Context-Awareness Supporting the Intelligence of Pervasive Systems

Maria A. Strimpakou, Ioanna G. Roussaki, Miltiades E. Anagnostou
School of Electrical and Computer Engineering
National Technical University of Athens
9 Heroon Polytechneiou Street, 157 73, Athens
Greece
{mstrim, nanario, miltos}@telecom.ntua.gr

Abstract – **In order for traditional computing systems to become pervasive, the service-enabling infrastructure must evolve accordingly. Services and applications need to be invisible, so that their interaction with the user is minimised. They need to be flexible, in order to respond fast to highly dynamic computing environments, and demonstrate localized scalability so as to efficiently reduce the interactions between distant entities. Finally, they must be personalized and adaptable, so that they maximise the utility of the user and become available anywhere anytime. Such a pervasive computing system that strives to be minimally intrusive and exhibits inherent proactiveness and dynamic adaptability to the current conditions and user preferences & environment, has to be context-aware. Context-awareness has the potential to greatly alleviate the human attention bottleneck, to increase the service flexibility and to support intelligent personalization and adaptability features. To enable the establishment of context-aware functionality, an infrastructure is required for the collection, management, inference and dissemination of context information to applications and users. This paper focuses on the formulation of an architectural framework that will be used to support pervasive computing integrated with the global telecommunication networks and based on an ambient context management system. This paper has sprang out from the work carried out in the Daidalos IST Integrated Project for a pervasive system, which is integrated with a heterogeneous all-IP network, and based on a robust context-aware environment, aims to provide intelligent pervasive services to mobile and non-mobile users.**

I. INTRODUCTION

Service provisioning systems are moving towards pervasive environments, where devices, networks and applications will all be expected to seamlessly integrate and cooperate in support of user requirements, desires and objectives. In a pervasive computing world [1], systems will be able to anticipate user needs, negotiating for services, acting on the user's behalf, and delivering services anywhere and anytime, all based on an environment saturated with computing and communication capabilities, yet having those devices integrated into the environment such that they disappear [2].

The abundance of commercial sensing technologies and the prevalence of powerful networked devices are bringing the pervasiveness vision closer. Nevertheless, before this vision can be realized, some issues need to be addressed, an indispensable of which is context-awareness (CA) [3]. CA provides pervasive environments with the ability to adapt the services or content they provide, by implicitly sensing and automatically deriving the users' needs from the context that surrounds them. CA distinguishes context-aware from traditional applications, in the sense that it makes them more attentive, responsive, predictive and aware of the user desires and environment. Context is any information that can be used to characterize the situation of an entity [4]. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves. User context may include a wide variety of data collected via sensors such as the current temporal and spatial location, the weather or even the user's biological state, and manually entered information such as user preferences and identity details. The current context-aware applications are usually built in an ad-hoc manner and lack the generality and standards to be useful in pervasive environments [5]. Most attempts to capture and use context within pervasive computing environments have focused on the physical elements of the environment, the user, or the user terminal. Reviews of such approaches of CA pervasive systems are provided in [6], [7].

While many authors acknowledge the importance of capturing the cognitive elements of user context, little work has been done to develop pervasive service provisioning platforms to support this. The research presented in this paper aims at this direction. It has sprang out from the work carried out in DAIDALOS IST Integrated project [8], focusing on its objective to integrate pervasive computing functionality with global telecommunication networks, so that users may be presented with non-intrusive services and the same personalized features and user interface, irrespectively of access locations, terminals and network infrastructure.

The rest of this paper is structured as follows. In *Section II*, a pervasive scenario example is briefly presented, in order to identify the fields our work aims at. In *Section III* the general requirements for pervasive systems are provided, while in *Section IV* an overall pervasive computing architecture supporting the previous requirements is proposed. In *Section V*, the designed context subsystem is presented, along with a context classification and model adequate for the needs of pervasive environments. Emphasis is given to the offered Context Inference Engine, aiming to enhance the intelligence of the system, inferring context and patterns not directly observable or retrievable. Finally, in *Section VI*, conclusions are drawn and future plans are exposed.

II. A PERVASIVE SCENARIO EXAMPLE

Pervasive computing is about moving away from the model of the single dominant general purpose machine (i.e., PC, laptop, PDA, ...), towards a population of machines hidden in the environment in a natural and intuitive manner. In such an emerging field, pervasive applications aim to exploit context information in order to provide simpler or less "annoying" human-machine interfaces, as well as intelligent & cost-effective consumer & business solutions. In order to design and develop a suitably flexible and feasible solution that will capture

users' needs and reveal the system profound requirements, a scenario driven approach has been adopted. Subsequently, such a pervasive scenario is presented.

It is an early morning in January 2010 and Johnny Insomnic cannot sleep. He picks up his watch, it is only five. He gets up and goes straight to the bathroom where he throws some water on his face. As his bathroom can sense activity and considering that he prefers to drink tea in the morning, a warm cup of tea is automatically prepared in the kitchen.

"He enters the kitchen where various flat surfaces host screens, including the table and the refrigerator. Shortly his cup is half empty on the table and he drags it back and forth in a playful mood. His eyes watch the news page restructuring itself around the cup to allow space for it. The embedded screen can re-arrange the content around the objects, when this is possible or necessary. Soon, he starts talking to the table, in order to edit a text, as a time always comes when a project 'deliverable' must be submitted.

After a while he realises how hungry he is and he spreads the very last portion of his favourite orange marmalade on a piece of bread, knowing that more marmalade is arriving soon. The automatic refrigerator is placed near the external wall of his apartment, and the employee of the food company can always 'refuel' it without entering the house. His expectation is confirmed later, as occasional hollow noises come from the fridge. He waits until nothing more is heard, then opens the door, picks up a jar full of orange marmalade and spreads a generous portion over a piece of bread. He mechanically accepts the bill for the just arrived provisions and moves his empty cup on top of yet another pop-up window to suppress a message saying '*Hello Johnny, you are 5.2 kilograms overweight*'. '*Where are the good old days when tables used to talk less*' he thinks.

A voice message is heard and at the same time a new window pops up on the table: '*Anna is expecting you in Hairstyle at half past nine, don't let her wait. Our address is ...*'. Anna is the hairdresser and he knows the address quite well."

In the above scenario, it is illustrated how users can make the best use of their services continuously, while everything is taken care of non-obtrusively, in the background. This is the most important aspect of pervasive computing, and is based on highly intelligent devices and mechanisms that infer user's intentions and needs and silently act accordingly.

III. REQUIREMENTS ON PERVASIVE SYSTEMS

In this section, the key requirements, which enable seamless pervasive service provisioning in heterogeneous, dynamically varying computing and communication environments, are briefly presented. Following an entity-based approach, five categories are distinguished, albeit alternative but compliant divisions can be found.

Context Related Requirements. As a variety of context-aware environments begin to flourish, efficient context discovery mechanisms should be employed for the extraction of the contextual information from the surroundings. Apart from the context collection, equally critical is the field of context lifecycle management, which involves the representation, interpretation, storage, dissemination, update and deletion of dynamic and static context information in real-time. Additionally, it is essential to define the appropriate inference mechanisms for deriving high-level context information based on primitive context data. Supplementary requirements include the reliability of context information, in the sense that it should somehow be represented, as well as the need for the platform to support registration of context-aware events, which will send notification on their occurrence to all interested users, applications or other entities.

User Related Requirements. Users should enjoy adaptation of their services, content and interfaces in a proficient and seamless manner, with regards to their current context values, taking into account cost/benefit and other factors (e.g., risk). Additionally, a personal data management interface is required that will support user-friendly methods for configuring the user preferences in a fast, robust and easy manner. Another significant trend is enabling the user to maintain multiple profiles, each representing different role (i.e., home-profile,office-profile, entertainment-profile etc.). Likewise, quite important is the existence of personal assistants and decision support modules that will monitor users' behaviour and exhibit considerable intelligence for their benefit. Finally, support for coherent user management is essential, e.g., the user should log-in the service platform only once.

Services Related Requirements. A pervasive service platform should provide sound mechanisms for service management, including service development and deployment, context-dependent service discovery, selection & composition, access to user-related context, service adaptability, service & session mobility, charging & accounting requirements, quality of service, etc.

Network Related Requirements. The network environment should provide a consistent, efficiently accessible, reliable and constantly available infrastructure for the provision of pervasive services, independently of the access network type (fixed or wireless), seamlessly integrating all network heterogeneities.

Privacy & Security Related Requirements. There is an inherent contradiction in trying to build pervasive albeit secure services, because pervasiveness is about spreading information around, while security and privacy is about preventing information dispersion. Thus, a robust, consistent and multilateral security & privacy framework should be carefully designed and developed.

IV. A PERVASIVE COMPUTING ARCHITECTURE

A pervasive computing architecture, which supports the range of services necessary for the realisation of the described scenario, has already been specified. Its main objective is to provide a robust intelligent service platform for pervasive systems, which goes beyond the current research. By integrating networking and service provision innovations with sophisticated context-aware service provision, this architecture aims to provide a platform for the future. An abstraction of the designed pervasive system overall architecture is depicted in Fig1.

The basic mechanisms, tools and subsystems supported by the pervasive services platform are briefly presented in the following paragraphs.

Context Subsystem: The platform will provide all the basic mechanisms needed for retrieving, processing, managing and disseminating all context data needed to enhance the pervasiveness and intelligence of the system. These

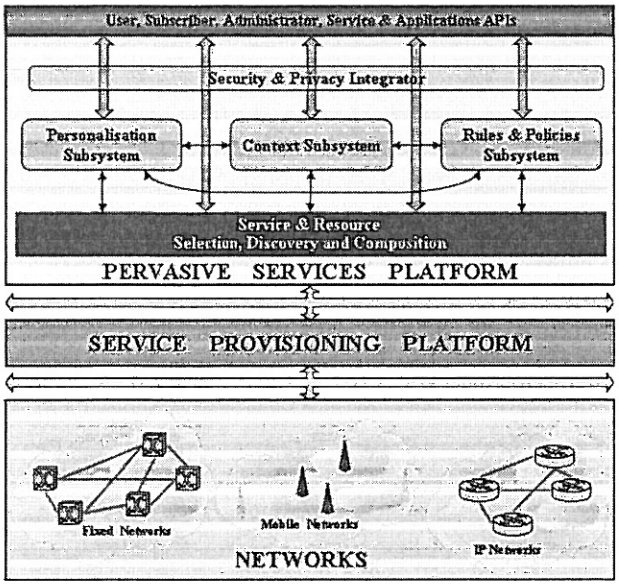

Fig. 1. A Pervasive Services enabling Architecture

mechanisms include federated context distribution and inference. The context concept and subsystem will be examined in detail in the following section.

Personalization Subsystem: Focusing on user needs and user-centric provision of invisible services, this subsystem will develop new forms of enhanced personalization functionality based on context information. It will provide a flexible and transparent mechanism that will adapt the services to the user preferences, agenda, environment, terminal, network, current conditions and all other user related context data.

Rules & Policies Subsystem: This subsystem will support: (1) user-centric configuration and control of context-aware services and their interactions, (2) enhanced user control of security / privacy and (3) detection and dynamic resolution of conflicts between rules & policies. The rules may be defined by the user, included in the content transmitted by the content provider, or automatically produced by the system based on intelligent mechanisms and learning algorithms.

Security & Privacy: Using the security mechanisms provided by the underlying IPv6-based network architecture, pervasive services will provide to the user seamless privacy protection and security, for any types of services, provider identities and terminal types.

Additionally Service and Resource discovery, selection and composition functionality will be supported, while the necessary entities for the communication with the underlying traditional service provisioning platform (SPP) and the interaction with peer pervasive services platforms will also be provided. The SPP will support overall network management, QoS control, security, A4C and other mechanisms necessary for the provision of conventional services. The described architecture addresses different types of networks: IP Networks (wired or wireless), Mobile Networks (GSM, GPRS, EDGE, UMTS,...) and Fixed Networks (PSTN, ISDN, xDSL,...).

V. THE CONTEXT SUBSYSTEM

There are many different types of context information that are vital for the realization of a fully pervasive system, since practically any piece of information could be considered as context. This section is structured as follows. In *subsection A*, a classification of context information is presented. In *subsection B*, the designed context subsystem architecture is described, as it is incorporated in the pervasive architectural approach presented in the previous section. Finally in *subsection C*, the functionality of the context inference engine is briefly described.

A. Context Classification and Model

In order to use context effectively and work towards intelligent context-aware systems, a generic context model should be developed, applying to all service categories and types and being abstract enough and easily adjustable to future context concepts. Thus, there will be a uniform way to design context sensitive applications, focusing on the determination of the appropriate context type to be used, and the selection of the mechanisms required to support context-aware computing.

Subsequently, a classification of the context types is presented that addresses the overall needs for information acquisition and processing in the provision of pervasive services, spreading from simple static applications to the more sophisticated applications that use 3G/4G wireless networks or sensor networks. A first level classification can be made according to the nature of the context information. Thus, the following context categories can be distinguished: (i) user "fingerprints", (ii) user preferences, (iii) user agenda, (iv) location, (v) time, (vi) user current state, (vii) terminal, (viii) network and (ix) any other data. Further analysis to the above context classes can be found in [9]. In Fig. 2, the structure of the context model is depicted.

Another kind of context distinction, which is quite important for service designers and developers, concerns the way that the information is provided to the system and also the duration and updating frequency of context data. Thus, there is a distinction between context that can only be provided to the system by the user in a manual way – filling forms or via voice recognition techniques– and context that may also be acquired automatically –via sensors, detectors, cameras, or microprocessor signal interpreters. Going further to this direction, we derive a classification that distinguishes between context types that are permanent –unlikely to be changed– or temporary – expected to be changed or updated often–, static –invariant within an access session– or dynamic –change constantly. In Table I the aforementioned context types are classified according to the above criteria, while an example of a service that is likely to use the particular type of context is also provided in each case.

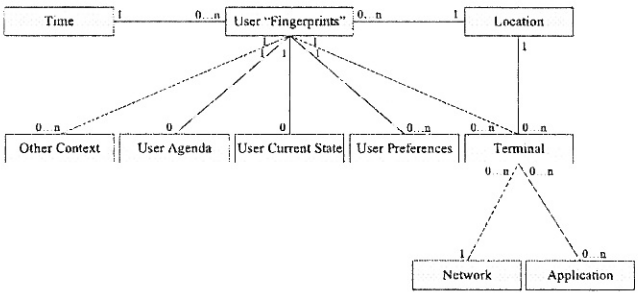

Fig. 2. Context Information Model

TABLE I

CONTEXT TYPES CLASSIFICATION AND SERVICE EXAMPLES

	Context type	Manually entered	Automatically acquired	Permanent	Temporary	Static	Dynamic	Service Example
User "Fingerprints"	Contact information / User identity	•		•		•		Directory service
	Professional context	•		•		•		Emergency service
	Social context	•		•		•		Chat service
	Health / medical context	•		•		•		Medicine Discoverer
	Physical features	•		•		•		Dating service
	Psychological features	•		•		•		Partner selector
	Background / skills / capabilities	•		•		•		Employment Agency
	Personal preferences / interests	•	•	•		•		Travel Agency
	User History context	•	•		•		•	Tourist Guide
User Preferences	Interface preferences	•			•	•		Any service
	Service preferences	•			•	•		Any service
User Agenda	Agenda / Calendar information	•	•		•		•	Meeting Scheduler
Location	Location information	•	•		•		•	Transportation Route Discovery service
	"Civilisation" information		•		•		•	Theft Recovery service
	Physical environment information		•		•		•	Excursion Planner
	User surroundings information	•	•		•		•	Professional Presentation Consultant
Time	Date & time information		•		•		•	Dinner Planner
User current state	User biological state	•	•		•		•	Health Monitoring
	User emotional state	•	•		•		•	Entertainment Planner
	User activity / availability	•	•		•		•	Telephone Call Filtering
Application	Application specific information	•	•	•	•	•	•	Any service
	Server specific information	•	•	•	•	•	•	Any service
Terminal	Terminal parameters and variables	•	•		•	•	•	Any service
Network	Network parameters and variables	•	•		•	•	•	Any service

B. Context Subsystem Architecture

In order to support the intelligence of pervasive systems, an efficient subsystem that establishes the context-awareness related functionality needs to be incorporated. The Context Subsystem (*CS*) designed is responsible for the following: (i) acquisition of raw context data from network resources, (ii) inference of additional and not directly observable context information, (iii) control of access to the context data, and (iv) sharing and manipulation of context data. The set of the software entities that consist the CS are depicted in Fig. 3.

The CS components, briefly described in the following paragraphs, enable the creation and provision of context-aware intelligent services to mobile or non-mobile users.

The *Context Broker* (CoB) answers to context requests. A context consumer requests context from the CoB that handles the subsequent negotiation. This negotiation is based on the requester's quality requirements, the context owner's authorization settings (e.g., access rights for privacy) and the available context and its quality parameters. Based on this information, it decides on the most appropriate context source among multiple sources (i.e., context handlers, peer CoB, or context providers).

The *Context Handler* (CH) retrieves context from the sensors (raw data) and the Inference Engine (processed context data), processes this data and reacts to context-related events. Retrieval implies polling data or receiving pushed data and converting the data in the CH's uniform format. The CH provides context to the CoB on demand. It hosts sensor wrappers that convert proprietary sensor data and protocols supporting a unified context data model, while it also manages their life-cycle and is aware of the sensors' states. The context data managed by this CH include all the information classes formerly presented in the context model subsection.

The *Inference Engine* (IE) infers any additional and not directly observable context information necessary for some pervasive features. It enhances the intelligence of the system, produces useful information and extracts patterns that have high added value for both consumers and providers. As it is a critical part of the context subsystem, it will be described in detail in the following subsection.

Apart from the information entered by the user or the services themselves (i.e., user fingerprint, preferences, agenda, ...), the raw data sources that are used directly by the Context Handler are physical, software and network sensors. The *Physical Sensors* collect information related to the physical environment or condition of the entities inside (e.g., location, temperature, weather, user current biological/emotional state, ...). These sensors are usually a piece of hardware that is connected through some rather low-level protocol (e.g., RS-232). The *Software Sensors* are software components that reside inside other software components such as applications. They retrieve context data inside the application and forward it to the context handler. Example of such sensors is one monitoring the calendar application that senses the user's planned activity and location. The *Network Sensors* use network components (such as base stations) to retrieve context information. Examples of types of data collected are: the identification of the network and terminal used, network traffic data or the user location/presence in a network (e.g., GSM, Bluetooth, WLAN, ...). All sensors along with other context sources provide their collected context information to the Context Handler via the *Raw Context Sources Gateway*, which handles the security and authorization functionality related to accessing the raw context data and provides transparency to the underlying infrastructure.

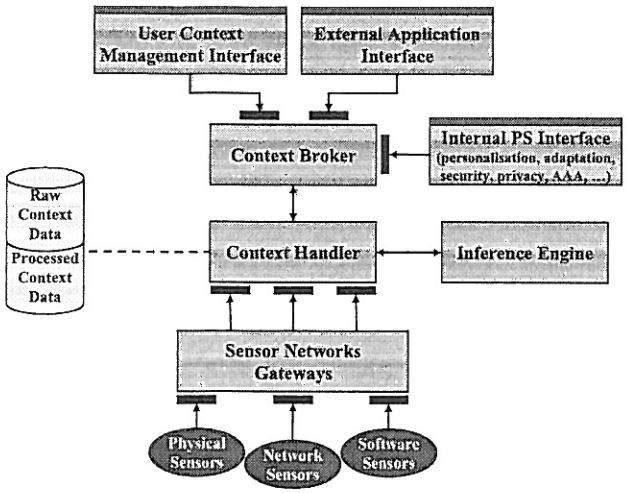

Fig. 3. Context Subsystem Architecture

Three main interfaces are offered to actors or other entities outside the CS by the Context Broker: the User Context Management Interface, the External Application Interface and the Internal PS Interface (Internal Pervasive System Interface). The *User Context Management Interface* is a user interface that allows the user to configure any data he/she has previously defined. This user related context includes: user profile & preferences, user identities, rights to other users, selected priorities, notification preferences, etc. The *External Application Interface* is an interface offered to third party service providers that supports access to the context data maintained by the platform. The *Internal PS Interface* is offered to the components of the rest of the subsystems in the pervasive system architecture. It supports access to the context data maintained by the platform. The entities that use this interface are: the Rules & Policies and the Personalisation subsystems, the service & resource discovery and composition entities and the service provisioning platform enabling components.

The designed context subsystem will interact and cooperate with peer systems in foreign domains via SLA & federation mechanisms. The context information will be available to authorized agents in a secure manner. Globally accessible, secure Context Brokers will be developed, so that context is available anywhere, anytime to a variety of authorised terminals, networks, services and applications.

C. Functionality of the Context Inference Engine

The spread of intelligence in everyday objects, which is due to the miniaturization and cost reduction of hardware, has boosted information gathering. Therefore, as the human-computer interface becomes more pervasive and intimate, system developers are called to deal with a rapidly increasing deluge of information, which is growing fast as user experience and situation are constantly monitored and integrated into the computer and communication systems design process.

In general, a pervasive system can be seen as an assistant that can exploit users' context and become responsible for making decisions in a proactive fashion anticipating users' needs and without disturbing them. The idea of exploiting a context inference engine is due to the necessity for the context provision layer to exhibit considerable intelligence, in order to produce simple conclusions and specify the values for various context-defining parameters from a wealth of incoming data. The objective here is to enable the system to judge whether a piece of context information is useful for an application, perform an information filtering and thus avoid creating a bottleneck in users' attention by minimising the user's redundant interaction with the system. In other words, context inferring can be used to reduce the amount of explicit input a person is required to provide to the system, by combining various information from diverse sources, in order to extract useful knowledge.

The first applications that exploited context have adopted a check-list approach, i.e., they have simply used location, weather information, presence of people in the neighbourhood of the interaction, etc, while restricting the interaction to the exchange of a small number of well-defined and simple pieces of information. At present, the research in pervasive systems moves towards a more sophisticated and general-purpose attitude: it anticipates building proactive systems that will not consume the valuable resource of human attention in a haphazard and inefficient fashion. Instead, they will be remarkably sensitive to the human context in which they operate and will establish advanced automated decision makers. The latter will exploit an inference engine in order to combine stand-alone context data and gain a full picture of the context-aware system so as to convey the appropriate information to the right place at the right time.

Our research work seeks to assess the feasibility of this kind of contextual inference and automatic prediction based on explicit or implicit input data. Specifically it seeks answers to at least the following questions:

- To what extent can the user intervention be reduced?
- How can a balance be found between overloading the user with information and making him/her able to customise services and decide for critical issues?
- Can inference engines and automatic decision making be effectively used in general purpose environments?
- What is an appropriate model of the environment, which could support automatic decision making?
- What are the dangers created by the process of inferring?
- What are the security and privacy problems in automatic decision making?

The value of inference mechanisms can be derived by various principles. Instead of having different applications perform the same calculations based on the same raw information, a common pre-processing step from an inference engine will result in making the calculation just once. Later the outcome will be distributed to all interested processes. A typical example is weather information processing. Most applications are interested in general statements of the type "it will rain" or "snow is coming", but they are not interested in any others specialised parameters. Besides the aforementioned economies of scale, of major importance is the complexity of the required processing in order to deduce usable information given the oversupply of input data, that is almost intangible to handle in a realistic and acceptable time frame for every single application. Moreover, the fact that quite often only the truth value of simple propositions is needed, e.g., not the exact pressure value in a car tire, but whether there is flat tire problem or not, highlights the need for an inference engine that will process raw data and produce critical

assertions that will be fed into applications for further elaboration.

In this framework, inferring can sometimes be reduced to the provision of simple results to a large number of context information, gathered by various sensors and interfaces. For example, a number of services related with outdoor activities could be interested in whether it is raining or will rain within the next few hours, while the exact evolution of the atmospheric pressure value would be useless. However, there are numerous examples of really demanding processing instances, which involve highly intelligent inference engines.

Despite the theoretical difficulties in building a complete and predictable intelligent system of context-aware services, available prediction and testing methodologies should be fully exploited and their limits should be explored. Fuzzy logic [10], which has been used in expert systems, is a possible choice. Another option would be machine learning techniques [11] (e.g., Bayesian networks, neural networks or reinforcement learning) that can be used to create predictive statistical models that utilise the context data to predict the user's behaviour and automatically decide about users intentions. Finally, rules and policies could be imposed in order to provide the essential guidelines for drawing the appropriate conclusions (i.e., If I am in my office working on my PC, then turn my availability on for business issues). Nevertheless, we should not expect in the near future to see anything close to an artificial angel, who will silently solve a great deal of our practical problems. In other words, any attempt to construct a complex inference engine as a panacea from every possible situation is likely to eventually create a nightmare. Therefore, it seems that the mandate towards theoretical contributions in the area of the methodology of inference engine design is to provide a systematical approach for general-purpose context information. In any case, the presence of an efficient context inference engine embedded in a context subsystem, which is incorporated into an integrated pervasive system, will result in an emerging and highly responsive environment that will create new forms of interactive experience between users and machines.

VI. CONCLUSIONS

In this paper, the design of a powerful integrated platform for pervasive computing is presented, focusing on the context-aware functionality and the relative enabling mechanisms and models. This is a context-aware platform, aiming to be minimally intrusive and demonstrating inherent proactiveness and dynamic adaptability to the current conditions, user preferences and environment. Thus, it will greatly alleviate the human attention bottleneck, increase the service flexibility and support intelligent personalization and adaptability features. It will enable the rapid development of sophisticated context-aware services, able to support users in their tasks in an environment saturated with sensors, and computing & communication devices.

We are currently working towards the integration of a pervasive system prototype, focusing on the support of intelligent context-aware personalised services. It will be evaluated over a blend of information processing and communication infrastructures –at terminal and network level– and will encompass multi-role domains. Its validation is expected to further contribute to the integration of pervasive systems, while this work will hopefully make a vital step towards the introduction of pervasive service provision in the wide market. Nevertheless, considerable effort should be spent in order to reconcile all the conflicts between automatic and user-controlled environments, balancing cost versus benefit from developing system's ability to offer proactivity, while carefully ensuring the users' privacy.

VII. ACKNOWLEDGMENT

This work has been partially supported by the Integrated Project DAIDALOS ("Designing Advanced network Interfaces for the Delivery and Administration of Location independent, Optimised personal Services"), which is financed by the European Commission under the Sixth Framework Programme. However, this paper expresses the authors' personal views, which are not necessarily those of the DAIDALOS consortium.

VIII. REFERENCES

[1] M. Satyanarayanan, "Pervasive Computing: Vision and Challenges," *IEEE Personal Comm.*, vol. 8, no. 4, Aug. 2001, pp. 10-17.

[2] P. Gupta and D. Moitra, "Evolving a pervasive IT infrastructure: a technology integration approach," *Personal and Ubiquitous Computing Journal*, vol. 8, no. 1, 2004, pp. 31-41.

[3] S. Xynogalas, M. Chantzara, I. Sygkouna, S. Vrontis, I. Roussaki, and M. Anagnostou, "Context Management for the Provision of Adaptive Services to Roaming Users," *IEEE Wireless Comm.*, vol. 11, no. 2, Apr. 2004.

[4] A. Dey, "Providing Architectural Support for Building Context-Aware Applications," Ph.D. thesis, College of Comp., GA Inst. of Tech., Dec. 2000.

[5] A. Dogac, G.B. Laleci, and Y. Kabak, "A Context Framework for Ambient Intelligence," Technical Report, No: EEEAG 102E035, Middle East Technical University, 2003.

[6] P. Prekop and M. Burnett, "Activities, context and ubiquitous computing," *Comp. Comm. J.*, vol. 26, no. 11, July 2003, pp. 1168-1176.

[7] K. Barrett, R. Power, "State of the Art: Context Management," *M-Zones Research Programme*, Deliverable 1.1: State of the Art Review, May 2003.

[8] DAIDALOS IST Integrated Project, (URL: http://www.ist-daidalos.org/).

[9] S. Xynogalas, I. Roussaki, M. Chantzara, and M. Anagnostou, "Context Management in Virtual Home Environment Systems," *Journal of Circuits, Systems, and Computers*, (in press).

[10] L.A. Zadeh, "Knowledge representation in fuzzy logic," *IEEE Trans. on Knowledge and Data Engineering*, vol. 1, no. 1, 1989, pp.89-100.

[11] T. Mitchell, *Machine Learning*, Boston: McGraw-Hill, 1997.