Mobile Netw Appl (2010) 15:137–147
DOI 10.1007/s11036-009-0163-8

A Framework for Engineering Pervasive Applications Applied to Intra-vehicular Sensor Network Applications

Antonio Coronato · Giuseppe De Pietro · Jong-Hyuk Park · Han-Chieh Chao

Published online: 29 April 2009

Abstract The vehicle sector is one of the most exciting application areas for wireless ad hoc networks and pervasive computing technologies. Vehicles are constantly being equipped with more sensors and devices able to collect real-time data on traffic, vehicle condition, passenger health, and so on. Being a relatively new field of application, this area needs tools and methodologies in order to specify requirements and prototype applications. This paper presents a Framework for Pervasive Applications and describes how it can be customized in the case of Vehicular Applications. The framework consists of a set of software requirements, some metrics, and some middleware services for rapid prototyping pervasive applications. In addition, the paper introduces an ongoing project that aims at granting high-quality on-road transportation services for fragile food (e.g. wine). In particular, its objective is to provide a system able to monitor several conditions, like temperature, humidity, light, shocks, etc., which carried products are subject to during transportation.

Keywords pervasive applications · intra-vehicular sensor network

A. Coronato · G. De Pietro
Institute of High Performance Computing and Networking, CNR, Naples, Italy

J.-H. Park (✉)
Department of Computer Science and Engineering, Kyungnam University, Masan, South Korea
e-mail: parkjonghyuk1@hotmail.com

H.-C. Chao
Institute of Computer Science & Information Engineering and Department of Electronic Engineering, National Ilan University, I-lan, Taiwan, Republic of China

1 Introduction

Vehicles are becoming a fascinating field of applications for sensors, and mobile and pervasive technologies in order to provide advanced services. The variety of sensors available on sensor motes, combined with the relative simplicity with which they can be distributed around an environment, such as a car, makes this an attractive prospect. Moreover, the maturing of some pervasive technologies like middleware infrastructures, software platforms and standard specifications that are focused on the possibility of an easy integration of sensors and mobile devices—the Open Service Gateway initiative (OSGi) [15] is just one of the most relevant examples in this sense—can really make VANET applications rapidly move from research and academic activities to industrial and commercial opportunities. This is confirmed by the great number of projects that record the joint of interest of research centers and industries, as reported in the next section.

However, since this is a relatively new field of application, there are still many research challenges and opportunities. Among these, it is necessary to develop methodologies and tools to specify and prototype applications. Indeed, several prototype projects have been proposed; typically, each project focuses on a specific set of topics/requirements for vehicular applications. In the next section, we report some examples of active projects or research activities that differ from each other in content; but, share the need to have rapid prototypes of vehicular applications.

In this paper, we present a framework for engineering and prototyping pervasive systems. The framework relies on the definition of software requirements for generic pervasive applications and the characterization of the behavior of the application in terms of such requirements.

The possibility of having a structure of requirements enables designers to focus on the relevance that each requirement should have in the current application. As we are going to see, pervasive applications present quite different characteristics from this point of view and require different attention compared to singular software requirements. Indeed, the possibility of having natural interaction mechanisms is quite important in smart environments, but it is not relevant in the case of remote monitoring applications for which functionalities are not accessed by humans. Metrics enable us to quantify the relevance of each software requirement. In particular, we have adopted five categories —*Useless*, *Quite Interesting*, *Interesting*, *Very Interesting* and *Crucial*—to classify software requirements.

Finally, the framework has been applied to an intra vehicular sensor network based application, which is being constructed in order to monitor the transportation of fragile food.

The rest of the paper is structured as follows. Section 2 describes some research projects on VANET and vehicular applications. Section 3 introduces the framework. Section 4 describes the case study. Section 5 concludes the paper.

2 Related works

In the following section we report some related works or relevant projects active in the field of vehicular applications.

A distributed Mobile Sensor Computing System (CarTel) is described in [3]. Among other objectives, the authors focus on the possibility of rapidly prototyping intra-vehicular sensor networks by means of a simple programming interface. The goal is to centralize and simplify the development of mobile sensor network applications. CarTel should be a framework that enables the realization of applications without considering distribution or mobility.

Within the DySCAS (Dynamically Self-Configuring Automotive Systems) project, a middleware infrastructure able to tackle the complexity in distributed embedded electronic systems is introduced. Such a middleware specifically focuses on self-managing capabilities that, as shown in the paper, represent a main requirement for several kinds of vehicular applications [2].

In [4], the primary goal is to construct a wireless sensor network which operates in and around a car, with the long-term goal of building a clamp on the automotive diagnostics system utilizing a sensor network. The success criteria for this project is as follows. First, the network includes nodes operating within the passenger compartment, external to the car, and within the engine compartment. Secondly, each node periodically takes a temperature reading and sends it to a base station node. Thirdly, the base station node is also capable of receiving packets whether it is within the passenger compartment or nearby. The authors focus on an architecture for security and access controlling Intelligent Transportation Systems. The proposed framework deals with ensuring security for the communication infrastructure by implementing access control at both the data link layer and the network layer. The authors demonstrate the applicability of the framework developing a case-study based on CVIS (Collaborative Vehicle Infrastructure system) [1].

Next, the DRiVE project (Dynamic Radio for IP Services in Vehicular Environments) is presented. In contrast to the previous example, this project aims at to enable spectrum-efficient high-quality wireless IP communication in a heterogeneous multi-radio environment to deliver in-vehicle multimedia services. The paper focuses on the network architecture for IP over multi-radio access networks and traffic control aspects within the emerging "DRiVE" network [6].

Another interesting framework is presented in [8]. It consists in a reconfigurable demonstrator that allows for an experimental validation of vehicular networking research. In addition to this, the authors present an application developed on top of OSGi.

The problem of complex vehicle applications requiring the integration of different software technologies is faced in [9]. In that work, the authors demonstrate that interoperability is one of the key issues in complex VANET applications and stress the SOA paradigm as the most suitable solution for integration.

Alternatively, some challenges of Engineering Ubiquitous Systems are faced in [5]. The authors present approaches that vary from agile and informal methods to very formal methods. However, all the methods and models defined aim to support a systemic approach to development. By adopting such approaches, the designer will model the system together with the relevant part of its environment [7].

Other interesting software engineering studies on vehicular applications are: [10], is which the authors focus on information accuracy in in-car information systems. In particular, they consider the relevant issue of designing an in-vehicle information system that does not negatively affect cognitive processing and driving performance by delivering inappropriate and/or inaccurate information to the driver; and [11], is which a simulation tool for deploying VANET applications is shown. In particular, a series of simulation models that account for street layout, traffic rules, multilane roads, acceleration-deceleration, and RF attenuation due to obstacles are presented. Such simulation models can be adopted successfully while prototyping VANET applications that apply particularly to urban environments.

Cars can be interconnected and can realize a Peer-to-Peer network as shown in [12]. In particular, the authors propose an alternative approach for developing traffic and weather monitoring applications that relies on the use of mobile devices within the car to get mobile access to the internet. Thus, the authors mainly focus on the issue of connectivity.

Security issues are discussed in [13] and [14]. In [13], the authors present a survey on many security aspects relating to automotive applications. In contrast, in [14] the authors mainly focus on aspects strictly related to ad hoc networks for vehicles.

Finally, the possibility of scaling VANET applications to wide area networks is conosidered in [16] and [17].

3 The framework

3.1 Software requirements

In this section we identify a set of software requirements that a generic pervasive application should support. As we are going to see, the requirements may differ significantly depending on the kind of pervasive application. For the framework, we have identified two levels of requirements, namely external and internal requirements. External requirements are in a sense macro requirements and can be seen as composed of internal requirements, which are in turn those that are used to determine external requirements. Figure 1 shows the set of external requirements that we believe can characterize a pervasive application. We define external requirements as follows:

- Pervasivity—This is the capability of the system to provide the final user with its services independently of his/her location and/or device;
- Context-Awareness—This is the capability of the system to be aware of the surrounding context;
- Usability—This is the degree of usability of the system services, as perceived by the final user;
- Autonomicity—This is the capability of the system to self-manage;
- Proactivity—This is the capability of the system to anticipate user needs;
- Dependability—This is the capability of the system to provide services that are justifiably trustable.

In contrast, we have formulated a very complex hierarchy of internal requirements that contribute to the external ones. Figure 2 shows a possible structure with some dependencies between internal and external requirements. The detailed definition of each internal requirement is reported in the Appendix. Here we describe, as an example, the case of *Connectivity*, which is the capability of the system to provide network connectivity. According to its definition, *Connectivity* affects at least two external requirements, namely, *Pervasivity* and *Usability*.

3.2 Metrics and measures

In this section, we present some metrics and mechanisms for classifying attributes depending on their importance to the application. In particular, we can define an index (R), which reports the relevance of an attribute with respect to a specific application; so that, R_X is the relevance of the attribute (requirement) X.

R can vary in the range [0-1] and, as indicated in Fig. 3, attributes are classified as *Useless*, *Quite Interesting*, *Interesting*, *Very Interesting* and *Crucial*, as their importance increases.

Now, with respect to a specific pervasive application, based on a qualitative reasoning, we could calculate, for an example, that the attribute *Location-Awareness* is a very interesting characteristic. This means that its index, within our framework, should obtain a value equal to or greater than 0.6; this is a target for the system under construction.

From another point of view, the designer needs metrics to evaluate such an index in the real system. For the case of *Location-Awareness*, we define

$$R_{\text{Location}-\text{Awareness}} = \frac{\text{PSL}}{\text{PS}}$$

where PS is the entire Physical Space of interest for the application [m^2], and PSL is the amount of Physical Space in which the system is able to locate mobile entities of interest. In other words, $R_{\text{Location-Awareness}}$ corresponds to the coverage of positioning systems within the environment; that is, in order to get a value of 0.6 for this index, the developers must equip the environment with positioning systems that cover at least 60% of the physical space of the environment itself. Obviously, this requirement should be integrated with a specification about the accuracy, reliability and number of location systems and techniques. As an example, by considering the case of RFID-based positioning systems we have that in a large room an active RFID reader can cover the entire room (i.e., $R_{\text{Location-Awareness}}=1$), whereas a passive RFID reader cannot (i.e., $R_{\text{Location-Awareness}}<1$). However, the possibility of locating a user with a finer grain resolution may also be relevant. In fact, it could be necessary to distinguish the case when the user is in front of a wall monitor within the room from other

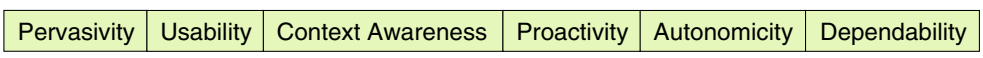

Fig. 1 External requirements for a generic pervasive application