships between concepts and not instances. Results of reasoning are reflected on instances at Layer 2 (this implies that markup/DB schemas include, at least as strings, all the terms in the ontology). Application developers have access to the ontology, that also provides the context semantics. The specific context terms required by an application will be found at Layer 2, and their values will be returned when required.

Even though the proposed hierarchical hybrid model determines clear advantages in terms of the requirements reported in the introduction, we point out that the integration of diverse reasoning techniques still poses open issues, e.g., how to integrate the open-world semantics of ontologies with the closed-world semantics of DB-based models and logic programming (see [53] for a thorough discussion of this aspect), and how to reconcile probabilistic reasoning with reasoning with languages not supporting uncertainty (e.g., OWL-DL).

9 Conclusions

In this paper we described the state of the art in context modelling and reasoning that supports gathering, evaluation and dissemination of context information in pervasive computing. Existing approaches to context information modelling differ in the expressive power of the context information models, in the support they can provide for reasoning about context information, and in the computational performance of reasoning. In the paper we presented a set of requirements that context modelling and reasoning techniques should meet. The discussion of the requirements was followed by a description of the three, currently most prominent, approaches to context modelling and reasoning. These approaches are rooted in database modelling techniques and in ontology based frameworks for knowledge representation.

The paper also presented state-of-the-art techniques to deal with two particularly relevant issues that should be addressed in any framework for context representation and reasoning: high-level context abstractions and uncertainty of context information.

We concluded our survey by introducing hybrid approaches as an attempt to combine different formalisms and techniques to better fulfill the identified requirements. Since we believe this is a promising direction, we discussed a possible architecture, as well as some research issues to be investigated.

References

[1] A. Agostini, C. Bettini, D. Riboni, Hybrid Reasoning in the CARE Middleware for Context-Awareness, International Journal of Web Engineering and Technology, (2009).

[2] A. Agostini, C. Bettini, D. Riboni, Experience Report: Ontological Reasoning for Context-aware Internet Services, in: Proceedings of the 4th IEEE Conference on Pervasive Computing and Communications Workshops, IEEE Computer Society, 2006.

[3] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (eds.), The Description Logic Handbook: Theory, Implementation, and Applications, Cambridge University Press, 2003.

[4] J. Barwise, J. Perry, The Situation Underground, Working Papers in Semantics, Vol. 1. Stanford University Press, 1980.

[5] J. Barwise, J. Perry, Situations and Attitudes, The Journal of Philosophy 78 (11) (1981) 668–691.

[6] C. Becker, F. Dürr, On location models for ubiquitous computing, Personal and Ubiquitous Computing 9 (1) (2005) 20–31.

[7] C. Becker, D. Nicklas, Where do spatial context-models end and where do ontologies start? A proposal of a combined approach, in: J. Indulska, D. D. Roure (eds.), Proceedings of the First International Workshop on Advanced Context Modelling, Reasoning and Management, Nottingham, England: University of Southhampton, 2004.

[8] C. Bettini, L. Pareschi, D. Riboni, Efficient Profile Aggregation and Policy Evaluation in a Middleware for Adaptive Mobile Applications, Pervasive and Mobile Computing 4 (5) (2008) 697–718.

[9] P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, H. Stuckenschmidt, Contextualizing Ontologies, Journal of Web Semantics 1 (4) (2004) 325–343.

[10] O. Brdiczka, J. L. Crowley, P. Reignier, Learning Situation Models for Providing Context-Aware Services, in: Proceedings of Universal Access in Human-Computer Interaction (UAHCI 2007), vol. 4555 of Lecture Notes in Computer Science, Springer, 2007.

[11] G. Castelli, A. Rossi, M. Mamei, F. Zambonelli, A Simple Model and Infrastructure for Context-Aware Browsing of the World, in: Proceedings of the 5th IEEE Conference on Pervasive Computing and Communications, IEEE Computer Society, 2007.

[12] P. Castro, P. Chiu, T. Kremenek, R. R. Muntz, A probabilistic room location service for wireless networked environments, in: UbiComp '01: Proceedings of the 3rd international conference on Ubiquitous Computing, Springer-Verlag, 2001.

[13] T. Chaari, E. Dejene, F. Laforest, V.-M. Scuturici, A Comprehensive Approach to Model and Use Context for Adapting Applications in Pervasive Environments, International Journal of Systems and Software 80 (12) (2007) 1973–1992.

[14] H. Chen, T. Finin, A. Joshi, Semantic Web in the Context Broker Architecture, in: Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications (PerCom 2004), IEEE Computer Society, 2004.

[15] H. Chen, F. Perich, T. W. Finin, A. Joshi, SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications, in: 1st Annual International Conference on Mobile and Ubiquitous Systems (MobiQuitous 2004), IEEE Computer Society, 2004.

[16] B. Clarkson, Life patterns: structure from wearable sensors, Ph.D. thesis, Massachusetts Institute of Technology (2003).

[17] J. Crowley, J. Coutaz, G. Rey, P. Reignier, Perceptual Components for Context Aware Computing, in: Proceedings of UbiComp 2002: Ubiquitous Computing, 4th International Conference, Springer, 2002.

[18] W. Dargie, The role of probabilistic schemes in multisensor context-awareness, in: Proceedings of the 5th IEEE Conference on Pervasive Computing and Communications Workshops, IEEE Computer Society, 2007.

[19] A. Dey, Understanding and Using Context, Personal and Ubiquitous Computing 5 (1) (2001) 4–7.

[20] A. Dey, J. Manko, G. Abowd, Distributed mediation of imperfectly sensed context in aware environments, Tech. rep., Georgia Insititute of Technology (2000).

[21] Z. Ding, Y. Peng, A Probabilistic Extension to Ontology Language OWL, in: Proceedings of the 37th Annual Hawaii International Conference on System Sciences, IEEE Computer Society, 2004.

[22] S. Dobson, J. Ye, Using fibrations for situation identification, in: Proceedings of Pervasive 2006 Workshops, Springer, 2006.

[23] R. Fagin, J. Y. Halpern, N. Megiddo, A logic for reasoning about probabilities, Information and Computation 87 (1-2) (1990) 78–128.

[24] A. Frank, Tiers of ontology and consistency constraints in geographical information systems, International Journal of Geographical Information Science 15 (7) (2001) 667–678.

[25] F. Gandon, N. M. Sadeh, A Semantic E-Wallet to Reconcile Privacy and Context Awareness, in: Proceedings of ISWC 2003, Second International Semantic Web Conference, Springer, 2003.

[26] H. W. Gellersen, A. Schmidt, M. Beigl, Multi-Sensor Context-Awareness in Mobile Devices and Smart Artifacts, Mobile Networks and Applications 7 (5) (2002) 341–351.

[27] P. D. Gray, D. Salber, Modelling and using sensed context information in the design of interactive applications, in: EHCI '01: Proceedings of the 8th IFIP International Conference on Engineering for Human-Computer Interaction, Springer-Verlag, 2001.

[28] M. Grossmann, M. Bauer, N. Honle, U. Kappeler, D. Nicklas, T. Schwarz, Efficiently Managing Context Information for Large-Scale Scenarios, in: Proceedings of Pervasive Computing and Communications, IEEE Computer Society, 2005.

[29] T. Gu, X. H. Wang, H. K. Pung, D. Q. Zhang, An ontology-based context model in intelligent environments, in: Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference, San Diego, California, USA, 2004.

[30] T. A. Halpin, Conceptual Schema and Relational Database Design, 2nd ed., Prentice Hall Australia, Sydney, 1995.

[31] T. A. Halpin, Information Modeling and Relational Databases: From Conceptual Analysis to Logical Design, Morgan Kaufman, San Francisco, 2001.

[32] K. Henricksen, A framework for context-aware pervasive computing applications, Ph.D. thesis, School of Information Technology and Electrical Engineering, The University of Queensland (September 2003).

[33] K. Henricksen, J. Indulska, Modelling and using imperfect context information, in: 1st Workshop on Context Modeling and Reasoning (CoMoRea), PerCom'04 Workshop Proceedings, IEEE Computer Society, 2004.

[34] K. Henricksen, J. Indulska, Developing context-aware pervasive computing applications: Models and approach, Pervasive and Mobile Computing 2 (1) (2006) 37–64.

[35] K. Henricksen, J. Indulska, A. Rakotonirainy, Modeling context information in pervasive computing systems, in: 1st International Conference on Pervasive Computing (Pervasive), vol. 2414 of Lecture Notes in Computer Science, Springer, 2002.

[36] K. Henricksen, S. Livingstone, J. Indulska, Towards a Hybrid Approach to Context Modelling, Reasoning and Interoperation, in: J. Indulska, D. D. Roure (eds.), Proceedings of the First International Workshop on Advanced Context Modelling, Reasoning and Management, Nottingham, England: University of Southhampton, 2004.

[37] J. Hightower, B. Brumitt, G. Borriello, The location stack: A layered model for location in ubiquitous computing, in: Proceedings of the 4th IEEE Workshop on Mobile Computing Systems & Applications (WMCSA 2002), IEEE Computer Society Press, 2002.

[38] I. Horrocks, L. Li, D. Turi, S. Bechhofer, The Instance Store: DL Reasoning with Large Numbers of Individuals, in: Proceedings of the 2004 International Workshop on Description Logics (DL2004), vol. 104 of CEUR Workshop Proceedings, CEUR-WS.org, 2004.

[39] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean, SWRL: A Semantic Web Rule Language Combining OWL and RuleML, W3C Member Submission, W3C (May 2004).
URL `http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/`

[40] I. Horrocks, P. F. Patel-Schneider, F. van Harmelen, From SHIQ and RDF to OWL: The making of a web ontology language, Journal of Web Semantics 1 (1) (2003) 7–26.

[41] J. Indulska, R. Robinson, A. Rakotonirainy, K. Henricksen, Experiences in Using CC/PP in Context-Aware Systems, in: M.-S. Chen, P. K. Chrysanthis, M. Sloman, A. B. Zaslavsky (eds.), Mobile Data Management, vol. 2574 of Lecture Notes in Computer Science, Springer, 2003.

[42] G. Klyne, F. Reynolds, C. Woodrow, H. Ohto, J. Hjelm, M. H. Butler, L. Tran, Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0, W3C Recommendation, Tech. rep., W3C (January 2004).

[43] H. Lei, D. M. Sow, I. John S. Davis, G. Banavar, M. R. Ebling, The design and applications of a context service, SIGMOBILE Mobile Computing and Communications Review 6 (4) (2002) 45–55.

[44] A. Leonhardi, U. Kubach, K. Rothermel, A. Fritz, Virtual information towers - a metaphor for intuitive, location-aware information access in a mobile environment, in: Proceedings of the Third International Symposium on Wearable Computers, IEEE Computer Society, 1999.

[45] L. Liao, D. J. Patterson, D. Fox, H. Kautz, Learning and inferring transportation routines, Artificial Intelligence 171 (5-6) (2007) 311–331.

[46] S. Loke, On representing situations for context-aware pervasive computing: six ways to tell if you are in a meeting, in: Proceedings of PerCom Workshops 2006, IEEE Computer Society, 2006.

[47] W. Y. Lum, F. C. M. Lau, A Context-Aware Decision Engine for Content Adaptation, IEEE Pervasive Computing 1 (3) (2002) 41–49.

[48] M. Mamei, F. Zambonelli, Programming Pervasive and Mobile Computing Applications with the TOTA Middleware, in: Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications, IEEE Computer Society, 2004.

[49] B. Mathieu, M. Song, M. Brunner, M. Stiemerling, M. Cassim, A. Galis, L. Cheng, K. Jean, R. Ocampo, Z. Lai, M. Kampmann, Autonomic Management of Context-Aware Ambient Overlay Networks, in: Proceedings of Communications and Networking in China, IEEE Computer Society, 2007.

[50] R. Mayrhofer, An architecture for context prediction, in: A. Ferscha, H. Hörtner, G. Kotsis (eds.), Advances in Pervasive Computing, vol. 176, Austrian Computer Society (OCG), 2004.

[51] L. McCowan, D. Gatica-Perez, S. Bengio, G. Lathoud, M. Barnard, D. Zhang, Automatic analysis of multimodal group actions in meetings, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (3) (2005) 305–317.

[52] I. Millard, D. De Roure, N. Shadbolt, The use of ontologies in contextually aware environments, in: J. Indulska, D. D. Roure (eds.), Proceedings of the First International Workshop on Advanced Context Modelling, Reasoning and Management, Nottingham, England: University of Southhampton, 2004.

[53] B. Motik, I. Horrocks, R. Rosati, U. Sattler, Can OWL and Logic Programming Live Together Happily Ever After?, in: Proceedings of the 5th International Semantic Web Conference (ISWC 2006), vol. 4273 of Lecture Notes in Computer Science, Springer, 2006.

[54] B. Motik, P. F. Patel-Schneider, B. Parsia, OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax, World Wide Web Consortium, Working Draft WD-owl2-syntax-20081202 (December 2008).

[55] M. Muehlenbrock, O. Brdiczka, D. Snowdon, J. Meunier, Learning to detect user activity and availability from a variety of sensor data, in: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications, IEEE Computer Society, 2004.

[56] D. Nicklas, M. Grossmann, J. Mnguez, M. Wieland, Adding High-level Reasoning to Efficient Low-level Context Management: a Hybrid Approach, in: Proceedings of the Sixth IEEE Annual Conference on Pervasive Computing and Communications Workshops, IEEE Computer Society, 2004.

[57] D. Nicklas, B. Mitschang, The Nexus Augmented World Model: An extensible approach for mobile, spatially aware applications, in: Proceedings of the 7th International Conference on Object-Oriented Information Systems, Springer, 2001.

[58] J. Pascoe, The stick-e note architecture: extending the interface beyond the user, in: Proceedings of the 2nd International Conference on Intelligent User Interfaces, ACM Press, 1997.

[59] D. Randell, A. Cohn, Modelling topological and metrical properties of physical processes, in: Proceedings 1st International Conference on the Principles of Knowledge Representation and Reasoning, Morgan Kaufmann, 1989.

[60] A. Ranganathan, J. Al-Muhtadi, R. H. Campbell, Reasoning about uncertain contexts in pervasive computing environments, IEEE Pervasive Computing 3 (2) (2004) 62–70.

[61] A. Ranganathan, J. Al-Muhtadi, S. Chetan, R. Campbell, M. D. Mickunas, Middlewhere: a middleware for location awareness in ubiquitous computing applications, in: Proceedings of the 5th ACM/IFIP/USENIX International Conference on Middleware, Springer, 2004.

[62] A. Ranganathan, R. E. Mcgrath, R. H. Campbell, M. D. Mickunas, Use of Ontologies in a Pervasive Computing Environment, The Knowledge Engineering Review 18 (3) (2004) 209–220.

[63] P. Reignier, O. Brdiczka, D. Vaufreydaz, J. Crowley, J. Maisonnasse, Context-aware environments: from specification to implementation, Expert Systems 24 (5) (2007) 305–320.

[64] R. Reiter, Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems, MIT Press, 2001.

[65] G. Rey, Contexte en Interaction Homme-Machine: le Contexteur, Ph.D. thesis, Université Joseph Fourier (2005).

[66] D. Riboni, C. Bettini, Context-aware Activity Recognition through a Combination of Ontological and Statistical Reasoning, in: Proceedings of the 6th International Conference on Ubiquitous Intelligence and Computing (UIC-09), Springer, 2009.

[67] M. Roman, C. Hess, R. Cerqueira, A. Ranganathan, R. H. Campbell, K. Nahrstedt, A middleware infrastructure for active spaces, IEEE Pervasive Computing 1 (4) (2002) 74–83.

[68] B. Schilit, N. Adams, R. Want, et al., Context-aware Computing Applications, Xerox Corp., Palo Alto Research Center, 1994.

[69] A. Schmidt, Ubiquitous computing - computing in context, Ph.D. thesis, Lancaster University (2002).

[70] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. V. Laerhoven, W. V. de Velde, Advanced interaction in context, in: HUC '99: Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing, Springer-Verlag, 1999.

[71] G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, 1976.

[72] U. Straccia, Towards a Fuzzy Description Logic for the Semantic Web (Preliminary Report), in: Proceedings of the Second European Semantic Web Conference (ESWC 2005), vol. 3532 of Lecture Notes in Computer Science, Springer, 2005.

[73] T. Strang, C. Linnhoff-Popien, A Context Modeling Survey, in: J. Indulska, D. D. Roure (eds.), Proceedings of the First International Workshop on Advanced Context Modelling, Reasoning and Management, in conjunction with UbiComp 2004, University of Southhampton, 2004.

[74] T. Strang, C. Linnhoff-Popien, K. Frank, CoOL: A Context Ontology Language to Enable Contextual Interoperability, in: Proceedings of the International Conference on Distributed Applications and Interoperable Systems (DAIS 2003), vol. 2893 of Lecture Notes in Computer Science, Springer, 2003.

[75] X. H. Wang, T. Gu, D. Q. Zhang, H. K. Pung, Ontology Based Context Modeling and Reasoning using OWL, in: Proceedings of Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, IEEE Computer Society, 2004.

[76] H. Wu, Sensor data fusion for context-aware computing using Dempster-Shafer theory, Ph.D. thesis, Carnegie Mellon University (2004).

[77] J. Ye, L. Coyle, S. Dobson, P. Nixon, Using Situation Lattices to Model and Reason about Context, in: Fourth International Workshop on Modeling and Reasoning in Context (MRC 2007), 2007.

[78] L. A. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets and Systems 100 (1999) 9–34.

[79] D. Zhang, T. Gu, X. Wang, Enabling Context-aware Smart Home with Semantic Technology, International Journal of Human-friendly Welfare Robotic Systems 6 (4) (2005) 12–20.