An Embedded Service Platform for the Vehicle Domain

José Santa, Antonio F. G. Skarmeta, Benito Úbeda
Department of Information and Communications Engineering
Computer Science Faculty
University of Murcia
Campus de Espinardo, 30071 Murcia, Spain
Email: josesanta@dif.um.es|skarmeta@dif.um.es|bubeda@um.es

Abstract— **Onboard services are becoming the cornerstone in vehicle equipment. Since navigation systems opened the way past decade, location based services (LBS), as well as new multimedia features, are more and more demanded by drivers. Vehicle manufacturers are working in attractive and useful solutions to be included in their cars. However, an important drawback of this development arises in the proliferation of devices in the driver compartment and the lack of a shared communication interface with the exterior. The space in a vehicle is limited and a friendly interface between the driver and the onboard services is needed. Due to these reasons, our work has been directed to develop an open service platform for vehicles with several communications capabilities. An embedded computer is used as the on board unit (OBU), and a multiplatform software architecture has been designed and deployed on it to let the implementation of onboard services. The navigation sensors installed in our prototype vehicle are used to implement location based services, and several network devices cover all connectivity requirements. Specifically, cellular networks are used in a peer to peer (P2P) based network architecture valid for communications among vehicles and between the car and the road infrastructure. In addition, several services have been implemented and tested to probe the feasibility of the whole system.**

I. INTRODUCTION

Due to the growing interest that current society has in technology, new products in the fields of information and communication technologies are emerging in new environments still unexploited. In this way, vehicles are a perfect frame for installing a lot of useful functionalities traditionally available at work or home environments [1]. However, this expansion needs a suitable hardware and software support adapted for the market and user demands.

So far, the amount of services that passengers could use in a vehicle required a large hardware deployment. Each new functionality is implemented in a new device which had to be installed. Up to now, this procedure has been feasible because the amount of onboard services has been limited to a radio, a CD player and, sometimes, a GPS navigator. However, nowadays the new location based services (LBS), just as the ones emplaced in other fields, make this production methodology a non scalable model. At this point, general purpose computers are a better option than the dedicated ones. Adding new services as executable software over a common hardware platform shows several advantages. First, the business model suffers a radical change because the service updates do not require new costs in hardware installation. At the same time, the user faces a much more intuitive and common interface with the system, similar to a standard PC.

Communications are essential these days as well. Practically everybody can connect to Internet at work or at home, and even in mobility environments using several portable devices. The car is another place where the user stays during long periods, so connectivity here is valuable. However, apart from the direct usage of the network, some onboard services need communication capabilities. Although every service could implement a proprietary protocol starting from the TCP/IP stack, a common high level communication technology would be useful in the development of a scalable service architecture.

If we want to introduce a service architecture really integrated in the vehicle, it is necessary an appropriate hardware support. In other words, several devices will be essential if we are interested in LBS, such as a GNSS (Global Navigation Satellite System) sensor and a cellular communications link, at least. [2] and [3] show two car architectures suitable for developing driving state oriented software. INS (Inertial Navigation System) sensors, odometry captors, and satellite navigation sensors are key factors in both systems, where the embedded autonomous control software merges all the information which comes from them in order to take a decision about the vehicle movement.

In addition to the suited hardware, the proper software platform necessary to deploy all the services must also be selected. The modularity of the implemented services, the portability of the services between different operating systems, and the easiness of deployment are requirements which have to be taken into account in order to obtain a robust architecture. The Open Services Gateway initiative (OSGi) [4] presents a large amount of features according to these requirements, against others extended solutions as Jini and Service Location Protocol (SLP).

Our work has been centered in the design and development of an extensible architecture for services based on an embedded general purpose computer, improved with communications capabilities at the service level. Using an onboard computer as a service gateway, the space problems in the implementation of new functionalities is solved. At the same time, our proposal shows a programming framework for services based on layers, which promotes reusability and modular development. A peer to peer (P2P) approach has been included in the architecture with the aim of providing vehicle to vehicle, vehicle to infrastructure and infrastructure to vehicle communications (VVC, VRC and RVC), whose usefulness is described in [5]. The whole system has been enriched with a large number of services, showing the feasibility of the proposed solution. The architecture has been included in a widely sensorized vehicle, which let us to create several context aware services. Literature about similar concepts is limited, mainly due to the major contributions are carried out by private companies. In [6] a framework to develop vehicle onboard software oriented to the user is shown. Our work, in contrast with this approach, does not show an architecture focused in programming APIs for final applications; we are interested in giving facilities to the programmer to develop visual applications, but also in the implementation of "drivers" for physical devices, very changeable during the vehicle life cycle. [7] is another example of a solution for onboard service development. This is mainly centered in a Jini and JXTA middleware to allow communications between local and remote services. However, the onboard platform is less integrated than ours and the communications between service edges are not managed using a generic strategy. [8] shows a framework really integrated in the onboard computer. This is developed as a .NET based class hierarchy, so new services must be programmed against the framework. However, a low level underlying technology (.NET) is not comparable with the one given by OSGi. [9] presents an architecture which tries to avoid the explained problem of hardware services. The solution is really directed to the provider side, so it is not suited to develop software oriented to the vehicle hardware. In [10] a communication technique between the vehicle and the road infrastructure based on the DSRC technology is presented. The main drawback here is the cost of deployment at the road edge, which requires the installation of multiple DSRC devices, in contrast to our approach, based on cellular networks. The same problem can be observed in [11], where a handover system for 802.11 networks in vehicles is presented. Although the idea could be valid for urban or fixed railway zones, the deployment is limited by the cost.

The rest of the paper is organized as follows. In Section II the service architecture is described. Section III explains the high level communication system developed. In Section IV details about a prototype of the whole system are presented. Finally, Section V contains the conclusions of our work.

II. A MULTIPLATFORM OSGi BASED ARCHITECTURE FOR SERVICE DEVELOPING

As it has been stated, OSGi is presented as the suitable container for a set of services implemented as PC software. Initially, the OSGi conception was based on the creation of residential gateways where the software which makes the house "intelligent" was installed. In [12] it can be found a good example where OSGi is used as the basis for the development of an intelligent house prototype oriented to the localization of inhabitants. However, the OSGi advantages have extended to other fields. The vehicle environment is one of these new applications, where the onboard computer can be considered as a service gateway as well.

Fig. 1 shows the system designed to create services over the onboard unit (OBU). All the sensors included in the

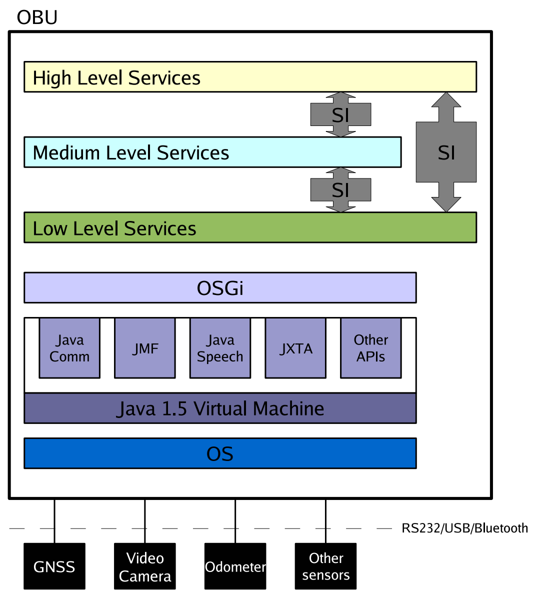


Fig. 1. OBU service architecture

vehicle are connected to the onboard computer via RS232, USB or Bluetooth links. The computer is an ordinary PC with an interchangeable operating system and a Java Virtual Machine (JVM). Several programming APIs are situated over it to let the implementation of a lot of services. Java Speech, for instance, provides a speech synthesizer to Java programs. OSGi is located over the Java basis, making the computer able to contain several services. In fact, as [13] summarizes, the software entities installed over OSGi are called *bundles*, which can offer services. Back to the diagram, the set of elements visible over the OSGi layer are the services implemented in the platform. These are classified according to its abstraction level. So they are divided into low, medium and high level services. Low level services are software to access to several devices in the vehicle. Medium level services act as middleware between low and high level services. In this sense, this layer performs transformation and adaptation tasks, improving the functionality offered by low level services. Finally, high level services are bundle software with a user interface. This hierarchical structure of services has a double purpose. Firstly, the creation of new services is easier because of the modular programming. If any functionality is used or is expected to be used by several applications, this can be encapsulated as a service. On the other hand, the problems for accessing to real devices when several services require the use of a specific one are solved, so synchronization problems are avoided. This issue is really important in widely used sensors, as the GNSS one. In addition, services situated in the medium and low level layers of the architecture carry out caching of information, so it is possible to serve some requests without doing a new petition to a sensor, for instance.

Communications between layers is carried out by service interfaces (SI). Each layer defines a set of service interfaces which indicate the available functionalities. A SI is a Java

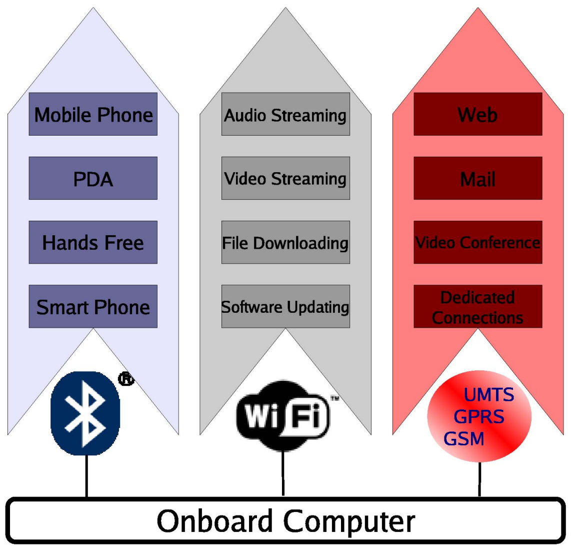


Fig. 2. Communication technologies considered

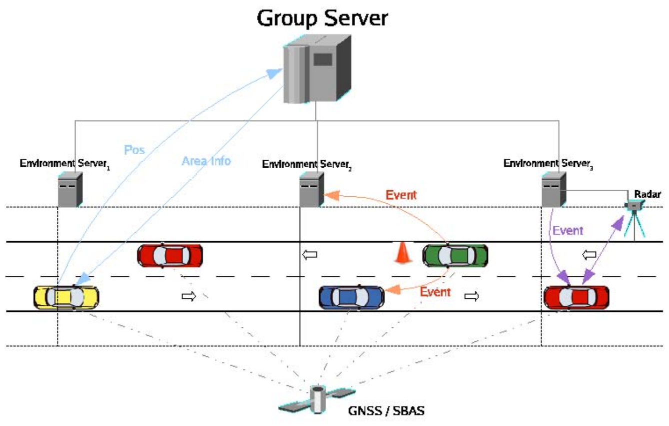


Fig. 3. High level communication architecture

interface which may be implemented by one or more OSGi bundles to offer services. If a bundle situated in the upper layers needs a feature provided by one of the SI, a query to the OSGi core infrastructure with the SI as a parameter is launched, in order to receive the set of available services which implement the functionality.

III. VEHICLE COMMUNICATIONS AT THE SERVICE LEVEL

III-A. THE IMPORTANCE OF VEHICLE COMMUNICATIONS

Although the direct usage of Internet by the driver and the occupants of the vehicle is an interesting functionality, it does not present a real innovation. Nowadays onboard services which use networks for creating communications between vehicles and between the car and the road infrastructure are the main point of researching. In this sense, different communication technologies have been studied in the vehicle domain. In [14] several wireless technologies are considered. Bluetooth, ZigBee, UWB and Wi-Fi are adapted to the car. These technologies are useful in low and mid range environments but, sometimes, a global connectivity to Internet is necessary. Cellular networks are the solution to this problem these days.

With the aim of providing network capabilities to our service architecture, three technologies have been chosen. These can be observed in Fig. 2. Through a Bluetooth adapter, it is possible to maintain a short range connection with other devices. Hands free devices, PDAs, and last generation mobile phones have popularized Bluetooth. As well as being used independently, these gadgets could be useful in controlling the onboard computer, for example. With the WiFi connection a wide band link can be established, useful in the reception of multimedia contents, file downloads, and even software updates for the onboard computer. Some services in the vehicle can use this temporary wide band to establish a connection in places such as parkings and petrol stations. Finally, cellular networks allow today permanent data connections almost everywhere. In Europe the current deployment of UMTS has increased the connection speed in urban and some interurban areas up to now. Although the data rates in GPRS/UMTS networks are far from the WiFi ones, a continuous link is important in services which demand a connection with a central station, such as road pricing or warning systems, for instance.

III-B. A SERVICE LEVEL COMMUNICATIONS ARCHITECTURE

As it has been stated previously, cellular networks through GPRS/UMTS have a special interest in this work. Using a P2P approach over cellular networks, the vehicle can receive and send contextual information about its current environment. Fig. 3 shows a general diagram of the high level communication architecture proposed. The traffic zones are organized in coverage areas, every one using a different P2P communication group. Vehicles are able to move from one P2P group to another through a roaming process between coverage areas. This roaming is based on the location of the vehicle, provided by the GNSS sensor. Information about areas is received from the Group Server using a TCP/IP link over GPRS/UMTS. A local element called Environment Server manages special events inside the area. Event notifications are sent and received by service edges located either at the vehicle or the road side (Environment Servers). All messages emitted are encapsuled in P2P packages. Two different techniques of emission have been developed, so a P2P message can be broadcasted in the area or sent to a specific vehicle.

Apart from the designed architecture, in Fig. 3 the three most representative sceneries are exposed from left to right. In the first one, a vehicle is passing from one area to another. The Group Server gives the P2P parameters to maintain the communication in all the active services. Because the Group Server sends the area geometry to the vehicle, this have to communicate with the central entity only when it is necessary, saving communication resources. In the second scenery, a safety service located in the vehicle notifies a repairs event. This event is broadcasted, so all vehicles located in the area receive the warning. Because the environment server receives all broadcasted messages, it can process any event which requires a special treatment, such as a collision, for example. Messages from special services (repairs, collision…) are forwarded by the Environment Server to the adjacent areas, in order to improve the warning mechanism. The last scenery in the diagram shows how the environment server can be connected to the rest of the road side infrastructure, which may be composed of speed radars, identification sensors (RFID), video cameras, etc. Thanks to this, local events can be notified to the vehicles in the area, and several reports can be sent to the central entity.

IV. PROTOTYPE DETAILS

The described platform has been implemented and tested over a real system. Moreover, several services have been developed to demonstrate the viability of the proposed solution.

IV-A. TECHNOLOGICAL ISSUES

The hardware platform used is a prototype of a vehicle widely sensorized used at the University of Murcia [2] in several research projects regarding autonomous navigation with GNSS and inertial sensors. Thanks to the hardware installed in the vehicle, it is possible to create services dependant on the current state of the car. Among all the sensors it is worth noting the odometer captors situated in each wheel, and the GNSS receiver. Odometry is useful to monitor the movement of the vehicle, while the GNSS receiver is a key tool for location tasks. The prototype vehicle is shown in Fig. 4. It includes an onboard PC based on a single board computer (SBC) architecture of VIA. A tactile screen allows the interaction with the user pressing directly on it. The common physical interface through mouse and keyboard are located in the lower part of the dashboard. A Linux Fedora Core 4 operating system and a Java Virtual Machine 1.5 have been used. Two OSGi frameworks have been probed and installed, Knopflerfish [15] and Oscar [16]. Both are open source implementations of the OSGi Release 3 [4]. The vehicle is provided as well with Bluetooth, WiFi and GPRS/UMTS transceivers.

Regarding to the high level communication architecture, all the road side entities have been developed as individual servers implemented in Java 1.5, using standard PCs with the Linux Fedora Core 4 operating system. The vehicle edge has been tested with a high level service included in the software platform.

IV-B. IMPLEMENTED SERVICES

The services which are deployed to be included in our architecture have special features in their JAR archive, distinguishing them from the rest installed in the OSGi framework. A special service has been developed to start, stop and update the rest of the implemented services. The low level services developed so far are:

- *GNSS Positioning*. It is a generic service for accessing to several positioning sensors.

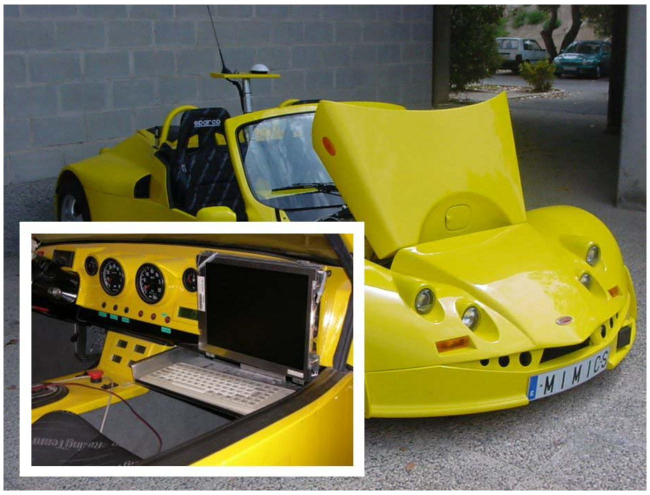


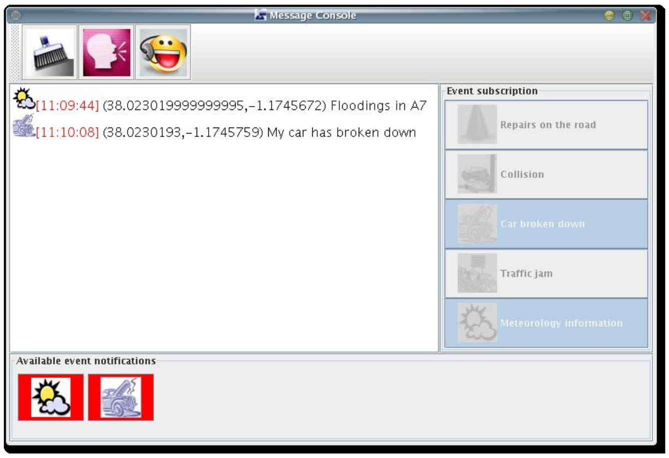
Fig. 4. Prototype vehicle used in the development

- *Video Camera*. Service for connecting with a camera situated in the rear part of the vehicle.
- *Speech Module*. Encapsulates a voice synthesizer available for the rest of services which can emit spoken events, so the user do not have to avert his sight from the road.
- *Odometer*. This service provides information from the odometry sensors.

As middleware services (implemented in the second layer) a SBAS processor utility and a JXTA manager are included. The SBAS tool contains several functionalities concerning the Satellite Based Augmentation Systems (SBAS). In this sense, SBAS information is used for correction and integrity purposes [17]. The JXTA module contains the resources used in P2P communications. JXTA (JuXTApose) [18] presents a P2P system which allows resource sharing through a *virtual network*. This underlying network is formed internally, so the programmer can abstract from the internal P2P details. The Java implementation and the group based system make the JXTA technology suitable for our P2P design.

High level services have a direct relation with the user. The set of implemented high level services are:

- *Integrity Monitor*. It is an application used to monitor the integrity of the location emitted by the positioning system. For this purpose, it uses the SBAS Processor Utility service, but the Speech Module service is necessary as well.
- *Rear Visor*. With this service the user can see the rear part of the vehicle in parking manoeuvres, thanks to the Video Camera service.
- *Media Player*. It is a multimedia player which can load several video and audio formats.
- *Navigator*. This program has navigation and electronic fee collection (EFC) functions thanks to a GIS (Geographical Information System), and the GNSS Positioning and SBAS Processor Utility services.
- *Message Console*. This is the vehicle edge of the service level communication architecture previously explained.



[11:09:44] (38.023019999999995,-1.1745672) Floodings in A7
[11:10:08] (38.0230193,-1.1745759) My car has broken down

Fig. 5. Message Console service

IV-C. THE HIGH LEVEL COMMUNICATIONS ARCHITECTURE IMPLEMENTATION AND TEST ENVIRONMENT

As can be read from the previous section, the high level communication infrastructure proposed in Section III-B has been implemented using our service arquitecture. Fig. 5 shows the Message Console service, which is the vehicle edge of the architecture. Using the P2P middleware, an event based mechanism has been created to notify security warnings. Five types of events are available in the right box, so the user can subscribe to any of them. Because every event type is implemented like a service in the communication architecture, its availability depends on the current area. If the service is active, the corresponding event can be thrown, and the icon is available in the lower part of the window. The received events are shown in the central area.

The Group Server and Environment Server entities have been deployed in local hosts located in our laboratory. Environment Servers do not have to be located physically at the coberage area. Several probes has been carried out in a predefined circuit along a highway near the University of Murcia, probing the feasibility of our implementation. In Fig. 5 the two services the user has subscribed to are active in the current area specified along this highway, so they appear as available notifications. Two events have also been received, as can be seen.

V. CONCLUSIONS

A modular and extensible architecture for creating onboard services has been designed. This platform has been conceived for a general purpose computer, and the prototype designed shows a SBC as a suitable device for this purpose. A sensorized vehicle has been useful to create several services oriented to the vehicle state. A friendly interface has been considered, using a PC based OBU adapted to the vehicle environment and improved with spoken alers, so the user do not have to avert the sight from the road. Also, vehicle communications have been considered in our proposal, which deals with VVC, VRC and RVC. This design uses P2P groups to create communication areas where services can send messages in a easy way.

The proposed solution implies a new business model with cost reductions in development, deployment and, over all, updating of existing systems. Current active works in this line are directed to complete the model at the road edge, adapting the information sent to vehicles according to the driver profile.

VI. ACKNOWLEDGMENTS

The authors would like to thank the Spanish Ministerio de Educación y Ciencia for sponsoring the research activities under the grant AP2005-1437, in frames of the FPU program. Special thanks as well to the Spanish Ministerio de Fomento for its continuous support in vehicular research.

REFERENCES

[1] Craig Simonds. “Software for the Next-Generation Automobile”. *IT Professional, November/December 2003, pp 7-11. January 2003.*

[2] *Skarmeta A.G., Martínez H., Zamora M.A., Úbeda B., Gómez F.C, Tomás L.M. “MIMICS: Exploiting Satellite Technology for an Autonomous Convoy”. IEEE Intelligent Systems. N IV. V. vol. 17, pp. 85-89. July 2002.*

[3] *Massaki Wada, Xuchu Mao, Hideki Hashimoto, Mami Mizutani, Masaki Saito. “iCAN: Pursuing Technology for Near-Future ITS”. IEEE Intelligent Systems, January/February 2004, pp 18-23. January 2004.*

[4] *OSGi Alliance. OSGi web site. http://www.osgi.org*

[5] *Luisa Andreone, Michele Provera. “Inter-vehicle communication and cooperative systems: local dynamic safety information distributed among the infrastructure and the vehicle as ”virtual sensors”to enhance road safety”. ITS Hannover 2005, Hannover. June 2005.*

[6] *E. C. Nelson, K. V. Prasad, V. Rasin, C. J. Simonds. “An embedded architectural framework for interaction between automobiles and consumer devices”. 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS’04). May 2004.*

[7] *Luciano Baresi, Carlo Ghezzi, Antonio Miele, Matteo Miraz, Andrea Naggi, Filippo Pacifici. “Hibrid service-oriented architectures: a case-study in the automotive domain”. 5th International Workshop on Software Engineering and Middleware (SEM’05), Lisbon. September 2005.*

[8] *Zoltán Benedek. “A Framework Built in .NET for Embedded and Mobile Navigation Systems”. 2nd International Workshop on .NET Technologies, Czech Republic. May 2004.*

[9] *Chatschik Bisdikian, Isaac Boamah, Paul Castro, Archan Misra, Jim Rubas, Nicolas Villoutreix, Danny Yeh, Vladimir Rasin, Henry Huang, Craig Simonds. “Intelligent pervasive middleware for context-based and localized telematics services”. International Workshop on Mobile Commerce, Altanta. September 2002.*

[10] *Gen Hattori, Chihiro Ono, Satoshi Nishiyama, Hiroki Horiuchi, “Implementation and Evaluation of Message Delegation Middleware for ITS Application”. 2004 Symposium on Applications and the Internet-Workshops (SAINT 2004 Workshops), Tokyo. January 2004.*

[11] *Toshiya Okabe, Takayuki Shizuno, Tsutomu Kitamura, “Wireless LAN Access Network System for Moving Vehicles”. 10th IEEE Symposium on Computers and Communications (ISCC05), La Manga del Mar Menor. June 2005.*

[12] *Sumi Helal, William Mann, Hicham El-Zabadani, Jeffrey King, Youssef Kaddoura, Erwin Jansen. “The Gator Tech Smart House: A Programmable Pervasive Space”. Computer. Vol. 38, no. 3, pp. 50-60. March 2005.*

[13] *Choonhwa Lee, David Nordstedt, Sumi Helal. “Enabling Smart Spaces with OSGi”. IEEE Pervasive Computing. Vol. 02, pp 89-94, Jul-Sept. July 2003.*

[14] *Tomas Nolte, Hans Hansson, Lucia Lo Bello. “Wireless Automotive Communications”. Euromicro Conference on Real-Time Systems (EC-TRS 05), Palma de Mayorca. July 2005.*

[15] *Knopflerfish OSGi web site. http://www.knopflerfish.org*

[16] *Oscar OSGi web site. http://oscar.objectweb.org*

[17] *José Santa, Benito Úbeda, Antonio F.G. Skarmeta. “Monitoring the position integrity in road transport localization based services”. IEEE Vehicular Technology Conference 2006 Fall, Montreal. In press.*

[18] *JXTA web site. http://www.jxta.org*