A proposal of an architecture for a Smart Parking based on intelligent agents and embedded systems

Lucas Fernando Souza de Castro
Universidade Tecnológica Federal do Paraná
PPGCC - Programa de Pós-Graduação em Ciência da Computação
Av. Monteiro Lobato - km4
Ponta Grossa, Brazil
lcastro@alunos.utfpr.edu.br

Gleifer Vaz Alves
Universidade Tecnológica Federal do Paraná
PPGCC - Programa de Pós-Graduação em Ciência da Computação
Av. Monteiro Lobato - km4
Ponta Grossa, Brazil
gleifer@utfpr.edu.br

André Pinz Borges
Universidade Tecnológica Federal do Paraná
PPGCC - Programa de Pós-Graduação em Ciência da Computação
Av. Monteiro Lobato - km4
Ponta Grossa, Brazil
apborges@utfpr.edu.br

ABSTRACT

A smart city presents interactions between human beings and technological objects. Moreover, it is an environment where technology helps the human life, becoming it easier and practical. The smart city has different branches, such as: governance, people living, economy, mobility, and traffic. The traffic in a big city is one of the most stressful and costly thing to manage and organize, where specifically the management of parking lots is a hard task indeed. This paper aims to propose an architecture for a smart parking, which is an intelligent parking lot that uses technology to help drivers get a parking spot. The proposed architecture will be based on a multiagent system through JaCaMo Framework and embedded systems using Arduino and Raspberry Pi.

Keywords

smart parking; multiagent system; jacamo; embedded system

1. INTRODUCTION

It is well known that drivers spend a considerable time searching for a parking spot. For example, in New York City, 40% of the time that a driver spends while drives the car is to look a spot [10]. Moreover, in cities, there are many private parking lots where a driver can pre-book a spot even before of get off home. Most of the parking lots have an electronic system to manage the spots and their price. Although there are computer solutions to get a parking spot or check if it is available [4] [12], as far as we know, it is uncommon parking systems that have a smart system used to manage and allocate the parking spots in an autonomous way.

Some solutions to smart parking are based on multiagent system (MAS), which is a system composed of agents able to use and share resources in a dynamic environment [14]. In the smart parking there is a dynamic and complex environment due to many drivers and cars running at same time and even competing for the same resource, a spot. Therefore, the multiagent system could be applied to solve the smart parking matter. Actually, there are some solutions to smart parking based on a multiagent system, for instance Naples - Italy has been used as a scenario to apply a MAS able to provide a negotiation among drivers to get a parking spot [5]. Finally, a multiagent system is known for providing such social interaction among the agents, hence, in the smart parking context the drivers could compete or share a resource [6].

The present work aims to propose an architecture designed to smart parking based on a multiagent system developed through the JaCaMo framework and embedded systems (Arduino and Raspberry Pi). This architecture will be divided into three stages: `mobile, core, and embedded`. The `core` part was previous developed as a MAS called MAPS (**M**ulti**A**gent **P**arking **S**ystem), able to receive a request from driver agents and allocate a spot in a private parking lot. However, the MAPS was not applied in a real parking lot. Thus, this paper proposes the architecture that will use the MAPS and extend it to become applicable and workable in a real parking lot.

2. MAPS PROJECT

The MAPS Project was created to allocate parking spots for drivers in an autonomous way. The system has been developed using the JaCaMo Framework, which it is a tool to develop multiagent systems [1]. The framework is composed by Jason (Agent programming language), Cartago (Framework to develop artifacts for the multiagent system) and Moise (Tool to create a social organization).

The process of allocating a spot for a driver is based on a trust degree, which is a value responsible for showing the agent driver commitment before the smart parking. The figure 1 shows how the trust degree is used to decide which driver shall obtain the requested spot. The winner driver received the spot because it has a greater trust degree ($trust = 870$) than the another driver ($trust = 290$). The current version of the trust degree is calculated based on how many times the driver uses the parking lot. Each

WPCCG '16 Setembro 28, 2016, Ponta Grossa, Paraná, Brasil

time the driver uses the parking lot its trust degree increases in 20 points.

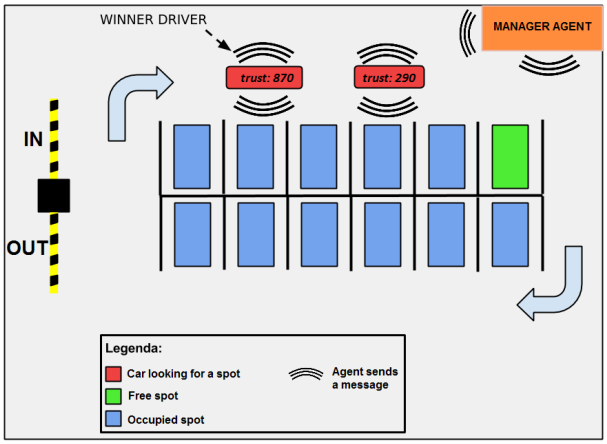

Figure 1: MAPS General representation

The trust degree value is stored by the manager agent. Every single driver agent has its own trust degree value, moreover, there is a database system created to store this value and update it whereas the driver agent uses the smart parking [7] [3]. Although the MAPS Project has been developed through JaCaMo framework, we use just the JaCa (Jason + Cartago) technology so far. Therefore, one of the possible extensions for the MAPS project is the implementation of social organization through Moise.

3. THE ARCHICTETURE

In order to develop the architecture, we shall face a complex situation, which different systems and technologies must work together appropriately. The figure 2 shows the general view of the architecture with its three main parts.

The architecture could be splitted into three stages: `mobile, core and embedded`. The `mobile` will be able to receive a request from the driver through an Android app and send to the `core`. The `core`, like the work suggests, is the kernel of the system, which is capable of managing the requests coming from the drivers and giving to them a parking spot. Finally, the `embedded` will be able to control the spots in the smart parking and inform the `core` how is going on the environment at the parking lot. The `embedded` is composed by a Raspberry Pi and an Arduino. The Raspberry Pi will be responsible for managing the Arduino boards, which will have sensors running. These sensors are going to check the presence of a driver in a spot and control the parking lot.

3.1 Architecture and its connections

Besides the architecture is composed of the stages (`mobile, core and embedded`), there are connections between each part of the system. Every connection will be based on the Java technology because it is done and ready to use. The connection among the `mobile, core` and `embedded` will be created using TCP/IP Java sockets. Further, the connection inside of the `embedded` will be done by the ARGO platform, which is a tool used to communicate the Jason language with the Arduino and Raspberry pi boards through a serial communication [11].

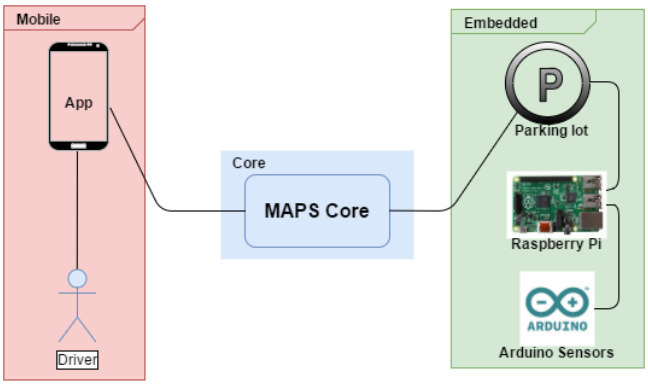

Figure 2: Proposed Architecture - MAPS

The connection using TCP/IP Java Sockets will be used to communicate among the stages because each state is separate of the another. The reason of choose Java socket is because the framework JaCaMo is based on Java language, thus it will be easier to use the Java to implement the connections among the stages.

3.2 Architecture and its stages

Implementing this architecture will be divided into sub-goals because they are independent to implement and each one has a different impact on the architecture. The first sub-goal is to extend the current MAPS Core developed using JaCaMo Framework to a new version which supports the requests from an Android app. This feature will be develop using the JaCa-Android [13]. The JaCa-Android tool is a project that provides the integration between Jason+Cartago and the Android platform [9]. The Android app will be able to provide to the driver the ability to request a parking spot to the MAPS Core. In Brazil, there are some new cars which support the usage of Android auto on streets. Moreover, it is possible to extend the JaCa-Android to develop an app to the Android auto platform [8].

As far as about the `embedded`, the second sub-goal is create an integration between the agents (in Jason) and Artifacts (in Cartago) with embedded platforms, like Raspberry pi and Arduino.

As explained before and showed in figure 1, allocating a parking spot depends on of trust degree value, a greater value will receive a parking spot faster than a lower value. The current way to calculate the trust degree is increment the trust degree value every single time the driver uses the system independently how long it stayed in the parking lot. Then, the third sub-goal is to develop a new way to calculate the driver trust degree and reward the driver to reinforce the use of this smart parking. This new trust degree will be showed in advance:

```
trustDegree(D) = trustDegree(D) + a + t + u
```

- **attendance (a)**: Value that shows how the driver is attending the parking lot. This value increases, according to with the frequency of the driver uses the parking;
- **time of usage (t)**: This value shows how long the driver spent in the parking lot;

- **usage (u)**: It shows the behavior of the driver inside of the parking lot. There will be social rules developed in Moise that govern the parking lot. For example, the driver will receive a specific parking spot and he cannot park in another parking spot. Otherwise, the drivel will be sanctioned.

4. FINAL REMARKS

As expected results, in Brazil, there are many cities waiting for smart parking technologies due to the growing of the urban traffic. The MAPS Project tries to bring some light to this issue offering to drivers the ability of a parking spot near where they are going at with a good price and availability. Moreover, there are rewards to the drivers who use the MAPS, they will gain a higher trust degree value. This value impacts the spot allocation time and consequently they could receive a spot faster. Finally, the smart parking is a part of a context called smart city, where there are many things to optimize and the parking lot is one and most important of them [2] [10]. Moreover, we hope this proposed architecture for the MAPS will offer a whole solution for a smart parking to manage the parking spots in different layers (`mobile, core` and `embedded`).

The future work is improving the proposed architecture to become more functional and effective. Therefore, to achieve that is important to decentralize the management of the system. The management of the smart parking is done by a central agent called manager. The manager is a JaCaMo agent. In the current version of MAPS, the manager works in MAPS core, however, to this new version of MAPS we intend to decentralize the manager agent to have many manager agents running at same time. There will be a hierarchy of the manager agents. In the MAPS Core, on the one side there will be managers able to receive a request for a spot and pick a parking lot for this spot, on the other side there will be another manager agent in every parking to manage the parking spots. Thereafter, it is going to be possible to support the negotiation of parking spots among driver agents. Hence, the risk of a fail will decrease whereas there will be more than one manager agent. The current version of MAPS could fail if the manager fails.

Finally, it is important to extend the MAPS Core to support more than one parking lot (scalability). This extension will be done through the multi-workspace concept in the Cartago. This main goal of this feature is provide to the driver multiple choices to park his car and the ability to the managers control multiple parking lots in order to approximate a real scenario.

5. ACKNOWLEDGMENTS

Thanks for the support of CAPES by means of the masters scholarship.

6. REFERENCES

[1] O. Boissier, R. H. Bordini, J. F. Hübner, A. Ricci, and A. Santi. Multi-agent oriented programming with jacamo. *Sci. Comput. Program.*, 78(6):747–761, June 2013.

[2] A. Caragliu, C. D. Bo, and P. Nijkamp. Smart cities in europe. *Journal of Urban Technology*, 18(2):65–82, 2011.

[3] L. F. S. Castro, G. V. Alves, and A. P. Borges. Utilização de grau de confiança entre agentes para alocação de vagas em um Smart Parking. In *Anais do X Workshop-Escola de Sistemas de Agentes, seus Ambientes e Aplicações – X WESAAC*, 2016.

[4] C. Di Napoli, D. Di Nocera, and S. Rossi. Negotiating parking spaces in smart cities. In *Proceeding of the 8th International Workshop on Agents in Traffic and Transportation, in conjunction with AAMAS*, 2014.

[5] C. Di Napoli, D. Di Nocera, and S. Rossi. Using negotiation for parking selection in smart cities. In Y. Demazeau, F. Zambonelli, J. Corchado, and J. Bajo, editors, *Advances in Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection*, volume 8473 of *Lecture Notes in Computer Science*, pages 331–334. Springer International Publishing, 2014.

[6] D. Di Nocera, C. Di Napoli, and S. Rossi. A Social-Aware Smart Parking Application. In *Proceedings of the 15th Workshop "From Objects to Agents"*, volume 1269, 2014.

[7] F. Ducheiko, L. F. S. Castro, and G. V. Alves. Implementação de recursos em um smart parking baseado em sistemas multiagente. In *Anais do X Workshop-Escola de Sistemas de Agentes, seus Ambientes e Aplicações – X WESAAC*, 2016.

[8] Google and A. Auto. Android auto. Available on <https://www.android.com/intl/eng-us/auto>, 2016.

[9] JACAMO. The JaCaMo approach. Available on <http://jacamo.sourceforge. net/?page_id=40>, 2011.

[10] A. Koster, F. Koch, and A. L. Bazzan. Incentivising crowdsourced parking solutions. In J. Nin and D. Villatoro, editors, *Citizen in Sensor Networks*, volume 8313 of *Lecture Notes in Computer Science*, pages 36–43. Springer International Publishing, 2014.

[11] N. M. Lazarin and C. E. Pantoja. A robotic-agent platform for embedding software agents using raspberry pi and arduino boards. In *9th Software Agents, Environments and Applications School (WESAAC)*, 2015.

[12] Parkingedge. Best parking. Available on <http://www.bestparking.com>, 2013.

[13] A. Santi. JaCa-Android. available on <http://jaca-android.sourceforge.net, 2011.

[14] M. Wooldridge. *An Introduction to MultiAgent Systems*. Wiley Publishing, 2nd edition, 2009.