

# Self-organized Data Ecologies for Pervasive Situation-Aware Services: the Knowledge Networks Approach

Nicola Bicocchi, Matthias Baumgarten, Nermin Brgulja, Rico Kusber,  
Marco Mamei, Maurice Mulvenna, Franco Zambonelli

**Abstract**—Pervasive computing services exploit information about the physical world both to adapt their own behavior in a context-aware way and to deliver to users enhanced means of interaction with their surrounding environment. The technology to acquire digital information about the physical world is increasingly available, making services at risk of being overwhelmed by such growing amounts of data. This calls for novel approaches to represent and automatically organize, aggregate, and prune such data before delivering it to services. In particular, individual data items should form a sort of self-organized ecology in which, by linking and combining with each other into sorts of “knowledge networks”, they are able to provide compact and easy-to-be-managed higher-level knowledge about situations occurring in the environment. In this context, the contribution of this paper is twofold. Firstly, with the help of a simple case study, we motivate the need to evolve from models of “context-awareness” towards models of “situation-awareness” via proper self-organized “knowledge networks” tools, and introduce a general reference architecture for knowledge networks. Secondly, we describe the design and implementation of a knowledge network toolkit we have developed, and exemplify and evaluate algorithms for knowledge self-organization integrated within it. Open issues and future research directions are also discussed.

**Index Terms**—Pervasive Computing, Self-organization, Context Awareness, Knowledge Networks.

## I. INTRODUCTION

**P**ERVASIVE computing researchers envision a future world in which computing and sensing devices will be embedded

Manuscript received May 30, 2008. This work was supported by the project CASCADAS (IST-027807), FET Program of the European Commission.

Nicola Bicocchi is with the DISMI -Università di Modena e Reggio Emilia, Reggio Emilia, Italy (e-mail: nicola.bicocchi@unimore.it)

Matthias Baumgarten is with the School of Computing and Mathematics, University of Ulster, Belfast, UK (e-mail: M.Baumgarten@ulster.ac.uk)

Nermin Brgulja is with the ComTec, University of Kassel, Kassel, Germany (e-mail: nermin.brgulja@comtec.eecs.uni-kassel.de)

Rico Kusber is with the ComTec, University of Kassel, Kassel, Germany (e-mail: rico.kusber@comtec.eecs.uni-kassel.de)

Marco Mamei is with the DISMI -Università di Modena e Reggio Emilia, Reggio Emilia, Italy (e-mail: marco.mamei@unimore.it)

Maurice Mulvenna is with the School of Computing and Mathematics, University of Ulster, Belfast, UK (e-mail: md.mulvenna@ulster.ac.uk)

Franco Zambonelli is with the DISMI -Università di Modena e Reggio Emilia, Reggio Emilia, Italy (e-mail: franco.zambonelli@unimore.it).

everywhere and will be able to produce digital information about almost each and every fact occurring in the physical world [Cas07, UliG07]. Such information can then be exploited at the user-level to deliver services for better perceiving/interacting with the physical world [Hon08], as well as to enforce the capability of dynamically and autonomously adapting services to the context in which they are invoked and exploited.

Paving the way for the full realization of the pervasive computing vision requires both: (i) technologies to capture digital contextual information and (ii) services able to access and exploit it efficiently and meaningfully.

With regard to the former point, the road is already being paved. Indeed, in the past few years, we have witnessed an increasing deployment of sensors [Est02], RFID tags [Wan06, Phi04], location systems [HigB01, Bel08], automatic users profilers [Bic08] that, together with the possibility of accessing from the Web a large variety of data items and facts about the world [Cas07], will soon form the basis of a globally shared and distributed infrastructure for the use of general-purpose pervasive services.

With regard to the second point, there is still a need to investigate principles and algorithms to organize, aggregate, and to enrich this growing amount of distributed information to make it more meaningful and, consequently, better understandable [Bau06]. In particular, we believe that there must be an evolution:

- From a model of simple *context-awareness* [DeyA00] in which the focus is providing services with simple interfaces to access heterogeneous context providers, leaving to services themselves the burden of understanding the retrieved information.
- Towards a model of *situation-awareness* in which a middle layer is in charge of organizing sparse pieces of information in order to provide services with a pre-digested and more comprehensive higher-level knowledge related to a “situation” of interest.

Our vision considers a world of networked knowledge that is made available via the concept of Knowledge Networks (KNs for short). We envision realizing the idea of a “self-organized data ecosystem” by defining proper models and tools to represent, analyze, self-organize, and self-aggregate contextual information, so as to form structured and

meaningful collections of related knowledge items [UliG07]. Thus, KNs may support services by allowing them to reach, with reduced efforts, a comprehensive understanding of “situations” and, consequently, a higher-degree of adaptability and autonomicity.

In this context, the key contributions of this paper are:

1. to motivate the need to evolve from models of “context-awareness” towards models of “situation-awareness” via proper self-organized “knowledge networks” tools, and to introduce a general reference architecture for knowledge networks;
2. to describe the design and implementation of a knowledge network toolkit we have developed, to exemplify algorithms for self-organization integrated in the toolkit and to evaluate their effectiveness.

The remainder of this paper is organized as follows. Section 2 motivates the need for pervasive services to bridge the gap between context-awareness and situation-awareness and introduces a case study to clarify the concept expressed (and to act as a running example throughout the paper). Section 3 introduces the key concepts of KNs and sketches a reference architecture for them. Section 4 describes the current prototype implementation of the KNs toolkit that has been implemented in the context of the European Project “CASCADAS”. Section 5 presents some representative knowledge aggregation and knowledge management algorithms currently integrated in the KNs toolkit. Section 6 presents experiments and performance measures to evaluate the KNs architecture and its algorithms. Section 7 discusses related works. Section 8 outlines open issues and future research directions and Section 9 concludes.

## II. FROM CONTEXT-AWARENESS TO SITUATION-AWARENESS

### A. Motivations

According to most assessed user-centric definitions [Buc03, DeyA00] “*context is any information that can be used to characterize the situation of an entity*” (i.e., a service or a software component) and that can be considered relevant to improve the interaction between such an entity and its users (e.g., exploit context-awareness to maximize its functional benefits). Recently, more software-centric viewpoints on context have emerged, which focus on context-awareness as a means for services to improve quality and reliability via autonomicity and adaptability (i.e., exploit contextual information to self-monitor, self-configure, self-reconfigure, etc.) [Cro03, Dob06, Ser04].

While we fully commit to the above definition, we also perceive that technological advances are creating a notable gap between “*context is any information*” and “*that can be used to characterize the situation of an entity*”. That is, acquiring contextual information does not necessarily imply the capability of understanding situations, especially in the presence of an overwhelming amount of data and a lack of relations between them.

As already mentioned, the imminent mass diffusion of pervasive technologies such as sensor networks [Est02,

ChoK03], RFID tags [Wan06,Phi04], localization tools [HigB01, Bel08], will soon make pervasively available an incredible amount of real-time information about the physical world, its processes, and its objects. Further, the dramatic success of participatory Web tools, aka Web 2.0, is feeding the Web with information of any kind about any topic. In particular, tools such as Google Earth get continuously enriched by geo-located and localized contextual information coming from very diverse social communities and relate to a variety of facts and events situated in the real world [Cas07].

Overall, the above trends contribute to increase the amount of contextual information that can be exploited by pervasive services in order to achieve a higher degree of contextual awareness. However, the fruitful exploitation of the above described information calls for:

- Notable communication efforts to retrieve, from a variety of diverse devices (and possibly from remote sources) all needed information;
- Notable computation efforts to analyze available information, with the goal of making them more meaningful (i.e., associated to situations) and ultimately machine understandable.

### B. A Case Study

To ground the discussion, let us consider the scenario of a modern exhibition center, like a big museum or a stadium. In this kind of scenarios, it is realistic to assume the presence of a pervasive infrastructure of embedded devices such as sensors of various types, lots of WiFi access points, RFID tags and location systems such as GPS devices. In fact, exhibition centers may afford the costs of deploying such infrastructures if this allows them to provide better services that consequently may attract a higher number of visitors, which in turn may lead to higher revenues. Furthermore, the same type of infrastructure may be used to increase security and to provide pervasive safety and communication mechanisms.

As a specific example of a service that can be attractive to visitors and that can also attract revenues, we consider the presence of a number of advertising screens that can be used to display to visitors information about the exhibition itself as well as commercials. Today, such advertising screens display generic information in a simple cyclic way that is independent of the situation they operate in (i.e., independent of who is actually in the proximity of that screen). Instead, a “smart” service can decide what information to display on the basis of the available contextual information (e.g., capturing user profiles by accessing Bluetooth-enabled PDAs owned by users, or by reading RFID tags worn by them). This would increase the value of the displayed advertisement both for users and for advertising companies.

The problem is that in a large exhibition center with many thousands of people, and with a large number of devices that produce contextual information, a single software component on a screen would have to manage an incredible amount of information to get a clue of what to do. Such information may include: (i) thousands of possibly incomplete user profiles that have to be synchronized with statistical information available

elsewhere or with some information extracted from other sources, (ii) a multitude of sensorial data detailing what users are currently doing, (iii) historical data detailing what they have done in the past to be possibly used for understanding what they will do in the future. Also, the components on dispersed screens may have to coordinate their actions to, e.g., limit the amount of commercials of a given company to show.

In summary, the case study outlines the potential for the emergence of the following paradox: the large amount of information available around, instead of being able to provide useful information can make services unable to act properly. That is, being able to access contextual information does not imply becoming aware of what's happening around and, ultimately, be able to act accordingly.

### III. THE KNOWLEDGE NETWORKS APPROACH

To make contextual information meaningful and useful, some tools must be made available to pervasive services that can properly correlate and pre-digest contextual information so as to provide them with a higher-level understanding of situations around, without forcing them to access and manage large amounts of data internally.

#### A. Knowledge Networks as Self-organizing Data Ecologies

To avoid pervasive services to access and digest large amounts of data directly, a sort of “middle-layer” must be placed in between the data sources and the services. Such a middle-layer will be in charge of digesting data items, analyze them and build a compact, higher-level view of the context.

With reference to the case study, such a layer could facilitate the aggregation of user profiles, possibly merging them with sensorial information, in order to provide situation-specific knowledge to services and enabling them to immediately act on its basis. For instance, one can think of aggregating individual data items describing users with similar interests. This can define a new, higher-level, aggregated data item eventually, representing in a compact way the overall situation of users around a screen, e.g., “there are 70% of women who are interested in modern art” or “80% of visitors are approaching the cafeteria”. By correlating such information with other sources (i.e., ambient sensors), one can easily infer, for example, “80% of visitors are approaching the cafeteria AND it is very warm and humid”. In the case study, having the possibility of accessing information of this kind can be very useful for quickly deciding what advertisement to show on a screen.

Clearly, to be effective in pervasive scenarios, the envisioned middle-layer has to rely on a distributed and lightweight architecture and must strongly exploit self-organization within. In particular, data organization, aggregation, and generation of higher-level data items must occur in an adaptive way and without requiring human intervention. That is, the envisioned middle-layer must define an underlying distributed “ecosystem” that is populated with data items that interact with each other, and as if they were simple cyber-organisms, are able to aggregate autonomously with each other, self-establish networks of relations with each

other, and also of combining with each other to generate high-level cyber-organisms (i.e., high-level knowledge items).

Our idea of knowledge networks is fully in line with the above perspective (see Figure 1). KNs are a kind of lightweight “middle-layer” concept in which atomic units of knowledge are automatically processed, combined into high-level concepts, and eventually made available to services via a dedicated querying interface.

A possible criticism of the proposed KNs approach is that it does not eradicate the problem of analyzing large amounts of information, but simply passes it to a different component at the KNs level. Although this may be true to some extent, one should consider that: (i) the approach promotes a clear separation of concerns that – as always in software engineering – can notably reduce the complexity of developing and maintaining services; and (ii) in a distributed setting, KNs can take care of knowledge management duties that would have been otherwise replicated inside each service. The latter point in particular has the potential to optimize the process of managing knowledge in a distributed environment.

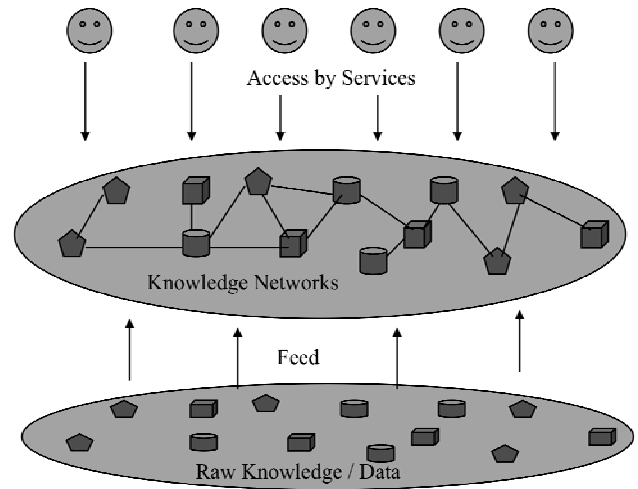


Figure 1: The Knowledge Networks approach.

#### B. A Reference Architecture for Knowledge Networks

It is important to define a reference architecture to implement our idea. Let's start with the assumption that there are a lot of various kinds of “sensors” (whether physical sensors, software sensors, or social Web 2.0 sensors) generating large amounts of (mostly) independent atomic units of contextual information (see Figure 1). We can call these “knowledge atoms”. The KNs approach considers exploiting self-organization approaches to aggregate/correlate/prune such knowledge atoms to facilitate their exploitation by services.

When considering that even relatively small network scenarios can generate enormous amounts of knowledge, it is necessary that KNs can provide different levels of abstraction as well as flexible means of correlating and managing knowledge. Furthermore, different kinds of services may have different needs in terms of type, scope and format of knowledge required.

Accordingly, one has to consider the possibility of a multiplicity of KNs to co-exist within a distributed knowledge

space where each network is limited by clearly defined knowledge boundaries in order to serve application-specific and/or service-specific goals. Although the context is the same for all situations (and thus the basic contextual information is the same) the way in which this has to be perceived and elaborated by services may depend on the specific type of service one wants to deploy. For instance, in the case study, a service to display commercials may be more interested in the gender distribution around in order to decide whether to advertise ties or perfumes, while a service to display information about specific events may be more interested in the cultural distribution of people around in order to decide whether to inform about a poetry lecture or about an on-going comedy show.

Obviously, it is illusionary to identify all possible dimensions in which knowledge may be organized. However, it is feasible to identify a given subset that is useful for various applications. This includes:

- A purely *semantic dimensions*, in which knowledge atoms that are related to a situation relate to each other according to the concepts that are available or inferred from e.g. a shared ontology. This can be the case for knowledge that facilitates and supports spontaneous interoperability in pervasive computing and service-orientated computing, or of knowledge related to inferring users' activities from a variety of heterogeneous sensorial information.
- A *spatial dimensions* in which knowledge atoms that are related to a local fact can network to knowledge atoms at different locations (or distribute/replicate themselves in different locations). This can be of use to express e.g. distributed situations, in which spatiality actually refers to physical spatiality, and which can be of great use for pervasive services. Also, we could conceive any class of spatially distributed P2P structures to distribute knowledge across a network and to facilitate access to knowledge (as in the case of e.g. knowledge brokers).
- A *temporal dimensions*, in which knowledge atoms express facts, which have occurred (or are about to occur) at different times. This can be the case for elaborating knowledge for predictive purposes: starting from a situation at a given moment in time, then analyzing and extracting new knowledge in the form of a KNs expressing the most likely future situation.

These considerations summarize into the conceptual reference architecture for KNs depicted in Figure 2. The figure also shows that KNs can also be organized around additional application-specific dimensions in which knowledge atoms may be organized in multiple KNs that serve different purposes, and possibly overlap with each other (as in the application example, where we have exemplified how a service may need to be aware of the gender situation and another of the cultural situation).

### C. Knowledge Networks Roadmap

The deployment and integration of KNs into pervasive applications will be necessarily an incremental process.

In the short-term, KNs can be integrated with existing

(legacy) systems by using wrapper and proxy components. On the one hand, KNs components can warp existing context providers allowing to integrate them into the KNs. On the other hand, it is rather easy to write proxy components to access KNs services from other applications. KNs components of our prototype implementation are already provided with flexible API that can be easily invoked by proxy components.

In the long-term KNs can penetrate systems to provide semantically-rich, high-level context information to pervasive application at all levels in a not-layered way. Indeed, pervasive and autonomic applications will need context information at all levels of the ISO-OSI stack (from low-level data useful from packet routing in the network, to high-level information describing the user profile for contextualized advertisement) to properly tune their behavior.

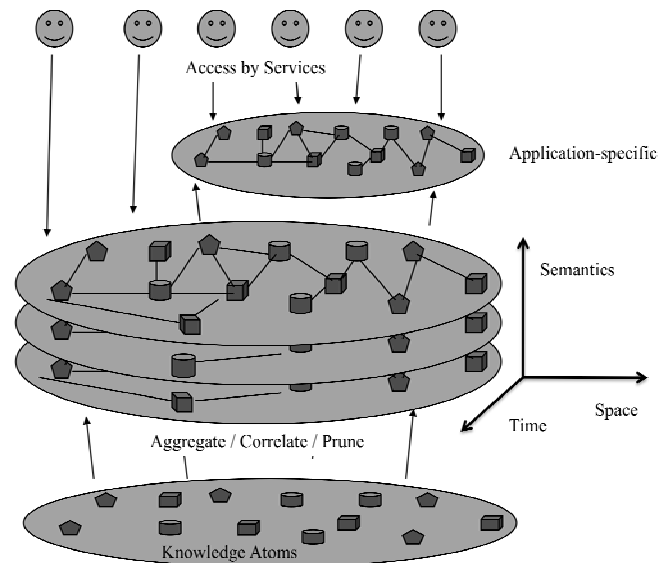


Figure 2: A Conceptual reference architecture for KNs.

To fulfill the KNs goals both in the short and in the long terms, self-organization is a fundamental property that is at the core of the KNs architecture. The self-organizing mechanisms that are embedded in KNs will gradually evolve to be able to deal with the ever increasing number of information sources spanned by the KNs. In order to manage and extract expressive information from large amount of context data in an effective way, several mechanisms are required. In particular, in order to comply with the decentralized nature of pervasive computing devices, the KNs framework make use of (bio-inspired) self-organizing and peer-to-peer approaches. Self-organization is endorsed in KNs both from a programming point of view (the architecture of the KNs facilitates the implementation of self-organizing mechanisms), and from an algorithmic point of view (some mechanisms at the basis of spatial self organization make use of self-organizing peer-to-peer protocols to spatially aggregated distributed data).

## IV. THE KNOWLEDGE NETWORKS TOOLKIT

In this section, we present the key features of the prototype KNs toolkit that we have implemented within the

CASCADAS project. To this end, we introduce the basic development toolkit upon which all the implementation relies, the key classes of components actually composing the KNs toolkit, and its overall architecture and functioning. The specific algorithms for knowledge management currently integrated in the toolkit are described later on in Section 5.

#### A. Autonomic Communication Elements

The CASCADAS project ([www.cascadas-project.org](http://www.cascadas-project.org)) aims at defining a general-purpose component-based paradigm for autonomic and situation-aware services for next generation network infrastructures and for pervasive computing scenarios. One of the goals of CASCADAS is to show that even middle-layer services, as KNs are, can be implemented by making use of the same paradigm.

The key concept at the root of CASCADAS is the *ACE* (*Autonomic Communication Element*), intended as a unifying software engineering abstraction for the development of component-based distributed services. The ACE model has been implemented in an associated ACE toolkit released by the CASCADAS consortium and is available in open source<sup>1</sup>. The ACE model takes inspiration and leverages from existing autonomic component-models and adaptive agent-based models with features conceived to facilitate the design and development of complex, self-adaptive and self-organizing network services running on a wide range of heterogeneous devices [Bel07,Cos07]. In particular:

1. ACEs are able to run both on high-end computers as well as on tiny devices like sensors, due to their internal light and modular structure. Moreover, ACEs are planned to be able to relocate themselves dynamically to different devices at run time by making use of mobile code techniques.
2. The internal functioning of ACEs relies on specific behaviors (i.e., roughly speaking, goal-oriented functions) that can be associated to individual ACEs or ACEs classes, and on an internal control loop that can enable self-monitoring and self-adaptation. More in detail, the service is co-modeled by a plan providing an explicit and machine processable representation of the actions the ACE will undertake (more concretely, this is a XML file encoding a finite-state automaton with the actions to be performed), and by a set of functionalities that can be dynamically invoked while the ACE executes its plan. The benefits of this separation are numerous: (i) plan generation or modification is possible without intervening at the code level, but only writing the plan representation. (ii) The plan is supervisable since it describes the operations to be performed from a high-level point of view (as a finite state automaton). (iii) Code (functionalities) is structured by means of plug-and-play individual activities.
3. The ACEs components are provided with two communication mechanisms: (i) a distributed publish-subscribe mechanism to advertise and look for services offered by other ACEs using semantic descriptions [EugF03]. This is called the GN/GA (Goal Needed/Goal Achievable) protocol in which ACEs publish the goals

they can achieve and look for goals they need for their task. The suitability of this kind of service discovery protocols is widely recognized in several pervasive and autonomic computing approaches [Bel07, Cos07, EugF03]. (ii) A direct message-passing mechanism allowing to flexibly contracting service usage. This mechanism supports bilateral and multilateral communication along previously defined connection partners and allows to implement properties, such as encryption or fixed-number-of-participants constraint [Hof07].

The combined use of all the above mechanisms allows ACEs to dynamically and adaptively connect with each other to provide advanced autonomic services. This characteristic is intended to fulfill the self-organization scientific aspect described in the introduction. Using the discovery mechanism ACEs can find interaction partners in open and dynamic environments where both the other ACEs around are unknown and they come and go at any time. Using the contracting mechanism ACEs can autonomously organize with other ACEs into contract chains to create advanced and complex services. These design principles work together to support the development of autonomic communication services.

ACEs do not explicitly support context-awareness. Simply, and in line with the overall CASCADAS vision (for which ACEs can be used to provide both user-level services as well as middle-layer services), it is assumed that access to contextual information can be provided by systems of specific middle-layer ACEs, to be contacted by application-level ACEs on need. In other words, this implies that KNs can be implemented by ACEs and dynamically accessed by other ACEs that need access to properly organized contextual information. The adoption of ACEs has enabled us to develop a complex and faceted KNs toolkit by focusing on architectural and algorithmic issues only. In fact, we have exploited the features of ACEs and utilized them in order to:

1. Deal with very diverse data sources and sensing devices: KNs components based on ACEs, each with a specified behavior, could be deployed on a number of heterogeneous platforms;
2. Deal with the unreliability and dynamicity of data sources and sensing devices by acting upon the ACE execution model. KN components can flexibly self-configure their behavior (e.g., modify the sampling rate in a sensor) by acting on the self model at run time;
3. Have the various components of the toolkit properly discover and interact with each other: the ACE GN-GA discovery mechanisms support KNs components to discover each other in a flexible decentralized way.

However, it is important to emphasize that the choice of implementing KNs in terms of ACEs and of making KNs available as an ACE-based service is not the only possible one. Indeed, the key concepts upon which the KNs approach relies are mostly technology-independent and could be developed on the basis of other component- or agent- based architectures expressing similar autonomic features. For example, other component based architectures such as RUNES [Cos07] or agent-based frameworks such as JADE [Bel07] could well serve the KNs purposes.

<sup>1</sup> <http://sourceforge.net/projects/acetoolkit>

### B. The KNs Basic Components

The implementation of the KNs toolkit relies on two basic classes of components, *knowledge atoms* and *knowledge containers* [Bau06], realized in terms of two specific hierarchies of ACE classes.

A knowledge atom represents the atomic unit of knowledge, and is typically connected to a data source. A knowledge atom provides a uniform abstraction to access contextual information independently of its type, size or context. This is required to provide generic access to knowledge from within the knowledge network as well as from services and components that are outside, independently of the specific characteristics of the data source (whether, e.g. a sensor, a tag, or a Web atom). In addition, a knowledge atom incorporates relevant descriptions of the knowledge/data associated with it, such as context, system and usage based information, as well as any information relevant for the creation, and maintenance of the knowledge atom itself. This makes each knowledge atom fully self-descriptive and as such provides information relevant for different organizational purposes as provided by the network. For instance, in the case study, each of the user profiles would be represented as individual knowledge atoms.

Most importantly, knowledge atoms can possibly link to each other to create clusters and networks of related information.

A knowledge container, on the other hand, is a structure capable of (virtually) encapsulating any number of knowledge atoms as well as other knowledge containers, thus providing a single point of access to multiple knowledge sources. The underlying concept of a knowledge container is similar to that of knowledge atoms (i.e. it encapsulates and makes available contextual information). However, the key point is that knowledge containers can “organize” knowledge, by making it possible to enforce and reify structural and behavioral relations between knowledge atoms and between other knowledge containers, in order to access such structured knowledge as if it were atomic information. Also, other than organizing knowledge, they can encapsulate algorithms and methods to manipulate knowledge, e.g., for analyzing, aggregating, pruning or transforming it. As a simple example, in the case study, a knowledge container could provide the “average profile” of visitors in one of the exhibition rooms, by encapsulating the, possibly large number of, atomic profiles that are required to compose a comprehensive profile. As another example, one can think of a knowledge container that, by analyzing the past and present information related to users, is able to predict and provide to services an estimation on the likely future position of a user.

More generically, by properly relating knowledge atoms and knowledge containers according to specific needs, it is possible to enable services to access contextual information according to different application-specific views and/or at different levels of granularity.

### C. The KNs Toolkit Architecture

The overall architecture of the implemented ACE-based knowledge networks toolkit is shown in Figure 3.

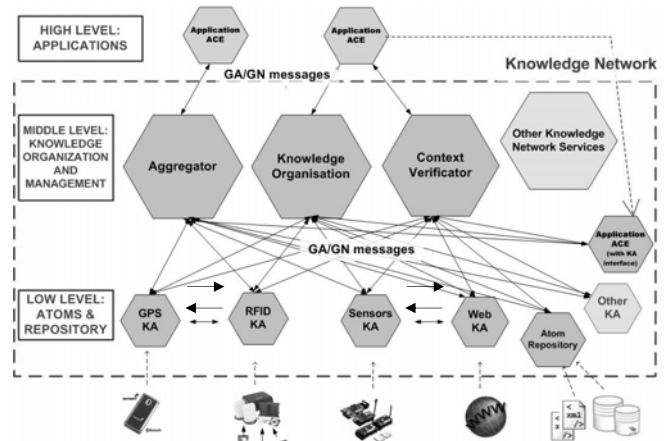


Figure 3: The architecture of the Knowledge Networks toolkit.

At the lowest level, the toolkit considers the presence of a number of ACEs, implementing the concept of knowledge atoms (KA for short in the figure) for specific data sources. The knowledge atom classes that we have implemented so far include: atoms for connecting to GPS devices, to CrossBow Micaz sensors, for accessing system properties in computational devices, for accessing Web information, and generic knowledge atoms for hosting static (pre-loaded) and historical information. Also, we emphasize that any application level service realized via an ACE can, by simply implementing a generic knowledge atom interface, publishes information/knowledge into the scope of the knowledge network and thus becomes a part of the population of knowledge atoms as depicted in the lower level of Figure 3.

For some kinds of data sources, either too resource constrained or too dynamic and volatile (e.g., RFID tags), it is unreasonable to allocate a dedicated knowledge atom to each of them. In these cases, a special kind of knowledge atom acting as a “Atom Repository” can be instantiated to provide, via a single component, access to a multitude of knowledge atoms. For instance, with regard to the case study, if the user profiles are stored in RFID tags and captured by one RFID reader (rather than being associated to a knowledge atom on the users’ PDA), it is possible to think of accessing individual profiles via a single atom repository associated to the RFID reader rather than allocating a knowledge atom for each of the captured profiles/tags.

At the *middle level*, we can find a number of knowledge container components that are used to organize, analyze, and manipulate the data provided by knowledge atoms, so as to actually reify the concept of knowledge networks. The KNs toolkit do not prescribe what knowledge containers should be instantiated at this level, nor does it limit the number and type of components that can be there. Depending on the specific needs of specific applications, new knowledge containers can be defined and allocated, also at run-time, to provide specific knowledge management functionalities and/or specific aggregation functions and/or specific knowledge views, also possibly building over existing knowledge containers. For instance, with reference to the case study, a knowledge container may be instantiated to collect all RFID knowledge

atoms related to user profiles and then produce an average profile.

All knowledge container components, the same as knowledge atoms, commit to provide a specific interface, centered around a simple “*getValue*” operation, with respect to the access to the contained information by application-level ACEs and by other containers. This does not prevent them from making available richer means of accessing and querying knowledge (e.g., some of the knowledge containers we have implemented provide “*à la Linda*” [Ahu86] access to the information).

As of now, we have already implemented a number of classes for knowledge containers, to test with a variety of knowledge organization algorithms/models. These include, other than simple containers applying simple aggregating functions (e.g. average, maximum, minimum) to an ensemble of knowledge atoms (as it can be the case for the user profiles in the case study), algorithms to facilitate advanced models of semantic knowledge organization, of spatial knowledge aggregation, as well as advanced models for knowledge consistency verification. Some of these algorithms are described in more detail in Section 5.

It is worth noticing that, although for the sake of simplifying the presentation, the organization of the architecture is presented as a layered one (knowledge atoms at the lower level, knowledge containers and supplementary components at the higher level), from the conceptual viewpoint, such layering does not exist. Instead, the various ACEs that compose the toolkit, apart from being of different classes, are peers with each other, and only the dynamic patterns of interactions that are established among them can eventually lead to some sort of structured (e.g. layered) organization. In particular, we can identify two main interaction patterns that will structure the KNs:

1. Links between knowledge atoms and knowledge containers. Knowledge containers will link to a number of atoms creating hierarchical relationship among concepts. Self-organizing mechanisms to create these links will be described in Section 5.1
2. Links between knowledge atoms. These links allow to organize and represent relationships among atoms and possibly instantiate knowledge containers. We used them in Section 5.2 to spatially self-organize distributed information atoms.

Concerning the allocation of the various components over a network, it is to be said that the components of a knowledge networks can execute and interact in a distributed setting, independently of their actual allocation. Thus, it is possible to have instantiated distributed KNs in which, for example, knowledge atoms are allocated directly on the data source devices (e.g., PDAs or sensors) and different knowledge containers are allocated on different nodes. For instance, in the case study, one knowledge container performing user profiles aggregation can execute directly on the advertisement screens, while a knowledge container devoted to monitor users’ for security reason can be allocated on the central server of the security office.

## V. KNOWLEDGE NETWORKS ALGORITHMS

In this section, we exemplify some of the specific algorithms for knowledge management and analysis that we have integrated within the KNs toolkit. In any case, we recall that the flexible architecture of the KNs toolkit enables the integration, even at run-time, of any new component to deal with specific knowledge management issues, and/or to build specific knowledge views.

### A. Semantic Self-organization

A fundamental dimension of interest with respect to self-organization of knowledge is represented by the semantic information to describe a knowledge atom. The necessity of semantic self-organization capabilities for KNs becomes particularly clear when considering that it is the basic tool in order to foster the universal use and exchange of data, information and knowledge at various levels of granularity.

As pointed out previously, the input layer for KNs is represented via the concept of knowledge atoms, which provides access to the knowledge source itself and is enriched with information describing the semantics of the knowledge represented. For most knowledge sources static concepts can be specified upon deployment. In some cases however, provided descriptions may be validated, enriched, or even gathered dynamically to accommodate for non-static information.

Typically, such semantics can be specified via simple keywords or alternatively could be described in RDF, which has already been established as a standard data model for the description of resources.

```

<Semantics>
<Keyword>Shopping Profile</Keyword>
<Keyword>Person.Female</Keyword>
<Keyword>Person.Middle Aged</Keyword>
<Keyword>Person.Professional</Keyword>
<Keyword>Interest.Clothes.Hat</Keyword>
<Keyword>Interest.Clothes.Handbag</Keyword>
<Keyword>Interest.Perfume</Keyword>
<Keyword>Location.GPS[Standardised GPS Information]</Keyword>
<Keyword>Location.NEARBY[Based on GPS]</Keyword>
</Semantics>

```

Figure 4: Example of semantic description.

In relation to the case study introduced earlier, an example description of semantic keywords is given in Figure 4, which could provide a generic and anonymous profile of individual shoppers. In turn, the KNs can utilize such profiles to generate different views about the population near advertisement screens. For example, the keywords depicted in Figure 4, reflect a simple shopping profile of a middle-aged woman that is interested in hats, handbags and perfumes. Notable here is that descriptive information may include various keywords as well as the relations with each other. This does provide relevant information to construct individual ontologies in a bottom-up fashion. Furthermore, a similar model may also be used to reflect more active semantic information, e.g. GPS data that reflect the current position of the “knowledge” item. Such data may either be utilized directly or translated into more meaningful semantic concepts, such as addresses or proximity statements (e.g. within 50m of the advertisement

screen, or nearby Times Square, etc.). Thus, semantic information may be categorized to be, firstly, of either static or dynamic nature and, secondly, to be of either lower or higher contextual value. Once registered within the network, the objective of the semantic self-organization algorithm is to analyze the descriptive part of each knowledge atom by synchronizing this into a global, distributed hierarchical or network-like structure where knowledge sources are clustered based on the semantics they support. Considering that such organization is, from a structural point of view, facilitated via knowledge containers, which reference to other knowledge containers as well as knowledge atoms. The top-level knowledge container does represent a distinct KNs. Atoms will be registered to the respective containers they support thus forming individual clusters based on the semantic meaning provided by their sources. In addition, links between containers are formed to provide the hierarchical or network like structure that links individual concepts, providing different granular levels on which knowledge can be further organized, processed or queried.

In the current implementation a simple yet effective self-organizing algorithm deals with the insertion and clustering of knowledge atoms into the KNs. As illustrated in Fig. 4, knowledge atoms are provided with a set of keywords describing the information they contain. Similarly, knowledge containers are described with a similar set of keywords indicating the kind of information they can contain. Once a context provider creates a new knowledge atom describing some kind of context information, a reference to the knowledge atom is sent to the main KNs infrastructure. The KNs distributed infrastructure performs a pattern matching process to link the new knowledge atom to the existing knowledge container on the basis of keyword matching. On this basis, the overall structure of the KNs -- that is based on how knowledge atoms and knowledge containers link to each other -- self-organizes as a result of the pattern matching process. Another important self-organizing aspect in our framework is that the infrastructure can autonomously create knowledge containers to organize a set of knowledge atoms. Once the number of knowledge atoms with any keyword exceeds a given threshold, a knowledge container with that keyword is automatically created by the infrastructure and the knowledge atoms with that keyword are linked to that container. This is another aspect of self-organization: the KNs dynamically changes its structure on the basis of the information knowledge atom being produced.

The organizational structure derived from the example depicted in Figure 4 is shown in Figure 5. There, three individual ontologies (not counting “Shopping Profile”) have been constructed that each reflect an individual concept and is represented by individual knowledge containers (note that the “location” ontology can be relevant for spatial organization as discussed in the next Subsection). If strictly hierarchical, the resulting construct is comparable to an ontology, which in the case of KNs is always shared among the entire network. That is that no part of any given construct is created or used only for a single object (atom or container) but shared among all

available objects that are within the same organizational space.

Although bottom-up construction maximizes flexibility and avoids the need to define overarching system-wide ontologies, we are aware it limits the interoperability among pervasive services connected to different KNs. This is a general problem at the core of the semantic Web community: while lightweight bottom-up ontologies are flexible and easy to use, system-wide ontologies enable more comprehensive interoperability and it is difficult to find the proper tradeoff among these two extremes. Still a number of researches to create meaningful mappings between ontologies have been proposed [Kal03] and we plan to integrate such mechanisms in the KNs framework.

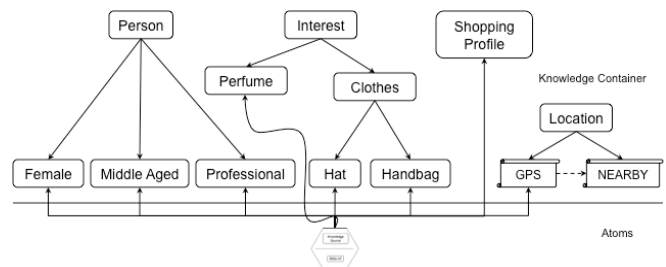


Figure 5: Knowledge network ontology.

### B. Spatial Self Organization

From the spatial self-organization point of view, we are studying algorithms to enable the correlation and aggregation of distributed spatial data. More in detail, we have explored the possibility of extracting high-level and compact knowledge about the structure of an environment as sensed by a sensor network [Bic07]. In the following example knowledge atoms self-organize themselves in order to obtain a simplified view of the environment. The basic idea is to have knowledge atoms executing a distributed gossip-based algorithm. They periodically exchange data with neighboring atoms. A logical link between two neighbors is re-enforced if the environmental characteristics are similar and weakened otherwise. When the status of the links reach a sufficient degree of stability, the network of knowledge atoms is able to self-organize itself into a set of distinct partitions (i.e., Macro Sensors) each corresponding to a region of the environment characterized by a specific sensing pattern, e.g. a room with a specific temperature or light levels (see Figure 6).

Then, a set of knowledge containers is instantiated in order to represent the network as if it was composed by several Macro Sensors. By this way, the possibly large amount of sensorial data generated by the network can be not necessarily perceived as a multiplicity of unrelated information. Instead, the algorithm makes it possible to perceive the sensor network as if it were made up of a more limited number of “macro sensors”, each associated to a well-characterized region of the physical environment. To some extent, the algorithm provides for the automatic construction in the KNs of specific spatial views by aggregating data to represent the overall “situation” of a region of the environment. This can facilitate usage by services. Moreover, it is worth emphasizing that every container (as a macro sensor) could provide application-specific aggregation functions on its region. In this way the

KN is not only a way to perceive the environment in a simplified fashion, but can also act as a computing layer able to produce derived knowledge.

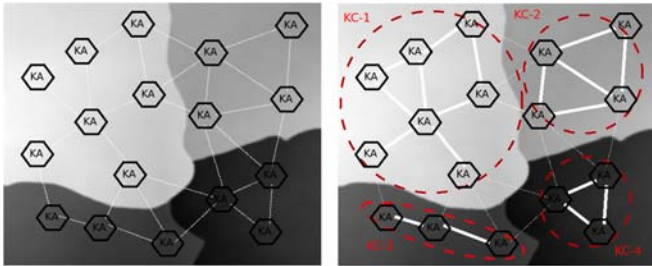


Figure 6: (left) 4 recognizable regions of an environment as identified by a specific property, and a network of Knowledge Atoms immersed on it. (b) Knowledge Atoms spatially self-organize into four virtual overlays, reflecting the environment. Each region is then represented by a single Knowledge Container acting as a sort of macro sensor for that region.

In the case study, one could think of realizing a similar aggregation algorithm that firstly partitions the cloud of all user profiles into clusters of users that are characterized by similar interests, and then averages data over each cluster. Since the distributed information of the various user profiles are clustered together, this can be a simple example of spatial self-organization. This can enable services to get a synthetic clue of what the overall preferences of users are and to reach a quick decision on what advertisement to show on a screen. Also, due to the fact that KNs can be realized both in a fully-distributed fashion (in sensor networks) as well as a centralized solution (in the Web-based repository), services can dynamically decide how to access information.

From a complementary perspective, we are also studying how location information about users (e.g. as provided by GPS devices) can be organized so as to infer spatial relationships among users and services/resources. For example, we can easily infer how close a user is to a given resource (e.g. a restaurant). Such kind of information may be useful in several applications. For instance, if a person enters a supermarket one may want to observe its movements in order to provide specific services, such as person-centric advertisement, seasonal offers, etc. To realize such services, once a person has entered a shopping mall, the person's profile could be registered with a dedicated knowledge container that employs a different, more specific means of organization. For example, such a container could analyze the specific location of the person within the shopping mall identifying if the person is for instance near an advertisement screen or if the person is entering a specific shop. Other forms of organization could rely on evaluating the surroundings of the person's location, and on creating a virtual orb around the person's position and querying other sources if they are within this orb or not. Theoretically, due to the fact that KNs can be strongly distributed among several computational resources, there are unlimited possibilities about how such knowledge sources can be organized and related with each other.

### C. Context Verification

For systems that organize large amounts of data, reliability

plays an important role. Including incorrect data into a self-organizing structure like a KNs will lead to error propagation and wrong inference results. Though not all errors can be prevented within KNs, we have the ambition to minimize the utilization of faulty information. For that reason we developed a *context verification* mechanism and integrated it into the KN toolkit.

With context verification we mean a process that examines contextual information and decides, based on historical reference data, whether this information is valid or suspicious. To decide upon the validity of contextual information, *context groups* are built by pooling together sets of semantically interrelated sensors. Each set of sensor values at one point in time and within one context group is then called a *context pattern*. For example, the readings of all the temperature sensors in a particular building at a given time may represent a context pattern.

A *context verifier*, which is an active component of KNs, is associated to one or more context patterns. Naturally, a multitude of context verifiers can exist within one KNs or system of KNs where each of them is responsible for surveying a dedicated set of context groups.

Our context verification method works probabilistically by collecting reference context patterns that occurred in the past. Depending on the occurrence probability of the pattern under examination within the history data, the verifier decides whether it is likely or not that this pattern is valid or suspicious. A parameter called *validity threshold*, which is a lower bound for a pattern's occurrence probability within a reference data set, is used for that classification. Therewith, the pattern does not necessarily need to exactly match the reference patterns but will be examined based on a similarity measure. Thus, if the present reading is far from the context pattern (i.e., its probability is below the *validity threshold*), the system can mark it as suspicious. In particular, for numeric values (sensor readings) we developed a distance measure called *data granularity level*. That granularity represents the distance between two neighboring values within the value range of a sensor (e.g. a data granularity level of 1 means that values are rounded to the next whole number instead of using accurate real numbers). We call the probability with which the verifier works correctly, i.e. classifies erroneous patterns as suspicious and correct patterns as valid, *verification accuracy*. This accuracy depends on a variety of parameters like the ratio between correct to erroneous patterns, number of reference patterns, data granularity level, or validity threshold. We have examined the influence of different parameters on the verification accuracy within a set of experiments whose results are presented in the next section.

## VI. EXPERIMENTAL RESULTS

We conducted a set of experiments to evaluate the KNs framework. Here we present experiments related to some of the main aspects of the KN approach, i.e., (i) results illustrating performances of the implemented KNs toolkit; (ii) results analyzing one of the self-organizing algorithms we

have in KNs; (iii) results from the context verification mechanisms.

### A. Infrastructure

To assess our implementation, we performed some experiments to show the performances of the presented toolkit. In particular, we tested the response time of the KNs to queries.

We put in place several atom repositories either local and connected through the Internet. The remote atom repositories were installed in several European research facilities (member of the CASCADAS project). We ran queries from different networks and monitored minimum and maximum response times of the networked KNs.

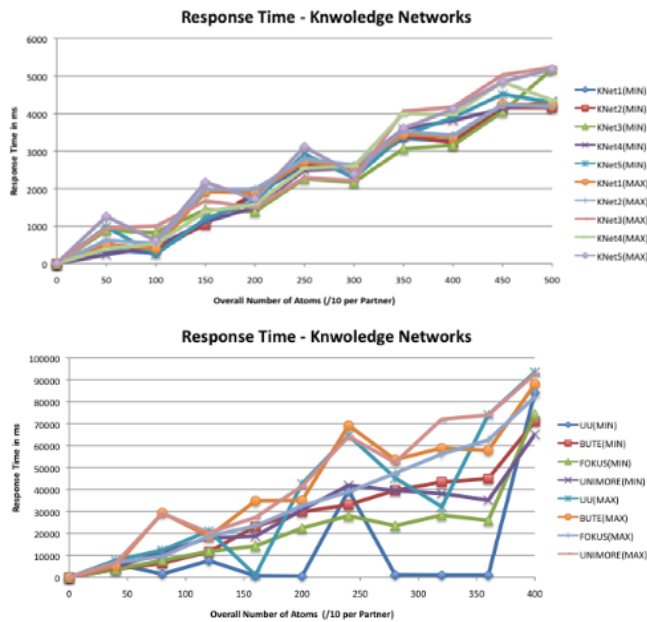


Figure 7: (top) Min. and Max. Response Times among several local KNs. (bottom) Min. and Max. Response Times among several KNs scattered through Europe: UU – University of Ulster (UK), BUTE - Budapest University of Technology and Economics (Hungary), FOKUS - Fraunhofer Institute (Germany), UNIMORE – University of Modena and Reggio Emilia (Italy).

Figure 7 (top) shows response times obtained from local Knowledge Networks. More in detail, we run several queries interrogating *all* the knowledge atoms in the local KNs and then we averaged the response times together. As expected, the results they are fairly regular and predictable. Still, some remarks can be made:

1. The linear increase in response time highlights that the system's performance does not deteriorate dramatically at a certain scale. Thus, the system does not exhibit trashing.
2. These tests illustrate a limit scenario, in which all the knowledge atoms are queried. In normal working conditions, instead, only a fraction of atoms will be queried (i.e., those referenced by a given knowledge container or knowledge view), thus performance will notably improve.
3. The implementation of the system is not optimized.

The construction of dedicated data structure like indexes, would dramatically speed up the query process.

Figure 7 (bottom) reports a similar experiment in a distributed setting. The experiment consisted in querying multiple distributed KNs connected over the Internet and measuring minimum and maximum response times. Also in this case, the reported results refer to the worst case possible, where all the knowledge atoms are queried.

In contrast with the previous experiment, these graphs show that the response time scales almost linearly with the size of the KNs and can have large fluctuations because of components workload and network delays.

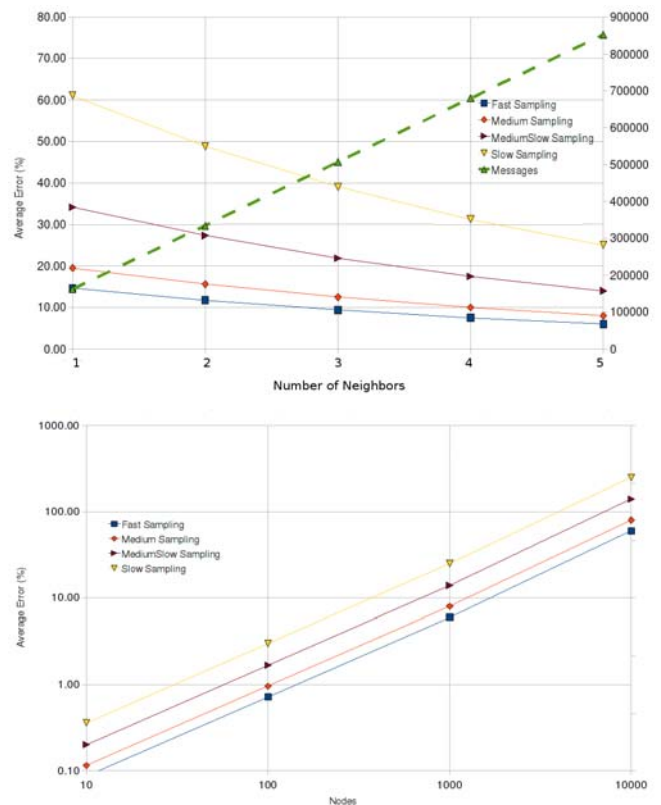


Figure 8: (top) Average error for different network densities. (bottom) Average error for different network sizes.

### B. Spatial Self Organization

To quantitatively evaluate one of our self-organizing mechanism, we focused on the algorithm at the basis of *spatial* self-organization. In particular, we conducted some experiments both in a simulation environment using the KNs toolkit to verify the convergence and accuracy level of our approach in large-scale scenarios and in a real sensor network test bed. This example clearly show how KNs could be exploited not only to manage relations between several knowledge sources but also to handle to inherent dynamism in physical systems. In particular, we used the algorithm to compute the average value among the data collected by a sensor network in a region. Knowledge atoms installed on sensors compute local averages and propagates them across the network by gossiping. This process leads iteratively to the

computation of the average in the region that will be made available by a knowledge container [Bic07].

In Figure 8 we summarized the performances of the proposed spatial self-organizing algorithm. In particular we measured the aggregated errors produced by the approach while partitioning a networks with different sizes and densities. Figure 8 (top) shows that the average error tends to decrease increasing the network density because of the number of interactions between knowledge atoms increases as well. More interactions produce an higher convergence speed. Obviously, also the number of messages being exchanged, and then the communication costs, increases. Figure 8 (bottom) shows the scalability of the algorithm varying the network size from 10 to  $10^4$  nodes. As expected the average error increases with the number of participating nodes. This happens because the number of iterations needed by the algorithm to converge increases with the number of nodes. However, an almost linear (in the logarithmic domain) trend guarantees good scalability and acceptable errors even if applied on networks with thousands of nodes.

Finally, it is possible to see that a decrease in the sensor sampling rate (that is equivalent to an increase in the environment dynamism) slowly degrades performances without altering the overall behavior.

These experiments show the effectiveness of the self-organizing algorithm we have implemented to spatially correlate several distributed knowledge atoms.

### C. Context Verification

Concerning context-verification, we have examined the influence of different parameters on the verification accuracy within a set of experiments in which we investigated the accuracy of our technique depending on the validity threshold, the data granularity level, and the size of context patterns.

Figure 9 shows how the data granularity level and the validity threshold influence the verification accuracy. It can be seen that, depending on the granularity, there is a small range of threshold values in which acceptable results can be achieved. The higher the threshold, the more often the pattern that is currently under verification must be found, within the reference data. A higher granularity leads to an increasing range within which the validity threshold has to be in order to achieve acceptable verification accuracy.

Figure 10 depicts how the verification accuracy is influenced by the size of the context pattern. The validity threshold has been fixed to 0,025% and the number of sensors involved in forming a context pattern varied from three to seven. What can be seen is that the larger the pattern is the higher is the verification accuracy.

By utilizing historical data, as provided by KNs, the context verification mechanism exploits the temporal dimension of knowledge networks. The more reference data is available the better are the verification results. Deciding how many bygone patterns need to be stored, and which is the optimal temporal difference between two patterns are interesting aspects for optimizing the context verification process.

Similarly, forming context groups out of semantically or

spatially related sensors could exploit the semantic and spatial dimensions of KNs's self-organization. This will be an interesting point for future research.

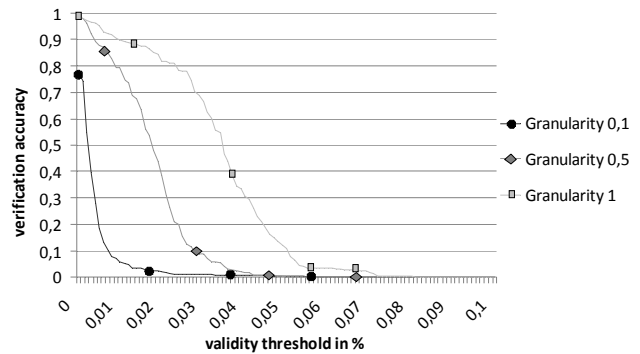


Figure 9: Verification accuracy over validity threshold using different data granularity levels.

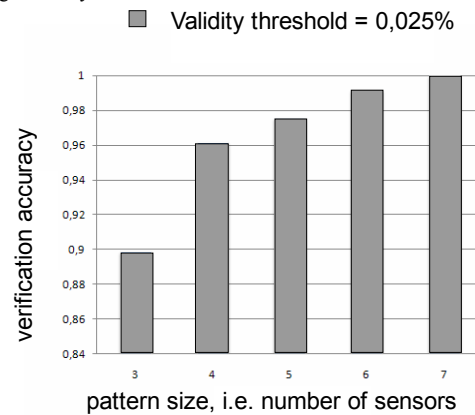


Figure 10: Verification accuracy over context pattern size.

## VII. RELATED WORK

The KNs approach is a new synthesis that grounds on a variety of past research works and relates to some current ones. Of particular relevance are those works on context awareness aimed at providing applications with a flexible common interface to heterogeneous context providers, and those works that apply sorts of data mining approaches to extract relevant patterns (i.e., situations) from contextual data.

### A. Context-awareness

Early works in the area of context-awareness, as from Schmidt et al. [Sch99] and Dey et al. [DeyA00], concentrates on the issue of acquiring context data from sensors and of the processing such data to make it available to processes/services in the form of abstract components. Such approaches have two main shortcomings: one the one hand they do not provide a uniform model and a common semantics to describe the data. This forces developers to build new query languages and new components in dependence of the kind of information at hand. On the other hand they neither address the problem of extracting high-level situations from raw sensor readings nor the problems that of providing application-specific views. The KNs framework overcomes these limitations via a single query interface and by embedding self-organizing mechanisms

to analyze data effectively and extract higher-level knowledge according to any needed view.

Several research works get inspiration from tuple space data models [Ahu86] to represent contextual information in the form of tuples and access them via associative (i.e., pattern-matching based) query operations. The basic idea is that associative access, within a uniform interface, can facilitate semantic-based access to a variety of knowledge sources, possibly enforcing application-specific views. Egospaces [JulR06] adopts this perspective to access contextual information according to user-specific views. However, it does not commit to a specific pre-defined structure for context tuple, which can make it difficult for services to uniformly deal with different aspects of the context represented in different formats. The Context Fabric model [Hon02] relies on well-structured context tuples each describing a single piece of context data in terms of entities (people, place, thing) and attributes (e.g., the name). However, it does not propose solutions for enforcing application-specific views. Recent proposals focusing on sensor networks suggest exploiting a tuple-based approach to flexibly access information on sensor networks [NewW04, MotP06]. Although not focusing on specific tuple structures, such proposals are of interest in that they consider the possibility of providing application-specific views in accessing sensorial data. The idea is to have services dynamically inject code into the sensors for aggregating/elaborating data within the sensor network, and eventually enabling services to directly access aggregated data according to their own specific needs. All these systems have commonalities with the ideas presented in our KNs framework, which however strives for more generality. So, while existing approaches focus on sensor networks or user-centric contextual information, KNs appear a more general-purpose model, suited for diverse devices and scenarios.

A proposal sharing a number of goals with the KNs proposal is the “knowledge plane” approach [Cla03]. There, the idea is to couple the service layer with a (heavyweight) control plane where both tools for the analysis of situational knowledge and sophisticated logic of application control and management are embedded. On the contrary, KNs have the goal of being lightweight, embedding the logics related to information management and only relatively simple logics for their internal unsupervised maintenance. That is, for KNs to be effectively usable, they must rely on simple self-organization algorithms for knowledge management and on simple self-management mechanisms to adapt their internal behavior accordingly.

### B. Pattern and Situation Discovery

The issue of having algorithms and tools to digest large amounts of data in order to transform it into more meaningful knowledge, is at the very centre of data mining research. However, most data mining research focuses on centralized architectures that do not fit our visions, and a few have been focused on distributed network architectures.

Recently, however, data mining approaches have been proposed to analyze large amounts of contextual data and infer hidden linkages, correlations, rules, and constraints in such data [Fay96, Bar03]. In general, all the mechanisms proposed

in this field (and in the wider data mining area) can be potentially employed within the knowledge network layer to extract knowledge from raw data collected by sensing devices.

Sensor networks in prime offer unique challenges and opportunities to distributed data mining, given the potentially large amount of sensors in a network and the consequently large amount of data to be analyzed. Some approaches [BonB05, McS05] focus on mining sensed data for prediction purpose. [BonB05] proposes a framework for data mining upon sensor network for supervised learning (prediction, classification, etc.) based on distributed sensor clustering and aggregation. Similarly, [McS05] proposes a framework for prediction based on the flow of local predictors through the network. Other approaches [GanEH04, KulD05] focus on the general problem of identification of pattern by using distributed AI algorithm. Whatever the case, all these approaches: (i) uphold the need for data mining to analyze the vast amount of data in pervasive computing application, (ii) show that decentralized approaches are effective and operable in distributed network with several nodes. Clearly, such approaches are relevant also to the KNs domain and could be well integrated in the structures of our framework.

Another trend of research in applying distributed data mining to pervasive computing scenarios consists in analyzing data coming from wearable sensors to infer and predict user’s behavior and social interactions. The work presented in [GiP06] applies data mining techniques to automatically identify social structures among a group of people, by making use of radio-based proximity sensors. A similar proposal can be found in [Pen05], where the use of microphones and IR badges is proposed to measure who is talking with whom, and derive social networks and other context information by mining such information. On similar lines, the work on “familiar strangers” [PauG04] records and mine Bluetooth-encounters to identify those people and places that are familiar to us, although we do not know or remember. All these algorithms could be implemented within the KNs and complement the existing self-organizing mechanisms for data analysis. For example, they would be extremely useful in the proposed case study application, where advertisements could be delivered also on the basis of the inferred social relationships of the user.

## VIII. OPEN ISSUES AND RESEARCH DIRECTIONS

Despite our success so far in the design and implementation of the Knowledge Networks Toolkit and of related algorithms for knowledge management, several open issues remain, calling for further research.

As we have already discussed, applications and services need to take advantage from knowledge organization along various dimensions other than semantic and spatial (which we have already somewhat explored), e.g. along the temporal dimension, along any combination of semantic, temporal, and spatial organization, as well as along additional application-specific dimensions.

From the temporal viewpoint, the basic idea is that the analysis (both spatial and semantic) of contextual information about the past can be used to infer information about the future. For instance, the analysis of the fact that a visitor at the

exhibition has already visited specific sections of an exhibition can be used not only to increase the accuracy of its profile but also to reasonably predict what sections/events in the exhibition he is most likely to visit next. Accordingly, one can tune the information displayed on the screens close to her/him. Such predictive knowledge mechanisms – to be grounded on a large body of existing research work on predictive technologies [Pat04] will soon be included in the scope of this research.

Concerning the building and instantiation of application-specific views, the toolkit already provides the possibility of doing that. What is missing is a clear understanding of how we can build application-specific views not simply as additional, stand-alone components having direct access to knowledge atoms only, but rather as well-integrated components that can take advantage of all the other components, and that can enable a multi-level perspective on the available knowledge. With this regard, an interesting open issue relates to the possibility of building semantically-enriched knowledge views where, other than organizing knowledge based on its raw semantic description, it is also possible to concurrently analyze spatial and temporal relations, discover and enact relations among apparently uncorrelated knowledge atoms and, eventually, generate new knowledge about facts and situations. With reference to the application example, one can consider analyzing the activities of a visitor to discover, by properly relating them, more information than those available in his/her personal profile. For instance, by relating the fact that a user walks very slowly, has been to the pharmacy, and has been to the doctor several times, one can detect he is an elderly person. Similarly, by analyzing the patterns of social relations of a visitor (e.g. by regarding the Bluetooth connections of its PDA) one can understand whether this person is accompanied by children or by friends. Such newly generated knowledge can then be used to tune the advertisements displayed in his presence accordingly.

To fulfill the above vision, mechanisms have to be identified that deal with the overall knowledge lifecycle. It is clear that the amount of information relevant for situational knowledge will be rather large, unstructured, unrelated and possibly redundant. This calls for advanced knowledge lifecycle mechanisms, dealing with the key issue of evaluating how long the information should be held and how much of its history should be stored for future use and/or for predictive features. In other words, it is important to develop mechanisms allowing the system to “forget” obsolete information.

As far as knowledge consistency verification is concerned, we intend to address information reliability and accuracy analysis in large-scale KNs, by extending the context verification mechanism we have developed so far to multiple abstraction levels of contextual information, and by studying how verification accuracy can be influenced by the hierarchical application of context verification algorithms. In hierarchical context verification, the mechanism is first applied to the top most level of abstraction, and, only if any inconsistency can be detected, context verification is

subsequently applied to the next lower abstraction level. In the case study, and in particular in relation to the above, introduce situation in which the age of a person has to be detected from his activities, the higher-level concept is that of “elderly”, while the lower-level concepts consist of all the facts that led to the inference of the fact that the concerned person is in fact an elderly person or not, e.g. the person's movement speed and destinations. Propagating context verification from higher to lower context abstraction levels may allow to reduce the computational complexity because not all related data has to be analysed at any given time.

Considering the distributed aspect of KNs, and although considering that the KNs toolkit already supports distribution of its components, the issue of what strategies and approaches to use for e.g. more flexible distribution, advanced caching, replicating and the re-location of knowledge atoms and knowledge containers is yet to be investigated in more detail. We plan to borrow from the lessons of P2P approaches to manage advanced distribution issues [AndS04], and to experience with biologically inspired distribution strategies to provide for better self-adaptation and self-organization [MenCT06].

Finally, from the viewpoint of security, a key problem is to understand how and to what extent services are allowed to access information in KNs and to decide when access should be denied. In particular, based on the fact that specific services may require the construction of specific KNs and the access to specific views on knowledge, one must provide the possibility for services to somehow access the inside of knowledge networks for re-configuration and dynamic instantiation of specific algorithms within. How this can be enforced, with what APIs, and with what security strategies, is still to be fully explored.

## IX. CONCLUSIONS

The increasing deployment of pervasive computing technologies such as sensors, tags, location systems, and user profilers will soon form the basis of a globally shared and distributed infrastructure, producing huge amounts of contextual information for the use of general-purpose pervasive services. However, this also introduces the need for novel models and tools to properly prune, organize and aggregate this growing amount of distributed information, so as to facilitate the successful exploitation thereof by pervasive services.

In this context, self-organizing KNs promise to become a very useful tool. By taking care of managing an increasing amount of contextual information in a fully self-organizing and self-managing way, KNs induce a separation of concerns that facilitates the development of pervasive services and that, at the same time, enables them to reach higher degrees of contextual and situational-awareness.

## X. REFERENCES

- [Ahu86] S. Ahuja, N. Carriero, D. Gelernter, “Linda and Friends”, IEEE Computer, IEEE CS Press, 19(8):26-34, 1986.

- [AndS04] S. Androutsellis-Theotokis, D. Spinellis, "A Survey of P2P Content Distribution Techniques", *ACM Computing Surveys*, 36(4):335-371, 2004.
- [Bar03] I. Bartolini, E. Bertino, B. Catania, P. Ciaccia, M. Golfarelli, M. Patella, and S. Rizzi. "PAtterns for Next-generation DAtabase systems: preliminary results of the PANDA project", Symposium on Advanced Database Systems, Cetraro, Italy, 2003.
- [Bau06] M. Baumgarten N. Biccocchi, M. Mulvenna, F. Zambonelli, "Self-organizing Knowledge Networks for Smart World Infrastructures", International Conference on Self-organization in Multiagent Systems, Erfurt, Germany, 2006.
- [Bel07] F. Bellifemine, G. Caire, D. Greenwood, "Developing Multi-Agent Systems with JADE", Wiley, 2007.
- [Bel08] P. Bellavista, A. Kupper, S. Helel, "Location-based Services: Back to the Future", *IEEE Pervasive Computing*, 7(2):85-89, 2008.
- [Bic07] N. Biccocchi, M. Mamei, F. Zambonelli, "Self-organizing Spatial Regions for Sensor Network Infrastructures", *IEEE Symposium on Pervasive and Ad-Hoc Communications*, Niagara Falls (ON), Canada, 2007.
- [Bic08] N. Biccocchi, G. Castelli, M. Mamei, A. Rosi, F. Zambonelli, "Supporting location-aware Services for Mobile Users with the Whereabouts Diary", International Conference on Mobile Middleware, ACM Press, Innsbruck, Austria, 2008.
- [BonB05] G. Bontempi, Y. Le Borgne, "An adaptive modular approach to the mining of sensor network data", Workshop on Data Mining in Sensor Networks, Philadelphia (PA), USA, 2005.
- [Buc03] T. Buchholz, A. Kupper, S. Schiffrers, "Quality of Context Information: What is it and Why We Need It", HP-OVUA Workshop, Geneva, Switzerland, 2003.
- [Cas07] G. Castelli, A. Rosi, M. Mamei, F. Zambonelli, "A Simple Model and Infrastructure for Context-Aware Browsing of the World", *IEEE Conference on Pervasive Computing and Communication*, New York (NY), USA, 2007.
- [ChoK03] C. Chong, S. Kumar, "Sensor Networks: Evolution, Opportunities, Challenges", *Proceedings of the IEEE*, 91(8): 1247 - 1256, 2003.
- [Cla03] D. Clark, C. Partridge, C. Ramming, J. Wroclawski, "A Knowledge Plane for the Internet", *ACM SIGCOMM Conference*, Karlsruhe, Germany, 2003.
- [Cos07] P. Costa, G. Coulson, C. Mascolo, L. Mottola, S. Zachariadis, "Reconfigurable Component-based Middleware for Networked Embedded Systems", *International Journal of Wireless Information Networks*, 14(2):149-162, Springer, 2007.
- [Cro03] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, A. Warfield, "Plutarch: An Argument for Network Pluralism", *ACM SIGCOMM Workshops*, Karlsruhe, Germany, 2003.
- [DeyA00] A. K. Dey, G. D. Abowd, "Towards a Better Understanding of Context and Context Awareness", Workshop on the What, Who, Where, When and How of Context-Awareness, *ACM Conference on Human Factors in Computer Systems*, Hague, Netherlands, 2000.
- [Dob06] S. Dobson, S. Denazis, A. Fernández, D. Gaiti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, F. Zambonelli, "A Survey of Autonomic Communication", *ACM Transactions on Autonomous and Adaptive Systems*, 1(2): 223 - 259, 2006.
- [Est02] D. Estrin, D. Culler, K. Pister, G. Sukjatme, "Connecting the Physical World with Pervasive Networks", *IEEE Pervasive Computing*, 1(1): 59 - 69, 2002.
- [EugF03] P. Eugster, P. Felber, R. Guerraoui, A. Kermarrec, "The many faces of publish/subscribe". *ACM Computing Surveys*, 35(2), 114-131, 2003.
- [Fay96] U. Fayyad, G. Piatesky-Shapiro, P. Smith, "Advantages in knowledge discovery and data mining", *Data Mining to knowledge Discovery: An Overview*, AAAI/MIT Press, 1996.
- [GanEH04] D. Ganesan, D. Estrin, J. Heidemann, "DIMENSIONS: Why do we need a new Data Handling architecture for Sensor Networks?", *Information Processing in Sensor Networks*, Berkeley (CA), USA, 2004..
- [GiP06] J. Gips, A. Pentland, "Mapping Human Networks", *Conference of Pervasive Computing and Communications*, Pisa, Italy, 2006.
- [HigB01] J. Hightower, G. Borriello, "Location Systems for Ubiquitous Computing", *IEEE Computer*, 34(8): 57-66, 2001.
- [Hof07] E. Höfig, "Autonomic Reliable Multicast: Application-Level Group Communication Using Self-Organization Principles", *International Conference on Bio-Inspired Models of Network, Information, and Computing Systems*. Budapest, Hungary, 2007.
- [Hon02] J. Hong., "The Context Fabric: An Infrastructure for Context-Aware Computing", *Conference on Computer Human Interaction*, Minneapolis (MN), USA, 2002.
- [Hon08] D. Hong et al., "Advances in Tangible Interactions and Ubiquitous Virtual Reality", *IEEE Pervasive Computing*, 6(2):90-96, 2008.
- [JulR06] C. Julien, G. Roman, "EgoSpaces: Facilitating Rapid Development of Context-aware Mobile Applications", *IEEE Transactions on Software Engineering*, IEEE CS Press, 32(5):281-298, 2006.
- [Kal03] Y. Kalfoglou, M. Schorlemmer, "Ontology mapping: the state of the art", *The Knowledge Engineering Review* 18(1):1-31, 2003.
- [KulD05] A. Kulakov, D. Davcev, "Data mining in wireless sensor networks based on artificial neural-networks algorithms", *Workshop on Data Mining in Sensor Networks*, Newport Beach (CA), USA, 2005.
- [McS05] S. M. McConnell, D. B. Skillicorn, "A Distributed Approach for Prediction in Sensor Networks", *Workshop on Data Mining in Sensor Networks*, Newport Beach (CA), USA, 2005.
- [MenCT06] R. Menezes, A. Charles, R. Tolksdorf, "On the Implementation of SwarmLinda", *ACM Southeastern Conference*, Huntsville (AL), USA, 2006.
- [MotP06] G. Mottola, G. P. Picco, "Logical Neighborhoods: A Programming Abstraction for Wireless Sensor Networks", *Conference on Distributed Computing in Sensor Systems*, San Francisco (CA), USA, 2006.
- [NewW04] R. Newton, M. Welsh, "Region Streams: Functional Macroprogramming for Sensor Networks", *Workshop on Data Management for Sensor Networks*, Toronto, Canada, 2004.
- [Pat04] D. Patterson, L. Liao, K. Gajos, M. Collier, N. Livic, K. Olson, S. Wang, D. Fox, H. Kautz, "Opportunity Knocks: a System to Provide Cognitive Assistance with Transportation Services", *Conference on Ubiquitous Computing*, Nottingham, United Kingdom, 2004.
- [PauG04] E. Paulos, E. Goodman, "The familiar stranger: anxiety, comfort, and play in public places", *Conference on Human Factors in Computing Systems*, Vienna, Austria, 2004.
- [Pen05] A. Pentland, T. Choudhury, N. Eagle, P. Singh, "Human dynamics: computation for organizations", *Pattern Recognition Letters*, Elsevier, 26(4):503-511, 2005.
- [Phi04] M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, D. Fox, H. Kautz, "Inferring Activities from Interactions with Objects", *IEEE Pervasive Computing*, 3(4):50- 57, 2004.
- [Sch99] A. Schmidt , K. A. Aidoo , A. Takaluoma , U. Tuomela , K. Van Laerhoven , W. Van de Velde, "Advanced Interaction in Context", *Symposium on Handheld and Ubiquitous Computing*, Karlsruhe, Germany, 1999.
- [Ser04] J. Serrat, J. Serrano, R. Marín, A. Galis, K. Yang, D. Raz, Efstathios D. Sykas, "An Approach to Context Aware Services", *NOMS2004*, Seoul, South Korea, 2004.
- [UliG07] M. Ulietu, S. Grobbelaar, "Engineering Industrial Ecosystems in a Networked World", *5th IEEE International Conference on Industrial Informatics*, IEEE Press, Vienna, Austria, 2007.
- [Wan06] R. Want, "An Introduction to RFID Technology", *IEEE Pervasive Computing*, 5(1) :25-33, 2006.

**Nicola Bilocchi** is a PhD student in Computer Science at the University of Modena and Reggio Emilia. He received the Laurea degree in Computer Engineering from the University of Modena and Reggio Emilia in 2004. His current research interests include: computer networks, pervasive computing technologies, data mining for behavioral prediction and activity-based computing.

**Matthias Baumgarten** is currently working as a research fellow in the Faculty of Informatics at the University of Ulster. His main research interests are in the field of data mining in particular web mining, pattern discovery and personalization techniques. He has received his PhD in 2005 and has (co) authored about 25 publications

**Nermin Brgulja** is a PhD Student in the department of Computer Science at the University of Kassel, since April 2005. He received his B.S. and M.S. in Electrical Engineering from University of Kassel in 2003 and 2005. His current research interests include: context awareness, pervasive computing technologies and autonomic communications.

**Rico Kusber** is a researcher in Computer Science at the University of Kassel, Chair for Communication Technology, since February 2006. He received his degree in Computer Science from the Chemnitz University of Technology in January 2006. During his study he specialized on artificial intelligence. His current research interests comprise artificial immune systems, autonomic computing and communication, and autonomic decision making processes

**Marco Mamei** is a research associate in Computer Science at the University of Modena and Reggio Emilia, since September 2004. He received the Laurea degree in Computer Engineering from the University of Modena and Reggio Emilia, in 2001 and the PhD in Computer Science from the same University in 2004. His current research interests include: pervasive computing technologies, data mining and data analysis for behavioral prediction, amorphous computing, and autonomic communications.

**Maurice Mulvenna** is a senior lecturer in Computing Science at the University of Ulster's School of Computing and Mathematics in the UK. He has published over 120 papers in the area of computer science, addressing both theory and practice. He is a senior member of both IEEE and ACM and a chartered member of the British Computer Society.

**Franco Zambonelli** is professor in Computer Science at the University of Modena and Reggio Emilia since 2001. He obtained the Laurea degree in Electronic Engineering in 1992, and the PhD in Computer Science in 1997, both from the University of Bologna. His current research interests include: distributed and pervasive computing, autonomic computing and communication, software engineering for large-scale agent and service systems. In these areas, he has published 1 monograph, over 50 papers in international peer-reviewed journals, co-edited 7 books, and received several best paper awards. He is currently Scientific Manager for the EU FP6 IP Project "CASCADAS", where he also plays the role of leader for the "Knowledge Networks" WP.