Implementação de Recursos em um Smart Parking baseado em Sistemas Multiagente

Felipe Felix Ducheiko[1], **Lucas Fernando Souza de Castro**[1], **Gleifer Vaz Alves**[1]

[1]Departamento Acadêmico de Informática
Universidade Tecnológica Federal do Paraná (UTFPR)
Caixa Postal 84.016 – 210 – Ponta Grossa – PR – Brasil

felipeducheiko@alunos.utfpr.edu.br, l.castropg@gmail.com, gleifer@utfpr.edu.br

Abstract. *One of the major problems faced every day by the population of the cities is to find a free parking spot. Usually, in times great traffic flow to find a parking spot, a driver will waste time or will drive a long way until find a spot. Therefore, wasting fuel and generating traffic jams. The main purpose of MAPS project (Multiagent Parking System) is develop a multiagent system to allocate parking spots. Specifically, this article presents the implementation of additional resources to the MAPS project, so as to facilitate the expansion and implementation of the project.*

Resumo. *Um dos grandes problemas enfrentado todos os dias pelos habitantes das cidades é o de encontrar uma vaga de estacionamento livre. Normalmente em horários de grande fluxo é gasto uma considerável parcela de tempo, ou são percorridas longas distâncias até que seja encontrada uma vaga, o que desperdiça combustível e gera engarrafamentos. Pensando nisto o projeto MAPS (MultiAgent Parking System) é idealizado com objetivo de desenvolver um sistema Multiagente para alocação de vagas de estacionamento. Especificamente, este artigo apresenta a implementação de recursos adicionais ao projeto MAPS, para assim viabilizar a expansão e aplicação do mesmo.*

1. Introdução

O desenvolvimento da sociedade irrompe em muitos benefícios, mas consequentemente em alguns malefícios. Novas tecnologias são utilizadas para aprimorar e aumentar os benefícios já trazidos por tecnologias anteriores ou solucionar novos problemas. O conceito de Cidades Inteligentes refere-se ao uso de tecnologias da informação na tentativa de solucionar problemas das cidades a fim de melhorar a qualidade de vida da população.

Segundo [Batty et al. 2012] o conceito de Cidade Inteligente surgiu durante a última década com a fusão de várias ideias, tendo o intuito de melhorar a eficiência e a competitividade das cidades, criando novas maneiras para solucionar problemas. A essência do conceito é integrar as tecnologias que até agora têm sido desenvolvidas separadamente umas das outras. Mas que tem ligações claras em seu funcionamento e podem ser desenvolvidas de forma integrada.

Cidades possuem inúmeros desafios a serem resolvidos, dentre eles destacam-se os de mobilidade urbana. Segundo [Koster et al. 2014] cerca de 40% do tráfego em Nova York é gerado por carros à procura de vagas de estacionamento, o que ocasiona um agravamento dos congestionamentos e por conseguinte aumenta a emissão de poluentes.

Quando se percebe que a grande demanda de vagas de estacionamentos não está sendo satisfeita, normalmente provém a noção de que a solução é um aumento quantitativo do número de vagas. Porém, nem sempre essa é a solução mais sensata, pois a utilização das mesmas vagas de modo mais inteligente pode solucionar ou amenizar o problema. Um *Smart Parking* (estacionamento inteligente) pode ser composto por dispositivos de *hardware*, capazes de detectar o nível de ocupação dos estacionamentos, e *softwares* integrados para gerir a atribuição desses espaços. Normalmente tais sistemas são concebidos para auxiliar os motoristas na localização de vagas disponíveis [Nocera et al. 2014].

Dentre os vários modelos computacionais que podem ser usados para a implementação de um *Smart Parking* destacam-se os Sistemas Multiagentes (SMAs). Em [Wooldridge 2009] SMAs são definidos como sendo sistemas compostos de vários elementos computacionais que realizam interações, sendo tais elementos conhecidos como agentes. Esses sistemas possuem duas características importantes: primeiramente são, ao menos em certa medida, capazes de ações autônomas e em segundo lugar têm a capacidade de interagir uns com os outros pela interação análoga às interações sociais humanas. Além disso, os agentes então envolvidos em um ambiente onde eles podem ter uma organização, comunicação entre outros aspectos.

Tendo justamente o objetivo de aplicar métodos e técnicas de SMA, na criação de uma solução para alocação de vagas e gerenciamento de um *Smart Parking*, foi concebido o projeto MAPS *(MultiAgent Parking System)*. Para o desenvolvimento deste projeto está sendo utilizado o framework JaCaMo [JACAMO 2011].

Especificamente, o trabalho aqui apresentado tem como objetivo principal desenvolver dois recursos adicionais para que sejam incorporados ao projeto MAPS: a implantação de uma interface gráfica e a persistência de dados. Tais recursos visam facilitar a utilização e possibilitar a expansão do projeto.

O restante do artigo está organizado da seguinte maneira: na Seção 2 descreve-se o projeto MAPS. Na Seção 3 são apresentados detalhes sobre a implementação de recursos adicionais. Na Seção 4 encontram-se as considerações finais.

2. Projeto MAPS

O projeto MAPS *(MultiAgent Parking System)* é desenvolvido no GPAS (Grupo de Pesquisa em Agentes de *Software* - UTFPR - PG) com a meta principal de elaborar soluções para estacionamentos inteligentes. Os primeiros resultados do MAPS são apresentados no trabalho de [Castro 2015], onde foi implementado um SMA por meio do framework JaCaMo para alocação de vagas em estacionamentos[1].

Para implementar o sistema foram criados dois tipos de agentes (implementados no Jason) : os agentes *drivers* que interagem e utilizam o sistema multiagente e o agente *manager* que é responsável por informar, alocar e gerenciar as vagas do estacionamento.

O sistema também é composto por dois artefatos (implementados no Cartago) : o artefato *control* que é responsável pelo gerenciamento dos motoristas na fila e também o artefato *gate* que é responsável pela abertura e fechamento da cancela.

A alocação de vagas é realizada conforme o grau de confiança (*degree of trust*,

[1]Repositório GitHub: github.com/MAPS-UTFPR/MAPS

ou apenas *trust*) de cada motorista. Tal conceito é corroborado em [Huynh et al. 2006] quando afirmam que a confiança e a reputação são temas centrais para a interação efetiva em um SMA aberto em que os agentes entram e saem do sistema. Em [Gonçalves and Alves 2015] é discutida uma proposta para o uso e cálculo do *trust* em um *Smart Parking*. Contudo, a atual versão do MAPS ainda não implementa o conceito de grau de confiança, pois não há armazenamento de valores de confiança. Após, a Seção 3.2 irá discutir tal questão.

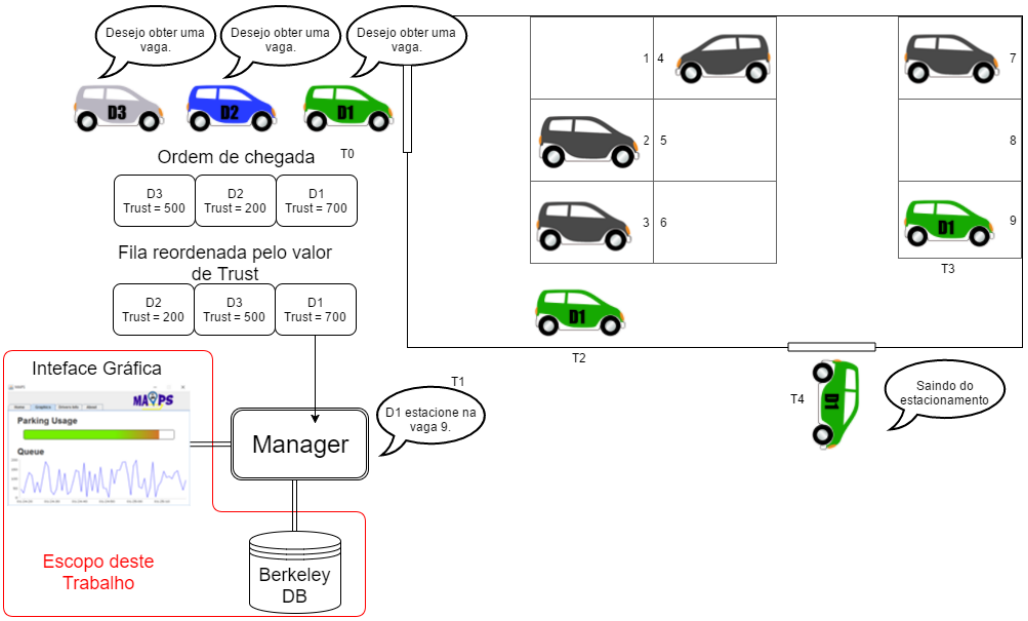

Figura 1. Diagrama Projeto MAPS - Fonte: Autoria Própria

Na Figura 1 é apresentado um diagrama que ilustra o funcionamento geral do Projeto MAPS. Neste diagrama é possível identificar os agentes *drivers* que chegam no estacionamento e requisitam vagas. O agente *Manager* gerencia a fila de *drivers*. Os instantes de tempos (t0 até t4) ilustram as diferentes ações de um agente no SMA.

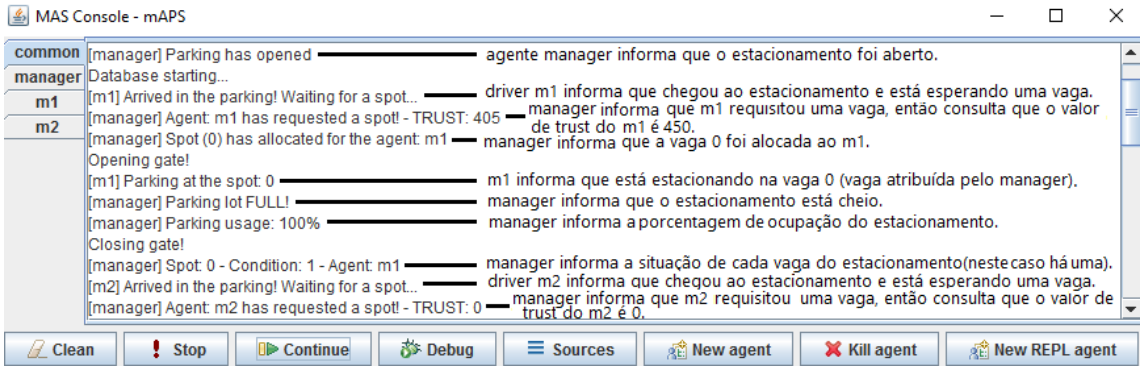

Figura 2. Console JaCaMo

Com a finalidade de acompanhar a execução do SMA o JaCaMo possui um console simplificado. Na Figura 2 é apresentado o início da execução do MAPS através do console e uma breve explicação das interações que ocorrem entre os agentes.

3. Implementação de Recursos

Esta seção aborda detalhes da implementação dos recursos adicionais. Na Subseção 3.1 é apresentada a Interface, ao passo que na Subseção 3.2 a implementação do Banco de Dados é discutida.

3.1. Interface gráfica

A implementação desta primeira interface gráfica do MAPS é voltada para os responsáveis pelo controle e gestão do *smart parking*, a fim de que possam ter acesso rápido a informações cruciais para otimizar o seu desempenho. A implementação está em fase de desenvolvimento, conforme é possível visualizar na Figura 3.

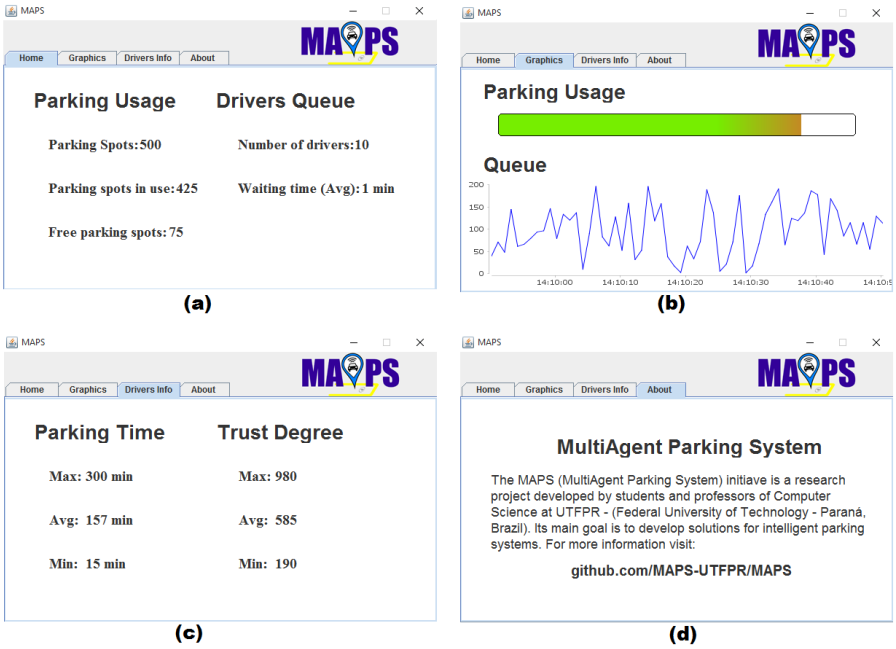

Figura 3. Protótipos das telas

Para desenvolver o protótipo foram definidos os seguintes requisitos: (i) design intuitivo e descomplicado; (ii) simplicidade, assim evitando uma sobrecarga cognitiva do usuário; (iii) desenvolvimento em Java, para facilitar a integração com o JaCaMo; e (iv) uma interface facilmente extensível, devido as futuras alterações nas versões do sistema.

Na tela inicial da interface gráfica, que pode ser visualizada na Figura 3 (a), o usuário já possui informações sobre o nível de ocupação do *smart parking* (número total de vagas, o número de vagas livres e em uso) e sobre a situação da fila de espera (número de motoristas na fila e o tempo médio de espera).

Na aba *graphics*, que pode ser visualizada na Figura 3 (b), o usuário possui informações sobre o nível de ocupação do *smart parking*, através de um gráfico de barra e sobre a situação da fila de espera, através de um gráfico de linha que relaciona o horário e quantidade de motoristas na fila. Estas informações são apresentadas através de gráficos simples para que sejam facilmente verificadas.

Na Figura 3 (c) tem-se a aba *drivers info* que traz os valores máximo, mínimo e médio do tempo de permanência *(Parking time)* e do *trust* dos motoristas que frequenta-

ram o *smart parking*. Essas informações são úteis para melhorias futuras, por exemplo se o valor médio do *trust* estiver alto pode ser um indício de que novos motoristas teriam dificuldade de conseguir vagas. Logo, essas informações podem ajudar o agente *Manager* a calibrar os valores para assegurar o bom funcionamento do SMA. Observando a Figura 3 (d) é possível visualizar a aba *About*, que trás informações básicas sobre o projeto.

Como já demonstrado, na Figura 2, o console do JaCaMo exibe o passo a passo da execução do MAPS, o que não é suficiente para o objetivo aqui abordado, visto que exibe apenas informações básicas do funcionamento do sistema. Contrastando, a interface exibe informações complementares como gráficos e valores máximos, médios e mínimos.

3.2. Persistência de Dados

A implementação inicial do MAPS possuía os valores de confiança fixos, ou seja, os agentes *drivers* sempre tinham o mesmo valor de confiança perante o *manager*. Com a implementação da persistência dos dados é possível que os valores sejam salvos para futuras utilizações de um mesmo agente *driver*. A partir da implementação da persistência dos valores de *trust*, é possível gerenciar estes valores baseado em um histórico de utilização. Na versão atual do projeto, o grau de confiança é obtido de forma simples e direta. Basta incrementar 20 unidades ao valor de confiança, cada vez que um dado *driver* utilizar o *smart parking*. O valor de 20 unidades é usado considerando que um *driver* poderia utilizar cinco vezes em uma semana o estacionamento e obter a pontuação igual a 100 unidades, o qual representa 10% da pontuação máxima (1.000 unidades).

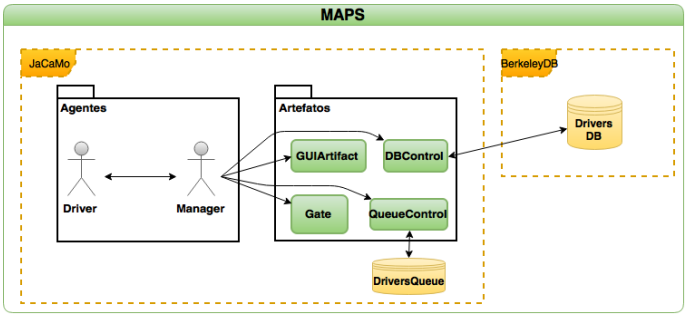

Figura 4. Projeto MAPS: perspectiva DB

O gerenciamento da persistência dos dados dá-se por meio de um artefato em Cartago chamado *DBControl*, o qual é responsável pelo controle dos valores de confiança e informá-los para o *manager*. O banco de dados utilizado para o armazenamento dos valores de confiança e dos dados dos *drivers* segue o modelo NO-SQL (*Not Only SQL*) do tipo chave-valor, devido a sua alta performance, escalabilidade e simplicidade [Moniruzzaman and Hossain 2013]. Com o objetivo de manter as características do modelo chave-valor na implementação do banco de dados no MAPS, foi utilizado o banco BerkeleyDB da Oracle. Porém a versão utilizada foi a *Java Edition* pra facilitar a integração com o Cartago, visto que o Cartago utiliza a linguagem Java.

A Figura 4 ilustra os componentes Jason, Cartago e do Banco de Dados implementados no projeto MAPS. Em especial, os novos recursos descritos neste trabalho, a saber: artefatos do Cartago (*DBControl* e *GUIArtifact*), bem como a adição do BerkeleyDB como banco de dados para armazenar os dados dos agentes *drivers*.

4. Considerações Finais

Este trabalho apresenta a implementação de uma interface gráfica voltada ao controle e gestão do MAPS e a implementação da persistência de dados. A etapa de implementação desta primeira interface gráfica é uma evolução para o projeto MAPS, pois pode auxiliar o agente *manager* a tomar decisões. Por exemplo, avaliar se os *drivers* estão permanecendo pouco tempo no estacionamento, ou seja, se há grande rotatividade de vagas. Se isso ocorre, o *manager* pode reorganizar a dinâmica das vagas para facilitar o estacionamento. Além da interface gráfica, há a extensão da persistência dos dados, a qual proporciona que o grau de confiança dos agentes *drivers* seja utilizado e incrementado de acordo a sua utilização.

O emprego do *framework* JaCaMo facilitou a implantação dos recursos aqui propostos, visto que no desenvolvimento do artefatos em Cartago através da linguagem Java proporciona um alto nível de abstração para o desenvolvimento dos recursos.

O projeto MAPS continua em execução, e pretende-se desenvolver os seguintes trabalhos: (i) desenvolver um aplicativo móvel onde os agentes *drivers* poderão requisitar as vagas; (ii) embarcar o sistema utilizando o Arduino e Raspberry Pi [Lazarin and Pantoja 2015]; (iii) criar outras formas para calcular o grau de confiança; e (iv) utilização do simulador de trânsito Sumo.

Referências

Batty, M., Axhausen, K., Fosca, G., Pozdnoukhov, A., Bazzani, A., Wachowicz, M., Ouzounis, G., and Portugali, Y. (2012). Smart Cities of the Future.

Gonçalves, W. R. C. and Alves, G. V. (2015). Smart Parking: mecanismo de leilão de vagas de estacionamento usando reputação entre agentes. 9º Workshop-Escola de Sistemas de Agentes, seus Ambientes e aplicações. Niterói - RJ. jun. 2015.

Huynh, T. D., Jennings, N. R., and Shadbolt, N. R. (2006). An integrated trust and reputation model for open multi-agent systems.

JACAMO (2011). The JaCaMo approach. Disponível em <http://jacamo.sourceforge.net/?page_id=40>.

Koster, A., Koch, F., and Bazzan, A. L. (2014). Incentivising Crowdsourced Parking Solutions.

Lazarin, N. M. and Pantoja, C. E. (2015). A Robotic-agent Platform For Embedding Software Agents using Raspberry Pi and Arduino Boards. 9º Workshop-Escola de Sistemas de Agentes, seus Ambientes e aplicações. Niterói - RJ. jun. 2015.

Moniruzzaman, A. and Hossain, S. A. (2013). Nosql database: New era of databases for big data analytics-classification, characteristics and comparison. *arXiv preprint arXiv:1307.0191*.

Nocera, D. D., Napoli, C. D., and Rossi, S. (2014). A Social-Aware Smart Parking Application.

Wooldridge, M. (2009). *An Introduction to MultiAgent Systems*. J. Wiley, New York, 2nd edition.