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, *Senior Member, IEEE*, and 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 becoming more 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 them 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" (KNs), 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. First, with the help of a simple case study, we motivate the need to evolve from models of "context awareness" toward models of "situation awareness" via proper self-organized "KN" tools, and we introduce a general reference architecture for KNs. Second, we describe the design and implementation of a KN toolkit that we have developed, and we exemplify and evaluate algorithms for knowledge self-organization integrated within it. Open issues and future research directions are also discussed.

Index Terms—Context awareness, knowledge networks (KNs), pervasive computing, self-organization.

I. INTRODUCTION

PERVASIVE computing researchers envision a future world in which computing and sensing devices will be embedded everywhere and will be able to produce digital information about almost every event occurring in the physical world [12], [44]. Such information can then be exploited at the user level to deliver services for better perceiving/interacting with the physical world [27], as well as to improve the capability of

N. Bicocchi, M. Mamei, and F. Zambonelli are with the Dipartimento di Scienze e Metodi dell'Ingegneria (DISMI), Università di Modena e Reggio Emilia, 42100 Reggio Emilia, Italy (e-mail: nicola.bicocchi@unimore.it; marco.mamei@unimore.it; franco.zambonelli@unimore.it).

M. Baumgarten and M. Mulvenna are with the School of Computing and Mathematics, University of Ulster, BT37 0QB Belfast, U.K. (e-mail: M.Baumgarten@ulster.ac.uk; md.mulvenna@ulster.ac.uk).

N. Brgulja and R. Kusber are with ComTec, University of Kassel, 34121 Kassel, Germany (e-mail: nermin.brgulja@comtec.eecs.uni-kassel.de; rico.kusber@comtec.eecs.uni-kassel.de).

dynamically and autonomously adapting services to the context in which they are invoked.

Paving the way for the full realization of the pervasive computing vision requires the following: 1) technologies to capture digital contextual information and 2) services being able to meaningfully access this information.

With regard to the former point, the road is already being paved. In the past few years, we have witnessed an increasing deployment of sensors [19], Radio Frequency IDentification (RFID) tags [41], [45], location systems [6], [24], and automatic user profilers [8] that, together with the possibility of accessing from the Web a large variety of data items and facts about the world [12], 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 enrich this growing amount of distributed information to make it more meaningful and, consequently, more understandable [4]. In particular, we believe that there must be the following evolution:

1) from a model of *context awareness* [17], [37] in which the focus is to provide services with simple interfaces to access heterogeneous context providers, leaving to them the burden of understanding the retrieved information;
2) toward 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 predigested 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). 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 [44]. Thus, KNs may support services by allowing them to reach, with reduced efforts, a comprehensive understanding of "situations" and, consequently, to accomplish a higher degree of adaptability and autonomicity. In this context, the key contributions of this paper are the following:

1) to motivate the need to evolve from models of "context awareness" toward models of "situation awareness" via proper self-organized "KN" tools and to introduce a general reference architecture for KNs;

2) to describe the design and implementation of a KN toolkit that 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 II motivates the need for pervasive services to bridge the gap between context and situation awareness and introduces a case study to clarify the concept. Section III introduces the key concepts of KNs and sketches a reference architecture for them. Section IV describes the current prototype implementation of the KN toolkit that has been implemented in the context of the European Project "CASCADAS." Section V presents some representative knowledge aggregation and knowledge management algorithms currently integrated in the KN toolkit. Section VI presents experiments and performance measures to evaluate the KN architecture and its algorithms. Section VII discusses related works. Section VIII outlines open issues and future research directions. The conclusions is drawn in Section IX.

II. FROM CONTEXT TO SITUATION AWARENESS

A. Motivation

According to most assessed user-centric definitions [10], [17], "*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 in improving the interaction between such an entity and its users.

Further, software-centric viewpoints focusing on context awareness as a means for services to improve quality and reliability via autonomicity and adaptability have emerged [16], [18], [43].

Despite committing to the aforementioned definition, 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, particularly in the presence of an overwhelming amount of data and a lack of relations between them.

The imminent availability of pervasive technologies such as sensor networks [13], [19], RFID tags [41], [45], localization tools [6], [24], and smart grids [11] will soon make pervasively available an incredible amount of real-time information about the physical world, its processes, and its objects. Furthermore, the widespread success of participatory Web 2.0 tools is enriching the Web with information of any kind. 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 [12].

Overall, the aforementioned trends are increasing 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 aforementioned information calls for the following:

1) notable communication efforts to retrieve, from a variety of diverse devices (and possibly from remote sources), all needed information;

2) notable computation efforts to analyze available information, with the goal of making it more meaningful (i.e., associated to situations) and ultimately machine understandable.

B. 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 scenario, it is realistic to assume the presence of a pervasive infrastructure of embedded devices such as sensors of various types, several 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 and generate 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 revenue, 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 using a simple method that is independent from 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 the following: 1) thousands of possibly incomplete user profiles that have to be synchronized with statistical information available elsewhere or with some information extracted from other sources; 2) a multitude of sensorial data detailing what users are currently doing; and 3) 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, for example, 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 available information, 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 awareness.

III. KN APPROACH

To make contextual information meaningful and useful, some tools must be made available to pervasive services that can properly correlate and predigest contextual information so as to provide them with a higher level understanding of surrounding
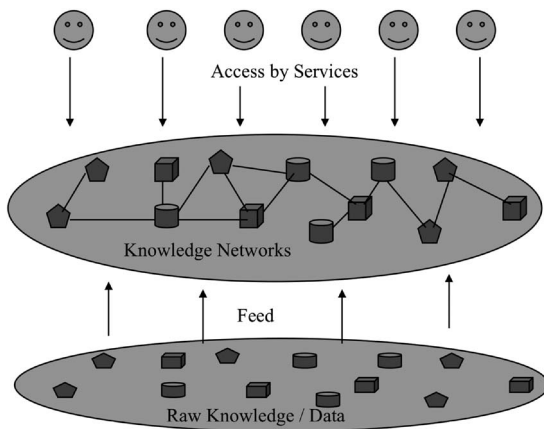
Fig. 1.   KN approach.

situations, without forcing them to access and manage large amounts of data internally.

A. KNs 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 collecting data items, analyzing them, and building 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 to enable them to immediately act on this knowledge. For instance, one can think of aggregating individual data items describing users with similar interests. This can define a new, higher level, and aggregated data item eventually representing, in a compact way, the overall situation of users near 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, that "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 in 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. 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 defines a distributed "ecosystem" populated with data items that autonomously interact, aggregate, and self-establish networks of relations with each other to generate higher level knowledge.

Our idea of KNs is fully in line with the aforementioned perspective (see Fig. 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 KN 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 KN level. Although this may be true to some extent,

one should consider the following: 1) 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 2) 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.

B. Reference Architecture for KNs

It is important to define a reference architecture from which to implement our idea. Let us start with the assumption that there are various kinds of "sensors" (whether physical, software, or social Web 2.0 sensors) generating large amounts of (mostly) independent atomic units of contextual information (see Fig. 1). We can call these as "knowledge atoms." The KN 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 coexist 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 that one wants to deploy. For instance, in the case study, a service that is used to display commercials may be more interested in the gender distribution in order to decide whether to advertise ties or perfumes, while a service that is used to display information about cultural events may be more interested in the education distribution in order to decide whether to inform about a poetry lecture or about an on-going comedy show.

Obviously, it is not possible 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 the following.

1) *Semantic dimension*, 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 and service-orientated computing or for knowledge related to inferring users' activities from a variety of heterogeneous sensorial information.

2) *Spatial dimension*, 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, for example, 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
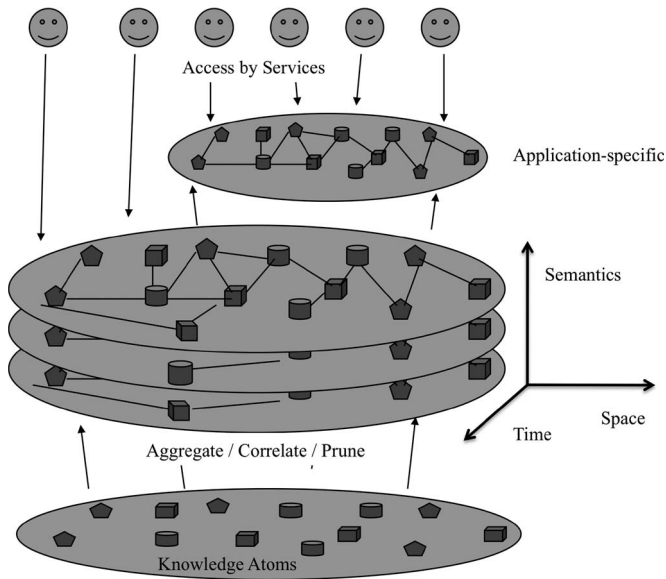
Fig. 2. Conceptual reference architecture for KNs.

of spatially distributed P2P structures to distribute knowledge across a network and to facilitate access to knowledge (as in the case of, for example, knowledge brokers).

3) *Temporal dimension*, 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 and then analyzing and extracting new knowledge in the form of a KN expressing the most likely future situation.

These considerations are summarized into the conceptual reference architecture for KNs, which is shown in Fig. 2. The figure also shows that KNs can also be organized around additional application-specific dimensions.

C. KN 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, KN components can wrap existing context providers, allowing them to integrate into the KNs. On the other hand, it is simple to write proxy components to access KN services from other applications. KN components of our prototype implementation are already provided with flexible application programming interface (API) that can be 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 nonlayered way. Indeed, pervasive and autonomic applications will need context information at all levels of the ISO-OSI stack (from low-level data that are useful for packet routing in the network to high-level information describing the user profile for contextualized advertisement) to properly tune their behavior.

To fulfill the KN goals both in the short and long terms, self-organization is a fundamental property that must be included within the core of the KN 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 KN framework makes 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 peer-to-peer protocols to aggregate distributed data).

IV. KN TOOLKIT

In this section, we present the key features of the prototype KN 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 comprising the KN toolkit, and its overall architecture and functioning. The specific algorithms for knowledge management currently integrated in the toolkit are described later on in Section V.

A. Autonomic Communication Elements

The CASCADAS project (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 *Autonomic Communication Element* (*ACE*), intended as a unifying software engineering abstraction for the development of component-based distributed services. The ACE model has been implemented in a toolkit released by the CASCADAS consortium, and it is available as an open source.[1] 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 [5], [15], [32]. In particular, the following is true about ACE.

1) ACEs are able to run both on high-end computers as well as on tiny devices like sensors, due to their light and modular internal structure. Moreover, ACEs will eventually 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 with individual ACEs or ACE classes and on an internal control loop that can enable self-monitoring and self-adaptation. Specifically, the

---

[1]http://sourceforge.net/projects/acetoolkit

service is comodeled by a plan providing an explicit and machine-processable representation of the actions the ACE will undertake (more concretely, this is an 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: 1) Plan generation or modification is possible without intervening at the code level, but only writing the plan representation. 2) The plan can be supervised since it describes the operations to be performed from a high-level point of view (as a finite-state automaton). 3) Code (functionalities) is structured by means of plug-and-play individual activities.

3) The ACE components are provided with two communication mechanisms. The first is a distributed publish-subscribe mechanism to advertise and look for services offered by other ACEs using semantic descriptions [20]. This is called the goal needed/goal achievable (GN/GA) 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 [5], [15], [20]. The second is a direct message-passing mechanism, allowing for flexible contracting service usage. This mechanism supports bilateral and multilateral communication along previously defined connection partners and allows various properties to be implemented, such as encryption or fixed-number-of-participants constraint [25].

The combined use of all of the aforementioned 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 nearby 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, 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 demand. 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 KN toolkit by focusing on architectural and algorithmic issues only. In particular, we exploited ACE features to do the following.

1) Deal with a broad range of data sources. KN components, based on the ACE model, can be deployed on a number of heterogeneous platforms.

2) Deal with the unreliable data sources. KN components can flexibly self-configure their behavior (e.g., modify the sampling rate in a sensor) by acting on their 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 KN components, discovering each other in a flexible decentralized way.

However, it is important to emphasize that implementing KNs in terms of ACEs and making KNs available as an ACE-based service is not the only possible choice. 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 Reconfigurable Ubiquitous Networked Embedded Systems [15], or agent-based frameworks, such as Java Agent DEvelopment Framework [5], could well serve the KN purposes.

B. KN Basic Components

The implementation of the KN toolkit relies on two basic classes of components, namely, *knowledge atoms* and *knowledge containers* [4], 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 that is independent of its type, size, or context. This is required to provide a generic access to knowledge from within the KN as well as from services and components that are outside, which are independent from the specific characteristics of the data source (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 to the creation and maintenance of the knowledge atom itself. This makes each knowledge atom fully self-descriptive, and as such, it provides information that is 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 that is 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
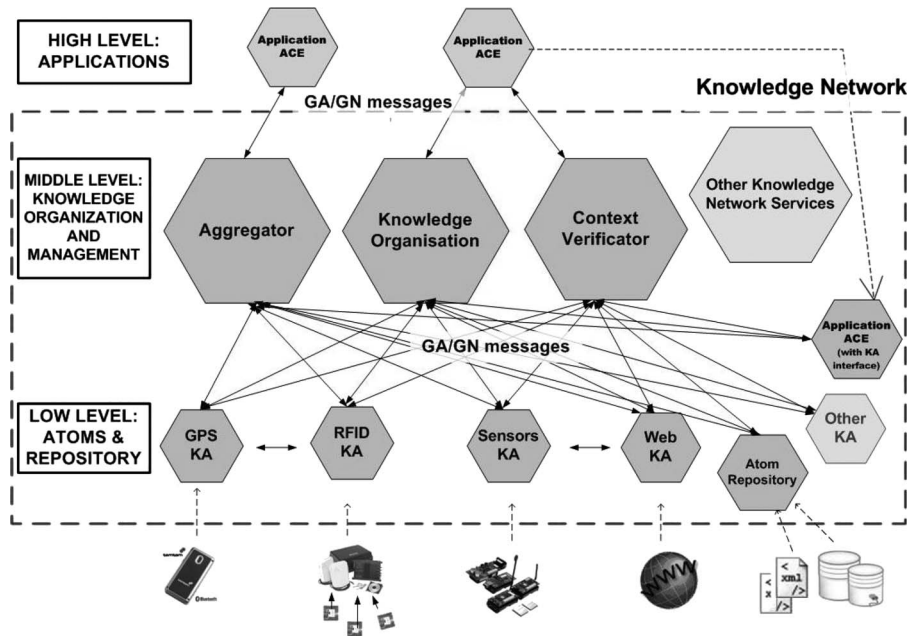
Fig. 3. Architecture of the KN toolkit.

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 of the future position of a user. In general, different configurations of knowledge atoms and containers enable services to organize and access contextual information according to specific application-dependent views.

C. KN Toolkit Architecture

The overall architecture of the implemented ACE-based KN toolkit is shown in Fig. 3. 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 the following: atoms for connecting to GPS devices and to CrossBow Micaz sensors, atoms for accessing system properties in computational devices, atoms for accessing Web information, and generic knowledge atoms for hosting static (preloaded) and historical information. Also, we emphasize that any application-level service realized via an ACE can, by simply implementing a generic knowledge atom interface, publish information/knowledge into the scope of the KN and thus become a part of the population of knowledge atoms, as shown at the bottom of Fig. 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 an "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 KNs. The KN toolkit does not prescribe what knowledge containers should be instantiated at this level nor does it limit the number or type of components that can be there. Depending on the needs of specific applications, new knowledge containers can be defined and allocated at run time to provide specific knowledge management functionalities, specific aggregation functions, and specific knowledge views. 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, to produce an average profile.

All knowledge container components, which are 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 that we have implemented provide "*à la Linda*" [1] 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 basic containers applying simple functions (e.g., average, maximum, and minimum) to an ensemble of knowledge atoms (as it can be the case for the user profiles in the case study). Moreover, these also include algorithms that are used to facilitate advanced models of semantic knowledge organization and of spatial knowledge aggregation, as well as advanced models for knowledge consistency verification. Some of these algorithms are described in more detail in Section V.

It is also worth noticing that, although the architecture is presented as a layered one, 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 the following 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 V-A.

2) Links between knowledge atoms. These links allow us to organize and represent relationships among atoms and possibly instantiate knowledge containers. We used them in Section V-B to spatially self-organize distributed information atoms.

KN components can execute in distributed environments. For example, knowledge atoms can be deployed to the data sources (e.g., PDAs or sensors), while knowledge containers can be deployed to other nodes of the network. In our case study, a knowledge container devoted to user profile aggregation is executed on the advertisement screen, while another knowledge container dealing with security monitoring is executed on the central server of the security office.

V. KN Algorithms

In this section, we illustrate some of the algorithms for knowledge management and analysis that have been integrated within the KN toolkit. We recall that the current architecture of the toolkit enables the integration, even at run time, of new components to deal with specific knowledge management issues and/or to build specific knowledge views.

A. Semantic Self-Organization

Semantic knowledge self-organization plays a key role in KNs. In fact, it is able to foster the universal use and exchange of data, information, and knowledge at various levels of detail.

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

In relation to the case study introduced earlier, an example description of semantic keywords is shown in Fig. 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 nearby advertisement screens. For example, the keywords shown in Fig. 4 reflect a simple shopping profile of a middle-aged woman that is interested in hats, handbags, and perfumes. Notable here is that a descriptive information may include various keywords as well as the relations with each other. This provides relevant information in constructing individual ontologies in a bottom-up fashion. Furthermore, a similar model may also be used to

```
<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>
```

Fig. 4.   Example of a semantic description.

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 50 m of the advertisement screen or nearby Times Square, etc.). Thus, the semantic information may be categorized, first, to be of either static or dynamic nature and, second, 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. This 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 KN. Atoms will be registered to the containers they support, thus forming individual clusters based on the semantic description 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 shown 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 KN infrastructure. The KN 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 change its structure on the basis of the information knowledge atom being produced.
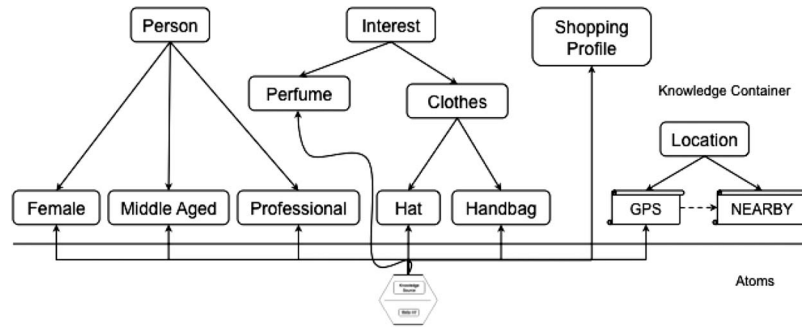
Fig. 5. KN ontology.

The structure derived from the example of Fig. 4 is shown in Fig. 5. Three individual ontologies (not counting "shopping profile") have been constructed. Each of them reflects an individual concept and is represented by a knowledge container (note that the "location" ontology can be relevant for spatial organization, as discussed in the next section). If strictly hierarchical, the resulting construct is comparable to an ontology, which, in the case of KNs, is always shared among the entire network. No part of any given construct is created or used only for a single object (i.e., atom or container), but it shared among all available objects that are within the same organizational space. Although bottom-up construction maximizes flexibility and avoids the need of defining overarching ontologies, 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. Unfortunately, it is often difficult to find the proper tradeoff between these two extremes [29].

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. Moreover, 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 [7]. 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 reaches 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 (see Fig. 6).

Then, a set of knowledge containers is instantiated in order to represent the network as if it was composed of several macro sensors. In this way, the possibly large amount of sensor data generated by the network cannot be necessarily perceived as a combination of unrelated information. Instead, the algorithm makes it possible to perceive the sensor network as if it was made up of a more limited number of "macro sensors," each associated to a well-characterized region of the physical environ-
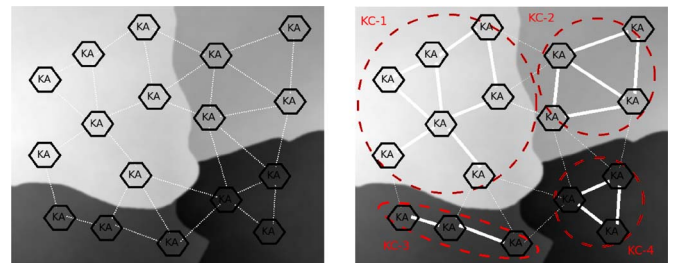

Fig. 6. (Left) Four recognizable regions of an environment, as identified by a specific property, and a network of knowledge atoms immersed on it. (Right) Knowledge atoms spatially self-organize into four virtual overlays, reflecting the environment. Each region is then represented by a single knowledge container acting as a sort of macro sensor for that region.

ment. To some extent, the algorithm provides for the automatic construction of 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 in its region. In this way, the KN is not only a way to perceive the environment in a simplified fashion, but it can also act as a computing layer that is able to produce the derived knowledge.

In the case study, one could think of realizing a similar aggregation algorithm that first 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 is 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 (sensor networks) and in a centralized solution (Web-based repository), services can dynamically decide how to access the 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 her 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 and 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 close to an advertisement screen or if she is entering a specific shop. Other forms of organization could rely on evaluating the surroundings of the person's location, on creating a virtual orb around the person's position, and on 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 managing large amounts of data, reliability is determinant. Inducing incorrect data into a self-organizing structure like a KN leads to error propagation. Although not all errors can be prevented within KNs, algorithms that are used to recognize a faulty information are needed. We developed a *context verification* mechanism and integrated it into the KN toolkit.

By context verification, we refer to a process that examines contextual information and decides, based on historical data, whether this information is valid or suspicious. To classify 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 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* is associated with one or more context patterns. Different context verifiers can coexist within one or more KNs.

Our context verification method is based on statistics. Depending on the frequency that a given pattern has been observed within a body of the historical data, the verifier classifies it as 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. If a reading is far from the context pattern (i.e., its probability is below the *validity threshold*), the system can mark it as suspicious. We also developed a measure, called data *granularity level*, for numeric values. That measure represents the distance between two neighboring values within the value range of a sensor (e.g., a data granularity level of one means that the values are rounded off to the next whole number instead of using accurate real numbers). We define as *verification accuracy* the percent of patterns correctly classified. It 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 KN framework. Here, we present experiments related to some of the main aspects of the KN approach, i.e., 1) results illustrating performances of the implemented KN toolkit; 2) results analyzing one of the self-organizing algorithms that we have in KNs; and 3) results from the context verification mechanisms.

A. Infrastructure

We performed experiments to show the performance of the presented toolkit. Specifically, we measured the response times of the KNs under different conditions. We deployed several atom repositories either local or connected through the Internet. The remote repositories were installed in European research facilities provided by members of the CASCADAS project. We ran queries from different networks and monitored response times of the networked KNs.

Fig. 7 (top) shows the response times obtained from local KNs. In detail, we run queried several times *all* the knowledge atoms in the local KNs, and we averaged the response times. As expected, the results are fairly regular and predictable. Still, some remarks can be made. Specifically, the linear increase in response time highlights that the system's performance does not deteriorate dramatically at a certain scale. Furthermore, these tests illustrate a boundary scenario in which all the knowledge atoms are queried. However, under average conditions, only a fraction of atoms are queried (i.e., those referenced by a given knowledge container or knowledge view), allowing a better performance. Finally, our current implementation has not been optimized. For example, the adoption of dedicated indexes would dramatically accelerate the query process.

Fig. 7 (bottom) shows a similar experiment in a distributed setting. The experiment consisted of 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 that it can have large fluctuations because of the component workload and network delays.

B. Spatial Self-Organization

To quantitatively evaluate one of our self-organizing mechanisms, we focused on the most fundamental *spatial* self-organization algorithm. In particular, we conducted some experiments both in a simulation environment using the KN 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 shows 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 propagate 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 [7].

In Fig. 8, we summarized the performance of the proposed spatial self-organizing algorithm. In particular, we measured the aggregated errors produced by the approach while partitioning a network with different sizes and densities. Fig. 8 (top) shows that the average error tends to decrease, increasing the network density, because the number of interactions between knowledge atoms increases as well. More interactions produce a higher convergence speed. Also, the number of messages being exchanged, and thus the communication costs, increases.
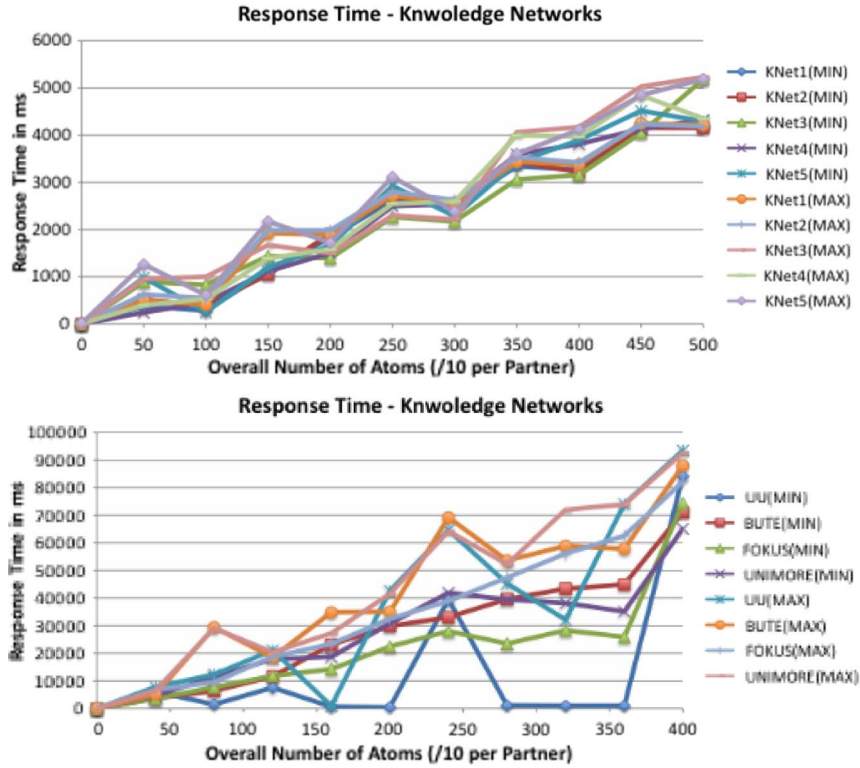
Fig. 7. (Top) Minimum and maximum response times among several local KNs. (Bottom) Minimum and maximum response times among several KNs scattered though Europe. UU—University of Ulster, U.K. BUTE—Budapest University of Technology and Economics, Hungary. FOKUS—Fraunhofer Institute, Germany. UNIMORE—University of Modena and Reggio Emilia, Italy.
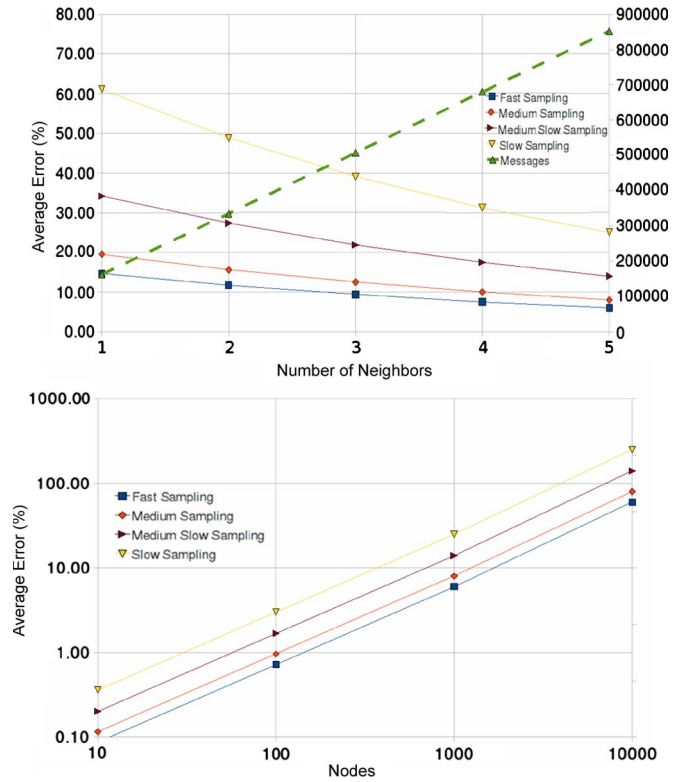

Fig. 8. (Top) Average error for different network densities. (Bottom) Average error for different network sizes.

Fig. 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 has been observed that decreasing the sensor sampling rate slowly degrades performances without altering the overall behavior. These experiments show the effectiveness of the self-organizing algorithm that 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.

Fig. 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 is, 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 an acceptable verification accuracy.

Fig. 10 shows how accuracy is influenced by the size of the context pattern. The validity threshold has been fixed at 0.025% through a set of empirical experiments showing optimal results for this value. The number of sensors collecting the context pattern ranged from three to seven. The results show
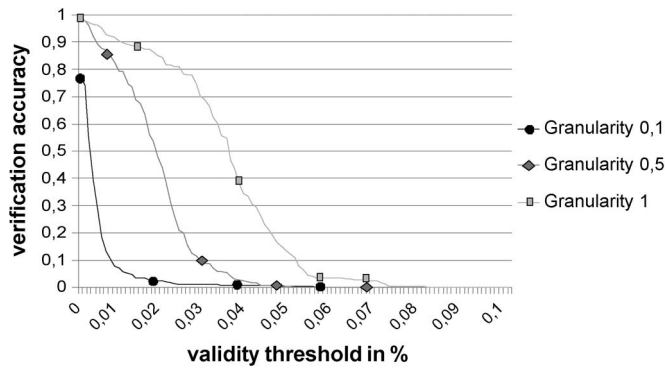
Fig. 9. Verification accuracy over validity threshold using different data granularity levels.
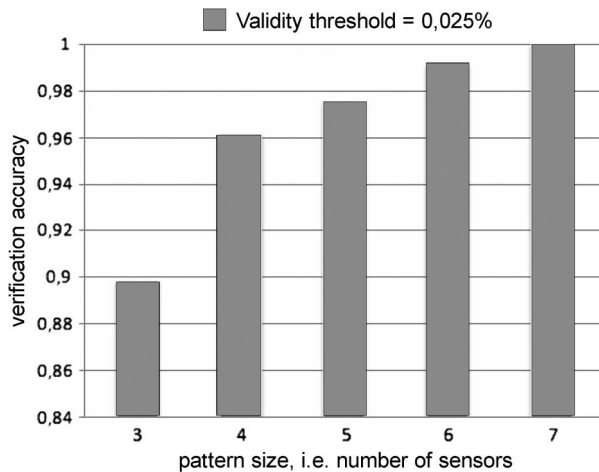

Fig. 10. Verification accuracy over the context pattern size.

that increasing the size of the context pattern increases (almost linearly) the verification accuracy.

By utilizing historical data, as provided by KNs, the context verification mechanism exploits the temporal dimension of KNs. The more reference data are available, the better are the verification results. Deciding how many previous patterns need to be stored and which optimal temporal difference between two patterns is an interesting aspect in 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' self-organization. This will be an interesting point for future research.

VII. RELATED WORK

The KN approach is a synthesis based on several research works. Specifically, relevant contributions come from the following: 1) works in the area of context awareness, providing applications with a common interface to heterogeneous context providers, and 2) works in the area of data mining, extracting relevant patterns from contextual data.

A. Context Awareness

Early works in the area of context awareness (Schmidt *et al.* [42] and Dey *et al.* [17]) concentrate on the issue of acquiring context data from sensors and processing such data to

make it available to processes/services in the form of abstract components. Such approaches have two main shortcomings: On the one hand, they do not provide a uniform model and common semantics to describe the data. This forces developers to build new query languages and new components depending on the information at hand. On the other hand, they address neither the problem of extracting high-level situations from raw sensor readings nor the problems of providing application-specific views. The KN 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 [1] to represent contextual information in the form of tuples and to 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 [28] adopts this perspective to access contextual information according to user-specific views. However, it does not commit to a specific predefined 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 [26] relies on well-structured context tuples, each describing a single piece of context data in terms of entities (e.g., people, places, and things) 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 [35], [36]. 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 sensor 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 of these systems have commonalities with the ideas presented in our KN framework, which however strives for more generality. Therefore, while existing approaches focus on sensor networks or user-centric contextual information, KNs appear as a more general-purpose model, suited for diverse devices and scenarios.

A proposal sharing a number of goals with the KN proposal is the "knowledge plane" approach [14]. Here, 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 development of algorithms that are able to process large bodies of data and that are able to extract from them meaningful knowledge is at the core of data mining and knowledge

discovery research. However, most data mining research focuses on centralized architectures that do not fit our vision, 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 [3], [21], [42]. In general, all the mechanisms proposed in this field (and in the wider data mining area) can be potentially employed within the KN layer to extract knowledge from the raw data collected by sensing devices.

Sensor networks offer unique challenges and opportunities for 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 [9], [33] focus on mining sensed data for prediction purpose. The authors in [9] propose a framework for data mining upon sensor network for supervised learning (prediction, classification, etc.) based on distributed sensor clustering and aggregation. Similarly, the authors in [33] propose a framework for prediction based on the flow of local predictors through the network. Other approaches [22], [30] focus on the general problem of identification of pattern by using a distributed AI algorithm. Whatever the case is, all of these approaches do the following: 1) uphold the need for data mining to analyze the vast amount of data in pervasive computing application and 2) show that decentralized approaches are effective and operable in distributed network with several nodes. Clearly, such approaches are relevant also to the KN 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 of analyzing data coming from wearable sensors to infer and predict user's behavior and social interactions. The work presented in [23] 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 [40], where the use of microphones and IR badges is proposed to measure who is talking with whom and to derive social networks and other context information by mining such information. Similarly, the work on "familiar strangers" [39] records and mines Bluetooth encounters to identify those people and places that are familiar to us. All of these algorithms could be implemented within the KNs and could 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 KN 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 of a 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; and 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 [38] 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 multilevel perspective on the available knowledge. With this regard, an interesting open issue relates to the possibility of building semantically enriched knowledge views 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 more information than those available in his/her personal profile. For instance, by relating the fact that a user has walked very slowly, has been to the pharmacy, and has been to the doctor several times, one can detect that 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 aforementioned vision, mechanisms that are used to manage the knowledge lifecycle have to be developed. It is clear that the amount of information that is 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 that allow 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 that 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 topmost level of abstraction. Context verification is subsequently applied to the lower level if, and only if, any inconsistency can be detected. In the case study, we introduce a 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.

Propagating context verification from higher to lower context abstraction levels may allow the reduction of the computational complexity because not all related data have to be analyzed at any given time.

Considering the distributed aspect of KNs and although considering that the KN toolkit already supports distribution of its components, the issue of what strategies and approaches to use for, for example, 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 [2] and to experiment with biologically inspired distribution strategies to provide for better self-adaptation and self-organization [34].

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 access the internal mechanisms of KNs for reconfiguration and dynamic instantiation of specific algorithms from within. Despite the fact that we are aware of these issues, how this can be enforced, with what APIs and security strategies, is still to be fully explored [31].

IX. CONCLUSION

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.

REFERENCES

[1] S. Ahuja, N. Carriero, and D. Gelernter, "Linda and friends," *Computer*, vol. 19, no. 8, pp. 26–34, Aug. 1986.

[2] S. Androutsellis-Theotokis and D. Spinellis, "A survey of P2P content distribution techniques," *ACM Comput. Surv.*, vol. 36, no. 4, pp. 335–371, Dec. 2004.

[3] 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," in *Proc. Symp. Adv. Database Syst.*, Cetraro, Italy, 2003.

[4] M. Baumgarten, N. Bicocchi, M. Mulvenna, and F. Zambonelli, "Self-organizing knowledge networks for smart world infrastructures," in *Proc. Int. Conf. Self-Organization Multiagent Syst.*, Erfurt, Germany, 2006.

[5] F. Bellifemine, G. Caire, and D. Greenwood, *Developing Multi-Agent Systems With JADE*. New York: Wiley, 2007.

[6] P. Bellavista, A. Kupper, and S. Helel, "Location-based services: Back to the future," *Pervasive Comput.*, vol. 7, no. 2, pp. 85–89, Apr.–Jun. 2008.

[7] N. Bicocchi, M. Mamei, and F. Zambonelli, "Self-organizing spatial regions for sensor network infrastructures," in *Proc. IEEE Symp. Pervasive Ad-Hoc Commun.*, Niagara Falls, ON, Canada, 2007.

[8] N. Bicocchi, G. Castelli, M. Mamei, A. Rosi, and F. Zambonelli, "Supporting location-aware services for mobile users with the where-abouts diary," in *Proc. Int. Conf. Mobile Middleware*, Innsbruck, Austria, 2008.

[9] G. Bontempi and Y. Le Borgne, "An adaptive modular approach to the mining of sensor network data," in *Proc. Workshop Data Mining Sens. Netw.*, Philadeplhia, PA, 2005.

[10] T. Buchholz, A. Kupper, and S. Schiffers, "Quality of context information: What is it is and why we need it," in *Proc. HPOVUA Workshop*, Geneva, Switzerland, 2003.

[11] I. Carreras, D. Miorandi, R. Saint-Paul, and I. Chlamtac, "Bottom-up design patterns and the energy web," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 4, pp. 815–824, Jul. 2010.

[12] G. Castelli, A. Rosi, M. Mamei, and F. Zambonelli, "A simple model and infrastructure for context-aware browsing of the world," in *Proc. IEEE Conf. Pervasive Comput. Commun.*, New York, 2007, pp. 229–238.

[13] C. Chong and S. Kumar, "Sensor networks: Evolution, opportunities, challenges," *Proc. IEEE*, vol. 91, no. 8, pp. 1247–1256, Aug. 2003.

[14] D. Clark, C. Partridge, C. Ramming, and J. Wroclawski, "A knowledge plane for the Internet," in *Proc. ACM SIGCOMM Conf.*, Karlsruhe, Germany, 2003, pp. 3–10.

[15] P. Costa, G. Coulson, C. Mascolo, L. Mottola, and S. Zachariadis, "Reconfigurable component-based middleware for networked embedded systems," *Int. J. Wireless Inf. Netw.*, vol. 14, no. 2, pp. 149–162, Jun. 2007.

[16] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, and A. Warfield, "Plutarch: An argument for network pluralism," in *Proc. ACM SIGCOMM Workshops*, Karlsruhe, Germany, 2003, pp. 258–266.

[17] A. K. Dey and G. D. Abowd, "Towards a better understanding of context and context-awareness," in *Proc. ACM Conf. Human Factors Comput. Syst.—What, Who, Where, When and How of Context-Awareness*, Hague, The Netherlands, 2000.

[18] S. Dobson, S. Denazis, A. Fernández, D. Gaıuml;ti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli, "A survey of autonomic communication," *ACM Trans. Auton. Adapt. Syst.*, vol. 1, no. 2, pp. 223–259, Dec. 2006.

[19] D. Estrin, D. Culler, K. Pister, and G. Sukjatme, "Connecting the physical world with pervasive networks," *Pervasive Comput.*, vol. 1, no. 1, pp. 59–69, Jan. 2002.

[20] P. Eugster, P. Felber, R. Guerraoui, and A. Kermarrec, "The many faces of publish/subscribe," *ACM Comput. Surv.*, vol. 35, no. 2, pp. 114–131, Jun. 2003.

[21] U. Fayyad, G. Piatetsky-Shapiro, and P. Smith, "Advantages in knowledge discovery and data mining," in *Data Mining to Knowledge Discovery: An Overview*. Cambridge, U.K.: MIT Press, 1996.

[22] D. Ganesan, D. Estrin, and J. Heidemann, "DIMENSIONS: Why do we need a new data handling architecture for sensor networks?" in *Proc. Inf. Process. Sens. Netw.*, Berkeley, CA, 2004

[23] J. Gips and A. Pentland, "Mapping human networks," in *Proc. Conf. Pervasive Comput. Commun.*, Pisa, Italy, 2006, pp. 159–168.

[24] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," *Computer*, vol. 34, no. 8, pp. 57–66, Aug. 2001.

[25] E. Höfig, "Autonomic reliable multicast: Application-level group communication using self-organization principles," in *Proc. Int. Conf. Bio-Inspired Models Netw. Inf. Comput. Syst.*, Budapest, Hungary, 2007.

[26] J. Hong, "The context fabric: An infrastructure for context-aware computing," in *Proc. Conf. Comput. Human Interaction*, Minneapolis, MN, 2002, pp. 554–555.

[27] D. Hong, T. Höllerer, M. Haller, H. Takemura, A. D. Cheok, G. J. Kim, M. Billinghurst, W. Woo, E. Hornecker, R. J. K. Jacob, C. Hummels, B. Ullmer, A. Schmidt, E. van den Hoven, and A. Mazalek, "Advances in tangible interactions and ubiquitous virtual reality," *Pervasive Comput.*, vol. 6, no. 2, pp. 90–96, Apr.–Jun. 2008.

[28] C. Julien and G. Roman, "EgoSpaces: Facilitating rapid development of context-aware mobile applications," *IEEE Trans. Softw. Eng.*, vol. 32, no. 5, pp. 281–298, May 2006.

[29] Y. Kalfoglou and M. Schorlemmer, "Ontology mapping: The state of the art," *Knowl. Eng. Rev.*, vol. 18, no. 1, pp. 1–31, Jan. 2003.

[30] A. Kulakov and D. Davcev, "Data mining in wireless sensor networks based on artificial neural-networks algorithms," in *Proc. Workshop Data Mining Sens. Netw.*, Newport Beach, CA, 2005.

[31] L. Lilien and B. Bhargava, "A scheme for privacy-preserving data dissemination," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 36, no. 3, pp. 503–506, May 2006.

[32] A. Masoud, "Decentralized self-organizing potential field-based control for individually motivated mobile agents in a cluttered environment: A vector-harmonic potential field approach," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 37, no. 3, pp. 372–390, May 2007.

[33] S. M. McConnell and D. B. Skillicorn, "A distributed approach for prediction in sensor networks," in *Proc. Workshop Data Mining Sens. Netw.*, Newport Beach, CA, 2005.

[34] R. Menezes, A. Charles, and R. Tolksdorf, "On the implementation of SwarmLinda," in *Proc. ACM Southeastern Conf.*, Huntsville, AL, 2006, pp. 297–298.

[35] G. Mottola and G. P. Picco, "Logical neighborhoods: A programming abstraction for wireless sensor networks," in *Proc. Conf. Distrib. Comput. Sens. Syst.*, San Francisco, CA, 2006.

[36] R. Newton and M. Welsh, "Region streams: Functional macroprogramming for sensor networks," in *Proc. Workshop Data Manage. Sens. Netw.*, Toronto, ON, Canada, 2004, pp. 78–87.

[37] A. Padovitz, S. W. Loke, and A. Zaslavsky, "Multiple-agent perspectives in reasoning about situations for context-aware pervasive computing systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 4, pp. 729–742, Jul. 2008.

[38] D. Patterson, L. Liao, K. Gajos, M. Collier, N. Livic, K. Olson, S. Wang, D. Fox, and H. Kautz, "Opportunity knocks: A system to provide cognitive assistance with transportation services," in *Proc. Conf. Ubiquitous Comput.*, Nottingham, U.K., 2004, pp. 433–450.

[39] E. Paulos and E. Goodman, "The familiar stranger: Anxiety, comfort, and play in public places," in *Proc. Conf. Human Factors Comput. Syst.*, Vienna, Austria, 2004, pp. 223–230.

[40] A. Pentland, T. Choudhury, N. Eagle, and P. Singh, "Human dynamics: Computation for organizations," *Pattern Recognit. Lett.*, vol. 26, no. 4, pp. 503–511, Mar. 2005.

[41] M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, D. Fox, and H. Kautz, "Inferring activities from interactions with objects," *Pervasive Comput.*, vol. 3, no. 4, pp. 50–57, Oct. 2004.

[42] A. Quteishat, C. Lim, and K. Tan, "A modified fuzzy min–max neural network with a genetic-algorithm-based rule extractor for pattern classification," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 3, pp. 641–650, May 2010.

[43] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. Van Laerhoven, and W. Van de Velde, "Advanced interaction in context," in *Proc. Symp. Handheld Ubiquitous Comput.*, Karlsruhe, Germany, 1999.

[44] J. Serrat, J. Serrano, J. Justo, R. Marıacute;n, A. Galis, K. Yang, D. Raz, and E. D. Sykas, "An approach to context aware services," in *Proc. NOMS*, Seoul, South Korea, 2004.

[45] L. Tao, W. Peng, C. Peng, M. Sheng, and H. Wang, "An integrated data-driven framework for computing system management," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 1, pp. 90–99, Jan. 2010.

[46] M. Ulieru and S. Grobbelaar, "Engineering industrial ecosystems in a networked world," in *Proc. 5th IEEE Int. Conf. Ind. Informat.*, Vienna, Austria, 2007, pp. 1–7.

[47] R. Want, "An introduction to RFID technology," *Pervasive Comput.*, vol. 5, no. 1, pp. 25–33, Jan.–Mar. 2006.

Nicola Bicocchi received the Laurea degree in computer engineering from the Università di Modena e Reggio Emilia, Reggio Emilia, Italy, in 2004, where he is currently working toward the Ph.D. degree in computer science.

His current research interests include computer networks, pervasive computing technologies, data mining for behavioral prediction, and activity-based computing.

Matthias Baumgarten received the Ph.D. degree in 2005.

He is currently working as a Research Fellow with the Faculty of Informatics, University of Ulster, Belfast, U.K. He has (co)authored about 25 publications. His main research interests are in the field of data mining, particularly web mining, pattern discovery, and personalization techniques.

Nermin Brgulja received the B.S. and M.S. degrees in electrical engineering from the University of Kassel, Kassel, Germany, in 2003 and 2005, respectively, where he is currently working toward the Ph.D. degree in the Department of Computer Science since April 2005.

His current research interests include context awareness, pervasive computing technologies, and autonomic communications.

Rico Kusber received the degree in computer science from the Chemnitz University of Technology, Chemnitz, Germany, in January 2006.

He has been a Researcher in computer science with the University of Kassel, Kassel, Germany, where he is also the Chair for Communication Technology, since February 2006. During his study, he specialized in artificial intelligence. His current research interests comprise artificial immune systems, autonomic computing and communication, and autonomic decision-making processes.

Marco Mamei received the Laurea degree in computer engineering and the Ph.D. degree in computer science from the Università di Modena e Reggio Emilia, Reggio Emilia, Italy, in 2001 and 2004, respectively.

Since September 2004, he has been a Research Associate in computer science with the Università di Modena e Reggio Emilia. His current research interests include pervasive computing technologies, data mining and data analysis for behavioral prediction, amorphous computing, and autonomic communications.

Maurice Mulvenna (AM'02–M'03–SM'08) received the B.Sc. degree (Honors) in science, the M.Phil. degree in information systems, and the Ph.D. degree in computing science from the University of Ulster, Belfast, U.K., in 1988, 1997, and 2007, respectively.

He is currently a Senior Lecturer in computing science with the School of Computing and Mathematics, University of Ulster. He is an Editorial Review Board Member of the *International Journal of Ad Hoc and Ubiquitous Computing*. He is a grant holder of the IST-BONUS, CASCADAS, and COGKNOW European projects and the Nestling Technologies project. He has published more than 120 papers in the area of computer science, addressing both theory and practice.

Dr. Mulvenna is a Senior Member of the Association for Computing Machinery and a chartered member of the British Computer Society. He was the recipient of the 2001 European IST Grand Prize.

Franco Zambonelli obtained the Laurea degree in electronic engineering and the Ph.D. degree in computer science from the University of Bologna, Bologna, Italy, in 1992 and 1997, respectively.

He has been a Professor in computer science with the Università di Modena e Reggio Emilia, Reggio Emilia, Italy, since 2001. He is currently the Scientific Manager of the EU FP6 IP Project "CASCADAS," where he also plays the role of leader for the "Knowledge Networks" WP. He has published one monograph and over 50 papers in international peer-reviewed journals and has coedited seven books. His current research interests include distributed and pervasive computing, autonomic computing and communication, software engineering for large-scale agents, and service systems.

Dr. Zambonelli has received several best paper awards.