

**SAPI Project**  
**(Automatic System For The Visually**  
**Impaired)**

# SAPI Project (Automatic System For The Visually Impaired)

Pina Russo [russog1@posteitaliane.it](mailto:russog1@posteitaliane.it) (1), Eugenio Manfredonia [manfredoniae@posteitaliane.it](mailto:manfredoniae@posteitaliane.it) (1), Alfredo Troiano [alfredo.troiano@its.na.it](mailto:alfredo.troiano@its.na.it) (2), Francesco Gragnani [f.gragnani@criai.it](mailto:f.gragnani@criai.it) (3)  
(1) Poste italiane, (2) ITS spa, (3) CRIAI scarl

## *Abstract*

Objective of SAPI project is the development of a software platform to provide content and services with innovative characteristic of adaptability, accessibility, flexibility, multi-methodologies and autonomy from the context of application. SAPI project is the first experience that Poste Italiane has in fully open source domain.

## *Keywords*

Adaptation, Intelligent Content, Context, Visually Impaired, Ontology, Intelligent Services.

## **1. Aim of the project**

Project is dedicated to all the users that use contents and services, but special attention is given at ensuring access to the broadest number of user categories that are visually impaired or in general to users with sight problems, disabled or users who exhibit poor computer literacy skills.

The project begins after an in-depth research and study activities about models, methodologies and techniques based on knowledge to improve the personalization of the interaction of man and machines in e-business contexts, to provide useful content and services on-line, to ensure access to the broadest number of user categories and to guarantee accessibility at visually impaired users.

Aim of the project is to adapt content and services on the basis of different features as user profiles, i.e. models of user interests, skills, preferences and habits, access device, interface channel and characteristics of the physical location in which the user-system interaction occurs.

The project not only wants to realize SAPI platform, but also to analyse thoroughly the research about *intelligent content* [1] in order to achieve automatic adaptation of the content based on user profile and context of application. The adaptation process is extended to the services, defining intelligent and adaptative services as services that adapt themselves to the user according to the interaction models previously defined. Intelligent content and services will be able to behave in a different way according to the user who interacts (considering both user profile than user capabilities) and to the context of application (that is, access device, interface channel and environmental characteristics).

## **2. State of play**

SAPI Project started with an existing study in term of legislation, standard, hardware and software that could help blind and visually impaired user, infrastructure such as projects, national and international experiences about adaptative platforms, intelligent services and contents. Once the functional architecture had been outlined, the user models' study and contests and domains of e-business were defined.

Once the functional architecture of the platform had been defined, a match was made between functional blocks and possible Open Source projects, which could be adopted for the blocks.

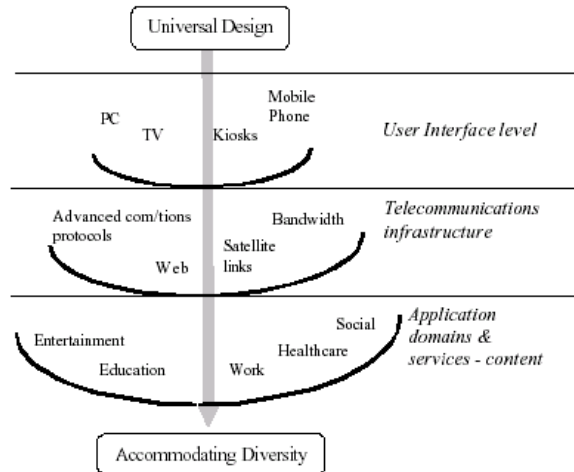
The result is a project list whose integration will be the core of SAPI platform.

At this time, most of the choices are final and the project is resolving communication and interoperable difficulties between different modules of the platform.

### 3. Reference Architecture

#### 3.1. Architectural approach

The approach adopted in SAPI has followed the principles of “Universal Design” which replaces the concept of the average user with consideration of the various user skill levels, requests and preferences, from the very start of the design process, according to proactive development.



**Universal Design Model**

In defining the SAPI architectural model, prime importance has been given to the following architectural drivers:

- **User Context awareness:** SAPI will have to be a platform that is capable of “adapting itself”, of altering its behaviour and the information provided, depending on the context in which the users finds themselves, so as to better satisfy their requirements and needs;
- **Decentralisation of the intelligence in the self-customisable SAPI entities:** makes the Intelligent Services and the Intelligent Content as autonomous as possible in relation to adaptivity to the context of the user;
- **Centralised adaptation** of the non-autonomously customisable SAPI entities (traditional approach);
- **"Ubiquitous" use of the services and content** with "user context sensitivity" characteristics.

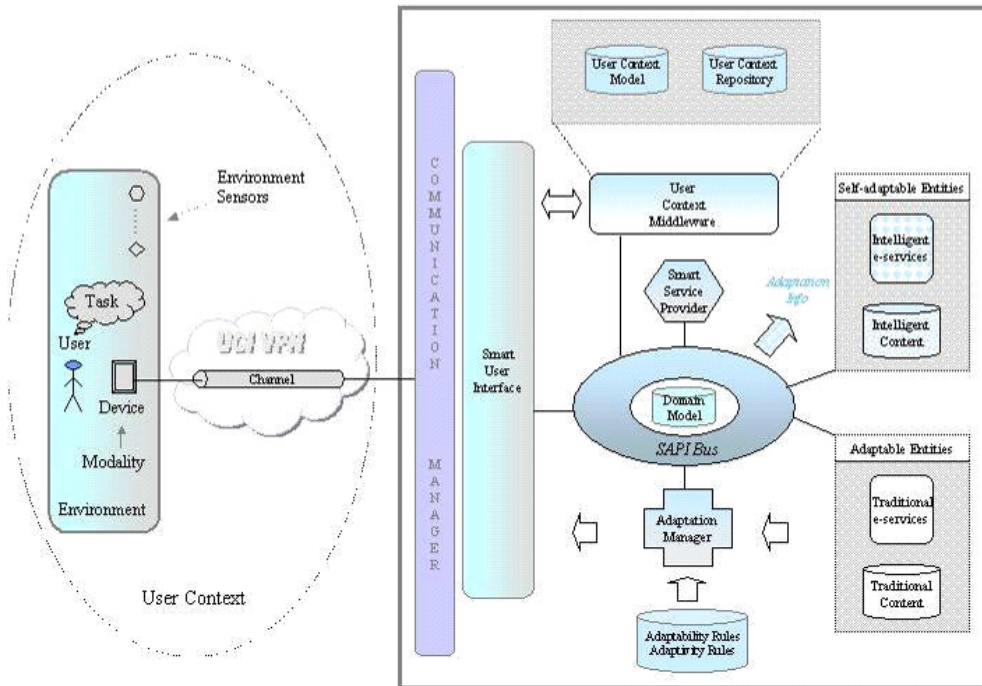
#### 3.2. Functional architecture

The architecture is created with a structure that envisages several basic functional building blocks. Such blocks will be interconnected so that, by means of their cooperation, all the functions of the platform can be achieved.

By means of breaking down the blocks into functional entities, it is easier to provide an overall view of the functional architecture of the SAPI framework, which, on the one hand, relates the functional decomposition of the platform, and on the other, the functional bonds that exist between the various components.

One of the peculiar characteristics of a ubiquitous and pervasive system is the extreme heterogeneity and complexity of the context in which the user is found, constituted by the environment, the various entities present in the environment, in particular the devices and connections (communications channels) the user can use to access the services available, in a given application domain, in the way most congenial to them. By following an ontological approach, the SAPI platform proposes to model the User Context (the

amalgamation of the User Model and the Context Model) and the Application Domain (Domain Model) allocating the properties relating to the individual User Context concepts into a multidimensional space (User Context Space)



**Main components of functional Framework SAPI**

The ontological definition and representation of the User Context will allow a syntactic and semantic interpretation shared by those acquiring and managing the contextual information on one hand (User Context Middleware) and the user (Intelligent Content and Services, adaptation motor) on the other.

We analyse building blocks, with reference to the follow paragraphs in which we underline open source solution that we needs to the implementation of SAPI platform.

### 3.2.1. User Context Middleware

User Context Middleware (UCM) manages, with a suitable logic, knowledge base that includes information about User, Context and Domain Model, furthermore UCM makes these models become available to the other SAPI entities. Changes, grow and updating of the user model will be faster than other models because it evolves in the most dynamic way.

To identify the solution of these problems, we need to pay attention to the following points:

- logic management of knowledge base and of model evolution rules
- logic management of context

### 3.2.2 Adaptation Manager, Self-adaptable entities

Adaptation Manager and Self-adaptable entities are two building blocks that have to execute services and

content adaptation according to the User Context linked to user request. This introduces the necessity to provide an adaptation management.

### **3.2.3 Communication Manager**

Communication Manager guarantees the access security to the platform, user authentication and privacy. To follow this purpose a building block is dedicated to access management.

### **3.2.4. Smart Service Provider, SAPI Bus**

Smart Service Provider has to answer to the user request and to plan the available services. SAPI bus has to ensure the communication among SAPI entities. These characteristics comply with SOA (Service Oriented Architecture) approach which will be adopted.

## **4. Methodology to implement functional macro-blocks**

To realize functional macro-blocks of architecture, we will delineate the solutions to adopt. These solutions cover almost all of functional macro-blocks, leaving possible integration, communication and organization of framework already existent and used as logic components to the design and development of Java source code.

With regard to this, the choice of Java to write source code besides ensuring the portability on different operating system, makes the integration between custom source code and the wide range of available open source tools easier. Java supply JNI (Java Native Interface) to wrap C code in Java modules. So it is possible, in case of need, to write code at the lower levels to interface drivers that Java does not support natively.

### **4.1 Knowledge-base and development rules management**

SAPI consider ontology formalisms [2] to obtain compatibility with Semantic Web project [3]. So the language adopted is OWL[4] (and its correlated: XML for serialization, OWL-QL for querying, etc.). To give an original touch, modelling provide for inserting explicit and implicit relation into ontology, respectively defined by domain expert and inferred by system. Furthermore to allow this extension will be considered probabilistic (Bayesian network [5]) and possibilistic (fuzzy logics [6]) approaches.

Besides ontology formalisms, SAPI does not omit to study other knowledge-based modelling techniques in depth (rule-based profiles, stereotypes, etc.). In reason of that it has been necessary to use flexible software and, at the same time, easily integrable with source code designed and wrote in case of need.

To support diffusion and use of ontology with all of users the contribution of Open Source world was important. It implements solution completer and completer, in respect with W3C recommendation, and allow to use ontology languages (e.g. OWL) and introduce user-friendly graphics interfaces to avoid constraining to define ontology to not much chosen designer.

To define methodologies to represent evolution rules of models (Bayesian network, state finite automaton, etc.) also based on story of user interactions with platform, further studies will occur. Choices made after these studies will aid SAPI designer to ingrate knowledge-base management tools with existing and/or custom reasoners in order to manage model evolutions

### **4.2 Adaptation Management**

It is expected an experimentation of customization, as proposed in [7] [8], to ensure to all of user (able and impaired able) a high quality level of interaction with system.

SAPI platform is to fit its behaviour to: objective (abilities, needs, interests, preferences) and subjective (behaviour) user peculiarities; context (environment, network, host, special devices); current state of

interaction and previous history.

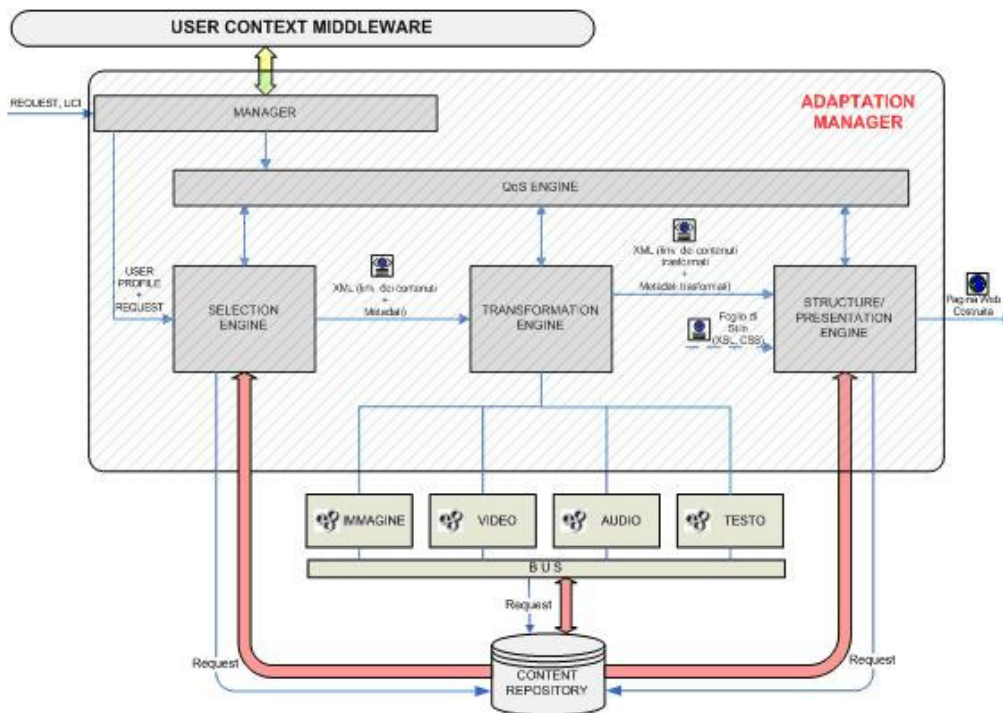
Adaptation could be divided into adaptability and adaptivity concepts. Adaptability refers to selection and modification of user interface during the starting phase of interaction session, with respect of user peculiarities and context known before interaction (e.g. user abilities) and unchanged during a session (e.g. user expertise).

Adaptivity refers to run-time selection and modification of user interface, context and run-time detectable event (e.g. error rate, inability to complete tasks, etc.), in respect with dynamic user peculiarities.

Adaptability and adaptivity allow defining the concept of customization shown in SAPI: “A Custom Information System is An Information System (it provides contents and services) that fit itself according to user peculiarities such as interests, preferences, behaviours and user context, and maximize user experience of final users during their task executions”.

Adaptation [9] will be tested in distributed mode (enriching services and contents with adaptation management logic) and in centralized mode (providing a adaptation management logic applies to custom and existing services and contents)

The figure below illustrates the implementation of SAPI adaptation management with regard to QoS, contents and presentation.



**Adaptation Management in SAPI**

With regard to content adaptation, a hybrid adaptation will be implemented based on real-time transcodification of content properties and on selection of more suitable content to satisfy user request [10].

With regard to functionality adaptation, it is studying a model refers to Abstract Interaction Units [11] to describe expected services. This study will allow to implement the suitable management adaptation logic with regard to functionality exported by services.

### 4.3 Context Management

Mobile systems work in a dynamic context because of two factors: network and devices.

This intrinsic “instability” of context justifies the implementation of adaptive services and mobile agent paradigm is the natural solution to implement just these services[12].

Moreover, in the presence of strongly dynamic network with many devices connecting and disconnecting or

frequently hand over (PDA, smartPhone, devices 3G, etc.) it is better that just the control elements are able to hand over in order to re-modulate their domain. In fact traditional approach monitoring context is insufficient.

Based on these considerations, control component should be autonomous.

This choice is shown the need to study user behaviour during interactions (using a particular kind of sensor called "virtual" sensors).

In reply to these requirements will be use mobile agents which present these features: dynamism, autonomy and responsiveness

Context management consists of implementation of a "Multi Agent System" (MAS).

## **4.4 Communication management among SAPI entities**

It is to be expected a study and a testing of a hybrid technological architecture "populated" by Agents and Services cooperating themselves to implement functionality. According to functionality required it is possible to use Agents and Services in order to reach the goal. For example one Agent could be function entity that, according with value obtained from noise and light sensors, sets volume and brilliance of the screen of a multimedia ATM.

So it is obvious that in SAPI will coexist and communicate heterogeneous entities which, supporting many different languages (ACL, WSDL), will need a gateway: SAPI Bus, it virtualizes different languages providing different SAPI entities with common communication language (SAPI Bus API). With the external world SAPI entities will communicate through standard firewall friendly communication mechanism (e.g. WSDL/XML/SOAP over HTTP) in order to ensure interoperability.

SAPI provides two concepts: user domain adaptation and interaction context adaptation. It offer a suite o services auto-adaptable (intelligent services) subject to an adaptation workflow (not intelligent but custom services)

In particular, entity assigned to adaptation request User Context Middleware, via SAPI Bus, for the instance of User Context and User Context History. According to these instances will be applied adaptation techniques. Furthermore, entity will request Domain Model Manager (DMM) for domain information. With regard to definition and management of a SAPI intelligent services, they will respect SOA paradigm[14].

Then the idea is to implement a SOA where Agents and Web Services could coexist and collaborate. Finally SAPI Bus will strictly interact with Smart Service Provider (SSP) that will allow orchestration of different services

## **5. Open Source Components**

SAPI also uses Open Source paradigm, which involves using well-known technologies (DBMS, Operating System, Web Server, Application Container, etc.). Both designer and developer researchers are able to use these technologies. In this section details about technologies will not contemplate because of many publications with regard to this. The next section will describe components necessary for platform implementation.

### **5.1 Knowledge-base and development rules management**

With regard to knowledge-base management tool, Jena was selected. In fact the other ontology management tools either are based on custom technologies (Cerebra, Autonomy) or evolve just to Jena (Kaon, Sesame, which version is not any more updated).

Jena framework allows to generate data models and to store them in memory or in a database (MySQL, HSQLDB, Apache Derby, PostgreSQL, still less custom solutions)

Jena2, besides introducing OWL-DL support, grown out of work with HP Lab Semantic Web Programme, which provides support on a "best-effort" basis by volunteers through the jena-dev mailing list. Jena implements an inference engine and offer many kind of solutions: GenericRuleReasoner, RDF reasoner and OWL reasoner.

Jena2's inference engine allows data validation, consistency check of schema and data value

Jena is easily integrable with other open source reasoners (e.g. Pellet, Fact++). To deduce from data implicit information risen to interest in using Jena Framework for testing and developed application.

Using ontology to represent a domain of knowledge wants to expose the meaning in term of words of a shared vocabulary, so using an inference engine over a knowledge-base allows exploiting semantics of data to deduce new information.

Ontology design is done using Protégé: an open source tool, developed at Stanford university, that provides graph and interactive editor which allows to modify and easily access to ontology, still less to interact with ontology exploring tree that represent just the ontology.

## **5.2 Adaptation Management**

With reference to adaptation management, open source available reference does not allow to cover the whole logic. Furthermore, the most of adaptation must be designed and implemented working on source code.

Component used will be:

Gaia Image Transcoder [16]: an open source library that allow to convert images on mobile devices using information obtained by WURFL [17] description devices.

Image Magick [18]: a collection of programs that allows identifying, modifying, converting and showing images.

## **5.3 Context Management**

Context management will be based on MAIS system that implements a context adaptation. A society with agents has to be created and it is necessary to plan the agent behaviours because the agents will interact with network component and with the access device of users.

Multi-agent systems are very complex software packages, therefore their building needs a methodological approach. For this reason we chose a framework that complies with standards and exploits a defined methodology.

Reference standards are FIPA specifications. JADE has particular implementation in the use of mobile devices.

JADE [19] is a JAVA development environment for agents. It is a middleware for multi-agent systems ad it provides a software infrastructure so developers have to implement only some characteristics of application.

In this system, software to acquire raw data and software to send structured data to other applications which need it will be properly integrated.

For this reason the following Open Source Components will be adopted:

Context-Toolkit (developed by Georgia Tech's GVU) is a set of JAVA libraries that is aimed at simplifying the process of building context-aware applications.

CMU Sphinx (a project of Carnegie Mellon University in collaboration with SUN) is a speech recognition system entirely written in Java.

DELI (DELivery context LIBrary): an Open Source library developed by HP Labs. It allows Java servlet to answer to HTTP requests that hold the description of the device according to CC/PP (Composite Capabilities / Preferences Profile) or UAProf (User Agent Profile) and it allows querying the achieved profiles.

## **5.4 Communication management among SAPI entities**

Purpose of the SAPI bus is to make different languages virtual and to provide a unique communication language for the different entities which are part of the system. In this context the integration among different software entities is necessary and the adoption of Open Source is useful.

In SAPI we plan the parallel use of Web Services and Agents, implementing a complex structure of WSIGS (Web Service Integration Gateway Service) to realize the SAPI bus. In particular we evaluate IONA solution (CELTIX) an Open Source version of the commercial Enterprise Service Bus ARTIX and LogicBlaze solution (ServerMix) that is the first ESB JBI-Compliant.

Definition and management Intelligent services a non intelligent and adaptable SAPI services will be conform to the SOA architectural paradigm. Further studies are planned before its planning and implementation.

Same effort is required to realize the Smart Service Provider that is the block that orchestrate the service supply, in other words, it connect several atomic services in an intelligent way, maybe that services are supported by ontological description. Moreover Smart Service Provider defines the workflow on the atomic services. It could be realized as a software agent or as a service.

## **5.5 Access platform management**

SAPI platform wants to propose itself as a tester project to investigate the possibility to make users access following ETSI UCI (Universal Communications Identifier) standard [13].

SAPI project propose the concept of VPN UCI for the following reasons:

- to ensure an unique identifier for a person during the interaction with the information system and to develop trusted security for communications with the platform;
- to code and to manage important characteristic of (explicit) user profile about his needs (for disabled and sight problems people) and his communication preferences (such as access device, preferred language, acceptable languages, etc...);
- to send to communication infrastructure the user information in a secure and trusted way.

To conform to the model proposed by ETSI access platform management has to plan the adoption of a system based on agents to query the entities defined in the standard (Personal User Agent, that cares about user profiles, and Service Agent, that is adopted to describe the service). Furthermore, to consider a SOA compliant system it is necessary to make the services available to the users.

All the technologies that ensure access security, are widely supported by tools and systems adopted by Open Source Community such as Operative Systems (Linux in all its versions, FreeBSD, Slackware, etc..), application container (Tomcat, Jboss, etc..), web server (Apache, Roxen, etc..).

Anyway connection between user and Authentication Server (AS) has to be protected because data move on public internet. Therefore VPN has not to consider only UCI environment and the set of its entities, but it has to be appropriately extended to include user too.

## **6. Conclusions**

SAPI project research, aimed to study and test of Innovative and Self-customizing Informative Systems, considers Open Source Software necessary and it cannot be disregarded.

To control the work tool is a requirement to development of complex and innovative framework software. The possible access to the source code allow to the developers to study the working and to modify it appropriately.

Open Source Software that have been identified, provide software that allow the development of a wide part of adopted solutions to realize SAPI prototype.

In this background the experience done with SAPI project represent a starting point and a reference both the successive research project and to use platform to the development of product and services implemented by Poste Italiane team work.

## 7. References

- [1] <http://www.acemedia.org/aceMedia/project/index.html>
- [2] Fensel D. Ontologies: A Silver bullet for Knowledge Management and Electronic Commerci. Springer-Verlag, 2000.
- [3] Michael C. Daconta, et al. The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management. John Wiley & Sons; (2003).
- [4] OWL Web Ontology Language Overview. W3C Proposed Recommendation 15 Dec 2003. Deborah L. McGuinness and Frank van Harmelen eds.
- [5] D. Heckerman, D. Geiger, D. Chickering. Learning Bayesian networks: The Combination of Knowledge and Statistical Data. Machine Learning, 20:197-243, 1995.
- [6] LA Zadeh. Fuzzy Logic. Computer, 1988.
- [7] Van Setten, M. (2001). Personalised Information Systems: Analysis of existing personalized information systems. Giga)E/D2.3 and Duine/D1.1, Ensc(ede: Telematica Instituut (TI/RS/2001/036).
- [8] Adomavicius G. and A. Tuzhilin. 2001. Using Data Mining Methods to Build Customer Profiles. IEEE Computer, 34 (2), 74-82.
- [9] Biemans, M. C. M., van Setten, M., van Vliet, H., Alberink, M., Eertink, H., de Heer, J., van Kranenburg, H., Kruse, H., de Poot, H., Reitsma, J., Slagter, R., Teeuw, W., & Veenstra, M. (2002). Adaptation. An integrated view on adaptation in telematics, Telematica Instituut.
- [10] Chung-Sheng Li, Rakesh Mohan and John R. Smith, "Multimedia Content Description in the InfoPyramid," IEEE Proc. Int. Conf. Acoust., Speech, Signal Processing (ICASSP) , Seattle, WA, Special session on Signal Processing in Modern Multimedia Standards, May, 1998
- [11] G. Ausiello, E. Bertini, A. Cali, T. Catarci, S. Kimani, G. Cantucci. Designing Adaptable Multi-device Applications. Progetto MAIS (Multi-channel Adaptive Information Systems), Fondo FIRB, Programma Strategico Tecnologie abilitanti per la Società della Conoscenza-ICT. [http://black.elet.polimi.it/mais/documenti\\_pubblico/Isemester/r7.3.2.pdf](http://black.elet.polimi.it/mais/documenti_pubblico/Isemester/r7.3.2.pdf).
- [12] G.Ventre et alii (CRIA), DICAMS: Un Middleware MultiAgente per Applicazioni Context-Aware, 2006
- [13] European Telecommunications Standards Institute (ETSI), Universal Communications Identifier (UCI); System framework, [portal.etsi.org/docbox/EC\\_Files/EC\\_Files/eg\\_202067v010101p.pdf](http://portal.etsi.org/docbox/EC_Files/EC_Files/eg_202067v010101p.pdf)
- [14] Eric Newcomer, Greg Lomow, Understanding SOA with Web Services. Addison Wesley (2005).
- [15] <http://protege.stanford.edu/>
- [16] <http://gaia-git.sourceforge.net>
- [17] [wurfl.sourceforge.net](http://wurfl.sourceforge.net)
- [18] [www.imagemagick.org](http://www.imagemagick.org)
- [19] <http://jade.tilab.com/>