# Moving forward on u-healthcare: A framework for patient-centric context management.

Ana Isabel Calvo-Alcalde[1], Ana M. Bernardos Barbolla[2], Josué Iglesias Álvarez[2], Juan José Andrés-Gutiérrez[3], Esteban Pérez-Castrejón[3], José R. Casar Corredera[2]

[1] Universidad de Valladolid,
Escuela Técnica Superior de Ingeniería Informática
Campus Miguel Delibes s/n, 47011, Valladolid, Spain
anaisabel.calvo@alumnos.uva.es
[2] Universidad Politécnica de Madrid,
ETSIT, Ciudad Universitaria s/n, 28040, Madrid, Spain
{abernardos, josue, jramon}@grpss.ssr.upm.es
[3] Telefónica I+D
Parque Tecnológico de Boecillo
47151 Boecillo, Valladolid, Spain
{jjangu, esteban}@tid.es

**Abstract.** Delivering remote healthcare services without deteriorating the 'patient experience' requires building highly usable and adaptive applications. Efficient context data collection and management make possible to infer extra knowledge on the user's situation, making easier the design of these advanced ubiquitous applications. This contribution, part of a work in progress which aims at building an operative AmI middleware, presents a generic architecture to provide u-healthcare services, to be delivered both in mobile and home environments. In particular, we address the design of the Context Management Component (CMC), the module that takes context data from the sensing layer and performs data fusion and reasoning to build an aggregated 'context image'. We especially explain the requirements on data modelling and the functional features that are imposed to the CMC. The resulting logical multilayered architecture -composed by acquisition and fusion, inference and reasoning levels- is detailed, and the technologies needed to develop the Context Management Component are finally specified.

**Keywords:** Semantic reasoning, context-aware architecture, data fusion, user experience, Ambient Intelligence.

## 1 Introduction.

Health care is a priority in Europe. The social challenge is to keep the costs of healthcare systems under control while maintaining a high quality service. ICTs are expected to contribute to addressing this challenge and to supporting a paradigm shift in health care delivery, by focussing on the autonomous citizen (i.e., proactive with respect to her/his own health, enabled to self-care, seeking services for prevention and disease management and aware of lifestyles) and independent living [1]. In particular,

the Ambient Assisted Living Joint Programme (AAL) [2] promotes strategies and technologies enabling elderly, chronic or disabled people to stay at their homes as long as possible, by increasing their autonomy and self-confidence.

On the other hand, current design trends for innovative products and services are focused on how to enhance usability and user experience. User experience may be understood as 'how well the user understands the product or service, how well he feels while using it, how well it serves his purposes, and how well it fits into the scenario of use'. Usability refers to how the whole interaction with the product/service is experienced by the user. There are multiple factors impacting on the user experience, but an important issue is how to adapt the services offer and their performance depending on the user's preferences and his real-time situation (or his 'context').

When applying these design drivers to healthcare, the objective is to achieve the highest patient satisfaction, through user-centric services which improve the patient experience in the very different environments where they are (or might be) delivered (e.g. hospitals, residences, primary care centres or homes) [1]. Then, the technology concept behind Ambient Intelligence (AmI) [3] becomes an enabler of AAL ubiquitous healthcare services, as AmI sketch a medium-term scenario where people will live surrounded by transparent embedded technology, opportunistically accessible through simple and natural interactions adapted to the users' preferences and context [4].

This work combines these three paradigms (user-centric design, AAL and AmI) to build patient-centric services on highly sensitive and interconnected environments. In particular, a functional architecture to provide context-aware healthcare services is described, together with its subsequent technological approach. The architecture is built on the assumption that context data are accessible through a number of sensors, and its objective is to define how to gather all this information synchronizedly, to extract sufficient information about how the user's context is, in order to provide him with suitable services. The highly adaptation of the resulting services will collaborate to get a positive 'patient-experience'.

The paper is structured as follows. Section two presents a brief review of the state-of-the art on context-aware frameworks and services for healthcare. Section 3 offers an overview of the global service-oriented architecture. Afterwards, Section 3 describes the specific logical architecture of the Context Management Component; the technologies that are being used to implement it are gathered in Section 4. Finally, Section 5 summarizes the most important issues addressed and explains some future lines of work.

## 2 State-of-the-art: AmI frameworks and u-healthcare services.

Several generic reference middleware architectures for Ambient Intelligence systems have already been defined. One of the main goals of this kind of frameworks is to set a common practice for building context-aware services, lessening the development process by decoupling sensor data acquisition and context information usage. In this respect, Context Toolkit [5] can be considered one of the first proposals

facing this issue; it allows encapsulating sensor particularities and also proposes an architecture for applications to subscribe in order to receive customized context updates.

Later developments have focused on processing the information from sensors. These frameworks differ in their architectural approaches, with different methods of context representation and reasoning engines. For example, CoBrA [6] is an agent based architecture which uses ontologies to model data and uses a rule based inference engine. CMF [7] and Gaia [8] adopt the blackboard and distributed middleware architectural designs respectively. Both of them handle uncertainty when representing and reasoning about context information.

Platforms that use service-oriented architecture concepts are emerging recently with the aim of improving the dynamism and scalability of this kind of systems. In this sense it is worth to point out the one defined in [9], which automatically integrates service discovery mechanisms and gateway protocols in order to maintain a service definition for each sensor in a smart home environments.

On the other hand, advanced healthcare applications have been under prototyping some years by now; they usually imply the target user to wear sensors, and their main objective is to anticipate or detect health risks. For example, Kang et al. [10] propose a wearable sensor system which measures the biological functions of a person to provide remote health monitoring and self health check. Korel and Kao [11] also monitor the vital signs and combine them with other context info such as environment temperature or person's condition, in order to detect alarming physical states and preventing health risks on time.

Medication prompting functionalities are also frequent in Ambient Intelligent developments. Hardware to facilitate the medication consumption in the house has been developed, for example, by Agarawala et al. [12] and by Fishkin and Wang [13]. The later have developed a prototype of a system consisting of a portable pad that combines Radio Frequency Identification (RFID) tags and a sensitive scale to detect when a bottle has been picked up. Lundell et al. [14] describe the CAMP system, a generator of context-aware medication reminders which tries to adapt its notifications to the patient activity.

Moreover, several prototypes encompass the functionalities mentioned above: Rentto et al. [15], in the Wireless Wellness Monitor project, have developed a prototype of a smart home that integrates the context information from health monitoring devices and the information from the home appliances. Becker et al. [16] describe the amiCa project which supports monitoring of daily liquid and food intakes, location tracking and fall detection. Haigh et al. [17] developed an agent-based monitoring and support system that issues reminders, alerts and notifications, generates summary reports of patient's behaviour and provides an overview of person's state and medication compliance.

Although some of these systems have been evaluated for a reduced number of users, the review of the applications in the domain of ambient home care system indicates that this application area is still in its early stage.

## 3 AmI architecture to provide healthcare services.

Current technology makes possible that a convalescent person gathers his biomedical data through portable sensors to be sent in real time to the hospital for analysis, before having a remote session with the doctor. On the other hand, wireless sensors scattered all over a house may accurately inform about the ambient conditions where a COPD patient is living. A person suffering dementia may benefit for a 'remainder' service which will help him to find personal objects and remember important dates.

These simple scenarios are just some realistic examples of how future healthcare services may be provided. To handle their put into operation, it is necessary to have an infrastructure capable of 1) handling context data acquisition and reasoning; 2) providing standardized interfaces to guarantee safe access to highly sensitive data (personal, biomedical, ambient and social information); 3) defining how integration of new data suppliers is made; 4) and securely interconnecting different service providers.

Figure 1 describes a general AmI architecture which considers the previous issues. The architecture aims at decoupling the context acquisition and reasoning levels from the applications development. It is an environment-independent approach, considering mobile and infrastructure sensors, interfaces and applications.

The lower layer ('Network Layer') acquires context information from different types of sensors (location, ambient, biometric, proximity ones) and supports the interaction with actuators (e.g. in a domotic system). The middle layer is the 'Context Management Component' (CMC), which hosts the algorithms needed to process the information from the 'Network Layer', in order to infer new knowledge and automate decision making. Finally, the upper level ('Value-added Services') contains the services using the information managed in the lower layers. The technological services are horizontal ones (e.g. videocall, monitoring, multimodal interaction, calendar, medication control, electronic prescription, etc.) which may be utilized by the functional services when needed.

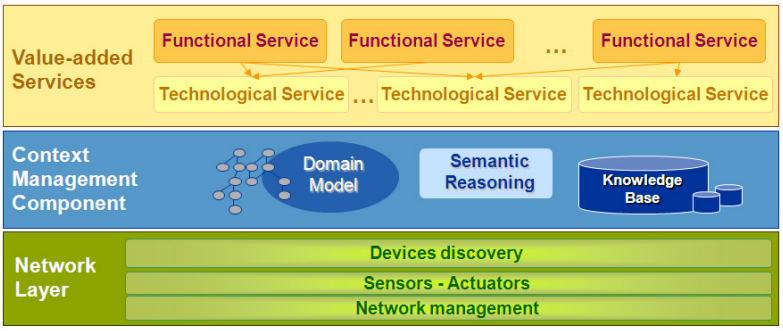



**Fig. 1.** AmI architecture general overview.

In the next Sections, the focus of attention will be the Context Management Component (CMC), which receives raw data from the hybrid sensor network, infers complex knowledge about the user's context –building a 'context image'- and translates it to a common and understandable format for the components of the system. With respect to technological services, this module adopts a dual working

mode: it can act as an information consumer, requesting data from horizontal services (e.g. from a positioning engine), or may work as an information provider, delivering data to other services (e.g. to an interaction module taking decisions on which interface to use to provide a given service). On the other hand, it is also capable of cooperating with the Network Layer to manage devices, in order to complete or increase the quality of the inferred information.

## 4 The Context Management Component: functional analysis and architecture.

### 4.1 General design principles for the Context Management Component.

Next there are the main requirements to the Context Management Component:

1) **A scalable and versatile data model**. Data from the sensing infrastructure, as well as high-level context information, needs to follow a data model which guarantees, on one hand, the integration of new and heterogeneous sensors, and on the other hand, the expressiveness needed to support reasoning processes. Apart from that, the data model has to be suitable for integration with off-the-shelf reasoning engines.

2) **A transparent solution for context data acquisition.** The Context Manager Component should provide access to the information coming from a wide variety of sensors, regardless their particular characteristics. Similarly, the CMC should define how to integrate information coming from different external context providers, defining the semantic interfaces needed, and providing translation/conversion functions.

3) **An inference and reasoning engine**. The reasoning engine, together with the data model, should allow opportunistic data fusion at different levels of abstraction, in order to transform data from deployed sensors into high-level context information.

4) **A decision-making engine**. The CMC must include algorithms and techniques to support decision making on context data at different levels of abstraction. This process may provide feedback to manage sensor and communication infrastructures, or even determine whether it is necessary to interact with the user.

5) **A standard interface to access context information**. The CMC should provide mechanisms to access context information through a common standard interface. This access may be synchronous (request – response), continuous (for a configurable interval of time) or event-based (subscription), depending on the needs of the application information requests. Again, standard access at different abstraction levels (signal, feature or context image level) is a must.

6) **A standard configuration interface**. An interface should be provided in order the system administrator to be able to configure the CMC. This interface can be used, for example, for defining new rules of inference or to configure new sensors.

7) **Context information storage services**. The CMC should provide configurable storage services with the objective of implementing potential mechanisms of machine

learning, state prediction and analysis of data for multi-purpose user's history processing.

8) **Quality of Context monitoring** (henceforth QoC). The CMC should be able to set certain rules or estimators of the quality used to infer context information (both, at feature and at context image level). This information may be used to control the CMC and to provide suitable information to upper levels, so they could execute their actions.

9) **User policy management**. Users can impose restrictions on the acquisition, use and dissemination of their context information, which will be managed by the CMC to ensure compliance. It must also ensure the safe handling and storage of information.

### 4.2 Describing the logical architecture for the Context Management Component.

From the design principles above, Table 1 summarizes the main functional features of the Context Management Component, and Figure 2 shows the multilayered approach.

The functionality of 'Evaluation and notification service' will be implemented by the first two levels in Figure 2. At its lowest level, **evaluation services** will check context values transitions, generating events managed later by the **notification management service**, which is responsible for notifying changes to the upper levels. This dispatch will be supported by the **subscription service** that will determine what entity needs each information unit.

**Context inference services** are designed to serve to different applications, and are shaped by their particular information requirements. On the one hand, they act as aggregators of context parameters that can be hierarchically combined to form aggregations of context data, that is, different 'context images'. Different applications may require instantiating the same context image, so the objective is to optimize the acquisition and processing by applying intelligent requirements management. These services are also used to classify 'context images' through pattern recognition, fuzzy logic, case-based reasoning, etc. To support the process, there is a **prediction service** and a **knowledge-base query service**. Finally, context inference services will have a mechanism to report transitions between contexts to the 'reasoning and decision' level.

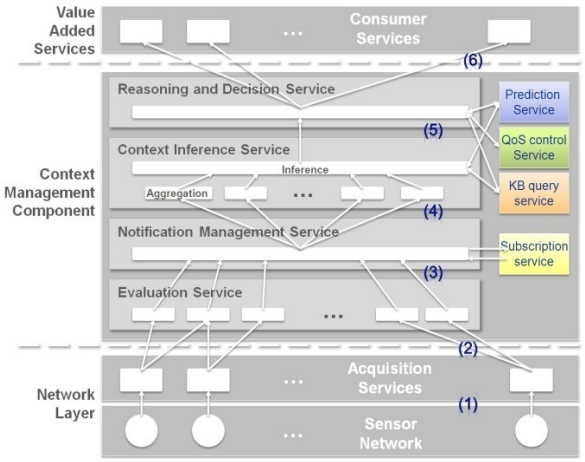
**Table 1.** Design requirements, ordered by functional services.

| | **Functional services** |
|---|---|
| **Evaluation & Notification Service** | Data association (synchronization, completeness and consistency, etc)<br>Monosensor feature extraction<br>Fusion for extracting composite features.<br>Error estimation<br>Storage |
| **Context Inference Service** | Application registry and definition of information requirements<br>Pattern matching<br>Prediction<br>Context image composition (situation and relation)<br>Context image classification<br>Storage |

| | |
|---|---|
| **Reasoning and Decision Service** | QoC estimation<br>Conflict detection<br>Privacy management<br>Sensor interaction and infrastructure control<br>User interaction<br>Resources assignment<br>Interface management<br>Application notification |

Functionalities related to 'reasoning and decision' will be implemented at the highest level of the CMC. The **prediction service** and the **knowledge-base query service** will be also used, and an estimation of the QoC will be calculated to support reasoning reliability and ensure that applications will offer the correct quality of service. The knowledge-base will be also updated, which may be later used by other control and learning services to verify the correct operation of the system. The reasoning and decision service will finally activate or send requested information to the final services, which will be in charge of executing decisions according to the reasoning service.

This approach, based on service layers, enables to modularize the processes of context instantiation and decision-making, adapting them consistently to the application that will finally consume the information. When a new application is integrated into the platform, its information needs have to be defined. The system will automatically configure the different service layers in order to deliver the information needed.




**Inputs/Outputs**

1. Raw data from sensors: signal strength, flight times, beacon ids, etc.
2. Estimators: position, temperature, humidity, preferences, etc.
3. Context-change events (position updates, user entering in zone Y, temperature higher than X, etc.)
4. Selective dispatch of event notifications, depending on the context images that each 'inference service' composes. Dispatch is supported by the 'subscription service', that will determine which 'inference service' needs each information unit.
5. Context images classified by patterns ('normality', 'emergency', 'intrusion', 'medication pending', etc.). They will be instantiated from aggregated context data ('X,Y position', 'temperature > 37º', 'caregiver presence', etc.).
6. Decisions. The 'reasoning service' assesses the quality of the context estimation and determines what action to take.

**Fig. 2.** Service composition of the Context Manager Module.

## 5 Technologies to implement the Context Management Component.

An operative middleware based on the Context Management Component previously defined is currently being implemented as part of a global architecture to handle u-healthcare service delivery. Following there is a brief description of the technologies that are being used to implement the CMC.

The general philosophy that drives the practical implementation of the global AmI architecture is service oriented (SOA); in particular, the OSGi platform has been chosen as general framework [18]. The CMC supports semantic reasoning by using software agents, in order to have a scalable system where new devices and agents may be easily integrated. Moreover, modules developed by third parties and distributed as OSGi bundles can also be included without refactoring the global system.

Then, the CMC is designed as a MultiAgent System (MAS) on the top of the OSGi layer. It is implemented in Java, using the Java Agent DEvelopment framework JADE [19], which is compliant to the FIPA standard [20]. Figure 3 shows the integration of the FIPA-compliant agent platform [21] with the OSGi framework.

On the other hand, the semantic reasoner which has been selected to provide intelligence to the CMC is Pellet [22]. Pellet is open-source and also implemented in Java, so it can be easily integrated with the JADE platform. It supplies reasoning functionalities by using an OWL ontology and declarative rules codified in Semantic Web Rule Language (SWRL [23]). It is also integrated with the Jena framework [24], which provides easy modes for programmatic ontology management through several APIs. Specifically, the integration has been carried out through the Inference API and the Ontology API has been used to fulfil the development process, aiming to access and manipulate the ontology. This framework also allows creating and managing its persistence using database models.

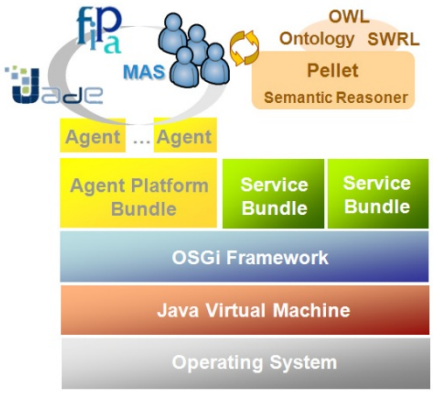



**Fig. 3.** CMC Technological Architecture.

As stated above, a data model is needed to support the reasoning process. In order to provide the u-healthcare services, the core data model has been designed to cover some basic entities, such as patients, caregivers, doctors and medical staff, apart from a description of the environments where the services are provided, in terms of

available devices (sensors, actuators, interfaces and communication infrastructure) or even physical structure (buildings, floors, rooms, etc.). Afterwards, each service defined in the global architecture is extending the data model depending on its particular needs. Data modelling has been done by using an ontology; following the 'reutilization rule', this ontology combines 'standard' previous works, in particular SOUPA, FOAF or CONON ontologies which has been suitable to model entities such as 'Time', 'Event', 'Location' or 'Policy'.

The ontology has been designed and developed with Protégé-3.3.1 [25] with the Web Ontology Language OWL-DL, which can be employed to explicitly represent the terms significance in vocabularies and the relationships between them.

Finally, BeanGenerator [26] has been chosen to act as an integration gateway to generates Java files representing the ontology that can be used with the JADE environment.

All the technologies used in the CMC are summarized in the following table.

**Table 2.** Technologies in the Context Management Component.

| | **Functionality** | **Technology** |
|---|---|---|
| **Software platform** | Platform | OSGi |
| | Programming Language | Java |
| **Data Model** | Ontology | OWL, OWL-DL |
| | Ontology Editor | Protégé |
| | Serialization | XML |
| | Ontology management | Jena |
| **Inference and reasoning** | Rules | SWRL |
| | Semantic Reasoner | Pellet |
| **Multi-Agent System** | Platform | JADE |

## 6 Conclusions and future work.

Delivering a good user experience has much to do with service design (workflows, interaction triggers, interfaces, etc.), but in case of context-aware services has also a close relationship with the quality of the context information, as it is the main input for the service's decision making processes. Context-aware systems often take for granted unreal assumptions on the context information quality. Typically, it is considered that the context they are dealing with is complete and valid without clear proofs, when actually, both sensed and interpreted context are imperfect.

Then, from raw sensor data acquisition to inference, it is necessary to be aware of what the quality of the information is. Each level of abstraction needs to manage this issue independently, and the whole system needs to be aware of how the errors or irregularities are transmitted when the information moves upwards the reasoning layers.

The design of the Context Manager Component architecture presented above includes this aspect, which has not been frequently considered in previous works. Nevertheless, to make the QoC control operative, it is needed to include specific data when modelling context (e.g. specific entities or attributes in the ontology representing sensors and systems), apart from designing decision algorithms to

efficiently use and update quality parameters. Handling quality of context allows the system to adequately interact with the sensing layer, both to improve context estimates and to avoid processing overloads when not necessary.

Nowadays, there exist several efforts trying to resolve context ambiguity. One of the open future lines of this work is related with the ability to handle imperfect context. Many researchers propose triggering direct interaction with the user as a reliable disambiguation mechanism. But this approach may not fit well within the Ambient Intelligence concept, where user interactions should be as minimum, simple and non intrusive as possible. Due to this, context ambiguity must be handled and regulated from the context estimation procedures themselves and from machine learning procedures having the user's behavioural pattern as an input. For example, if a user refuses some services repeatedly, this information may be used to create a more adapted service offer.

On the other hand, an important pending aspect is how to evaluate the performance of the proposed approach with respect to the patient experience. The particularities of the niche target users of these services (elderly, disabled and chronic patients) impose demanding performance requirements. We find that there is wide room for research on empirical user satisfaction modelling.

**Acknowledgments.** This work has been supported by the Spanish Ministry of Industry, Trade and Commerce through the CENIT AmIVital Programme and by the Government of Madrid under grant S-0505/TIC-0255.

## References

1. Aarts, E.H.L., Encarnação, J.L.: "True Visions: The Emergence of AmI". Springer. 2009.
2. Ambient Assisted Living Joint Programme. http://www.aal-europe.eu. 2008.
3. Weber, W., Rabaey, J.M., Aarts, E.: "Ambient Intelligence". 2005.
4. Information Society Technologies Advisory Group: http://cordis.europa.eu/ist/istag.htm
5. A. K. Dey, G. D. Abowd, D. Salber, "A Conceptual Framework and a Toolkit for Supporting the rapid Prototyping of Context-Aware Applications", Human-Computer I. J., Vol. 16 (2-4), 2001, pp. 97-166.
6. Chen, H. "An Intelligent Broker Architecture for Pervasive Context-Aware Systems", PhD Dis., University of Maryland, Baltimore County. 2004
7. Korpipää, P., Mantyjarvi, J.: "Managing context information in mobile devices. Pervasive Computing", IEEE, vol. 2, págs. 42-51. 2003
8. Ranganathan, A., Al-Muhtadi, J.: "Reasoning about uncertain contexts in pervasive computing environments". Pervasive Computing, IEEE, vol. 3, págs. 62-70. 2004
9. Helal, S., Mann, W., El-Zabadani, H., King, J., Kaddoura, Y., Jansen, E.: "The Gator Tech Smart House: A programmable pervasive space", IEEE Computer Society Press, Volume 38, Issue 3, pp. 50-60. 2005.
10. D-O. Kang, H-J. Lee, E-J. Ko, K. Kang, J. Lee, "A Wearable Context Aware System for Ubiquitous Healthcare", Proc. of the 28th IEEE, EMBS Annual Int. Conf., New York, USA, Aug-Sep 2008, pp.5192-5195.
11. B. T. Korel, S. G.M. Kao, "Addressing Context Awareness Techniques in Body Sensor Networks", 21st International Conference on Advanced Information Networking and Applications Workshops 2007, Volume 2, Niagara Falls, Canada, May 2007, pp.798 - 803.

12. A. Agarawala, S. Greenberg, G. Ho, “The Context-Aware Pill Bottle and Medication Monitor”, In Video Proceedings and Proceedings Supplement of the UBICOMP 2004, Nottingham, England, September 2004.
13. K. Fishkin, M. Wang, “A Flexible, Low-Overhead Ubiquitous System for Medication Monitoring”, Intel Research Seattle Technical Memo IRS-TR-03-011, October 2003.
14. J. Lundell, T.L. Hayes, S. Vurgun, U. Ozertem, J. Kimel, J. Kaye, F. Guilak, M. Pavel. “Continuous Activity Monitoring and Intelligent Contextual Prompting to Improve Medication Adherence”. Proc. 29th Annual Int. Conf. of the IEEE EMBS, Lyon, France, August 23-26, 2007.
15. K. Rentto, I. Korhonen, A. Vaatanen, L. Pekkarinen, T. Tuomisto, L. Cluitmans, R. Lappalainen, “Users’ Preferences for Ubiquitous Computing Applications at Home”, First European Symposium on Ambient Intelligence 2003, Veldhoven, The Netherlands, November 2003.
16. M. Becker, E. Werkman, M. Anastasopoulos, T. Kleinberger, “Approaching Ambient Intelligent Home Care System”, Pervasive Health Conference and Workshops 2006, Innsbruck, Nov-Dec 2006, pp. 1-10,.
17. K. Z. Haigh, L. M. Kiff, J. Myers, V. Guralnik, C. W. Geib, J. Phelps, T. Wagner, “The Independent LifeStyle AssistantTM (I.L.S.A)”, 16th Innovative Apps of Artificial Intelligence C, San Jose, CA, July 2004.
18. Open Service Gateway Initiative (OSGi) at (http://www.osgi.org)
19. Java Agent DEvelopment Framework (JADE) at (http://jade.tilab.com/)
20. Foundation for Intelligent Physical Agents (FIPA) at (http://www.fipa.org)
21. Bellifemine, F., Rimassa, G., Poggi, A., “JADE - A FIPA-compliant Agent Framework”. In Proceedings of the 4th International Conference and Exhibition on The Practical Application of Intelligent Agents and Multi-Agents, London. 1999.
22. Pellet, open source reasoner for OWL 2 DL in Java at (http://clarkparsia.com/pellet)
23. SWRL, a Semantic Web Rule language at (http://www.w3.org/Submission/SWRL/)
24. Jena, a Semantic Web Framework for Java at (http://jena.sourceforge.net/)
25. Protégé Home Page at (http://protege.stanford.edu)
26. BeanGenerator at (http://protege.cim3.net/cgi-bin/wiki.pl?OntologyBeanGenerator)