Pesquisa Atual e Futura no Laboratório de Sistemas de Computação da UFSM: Soluções para Cidades Inteligentes

Marcia Pasin[1], Liza Lunardi Lemos[2], Rafael Fao de Moura[1], Ricardo Silveira Rodrigues[1], Emmanuel Katende Dinanga[3], Eric Thomas Zancanaro[1], Marina Laisa Mota da Silva[1], Felipe Silvano Perini[4], Cleandro Flores De Gasperi[1], Leonardo de Abreu Schmidt[1], Ezequiel Rodrigues Ribeiro[1]

[1] Universidade Federal de Santa Maria
Av. Roraima, 1.000 - Cidade Universitária, Santa Maria - RS, 97105-900
[2] Pós-Graduação em Ciência da Computação
Universidade Federal do Rio Grande do Sul
Avenida Bento Gonçalves, 9500 - Agronomia, Porto Alegre - RS, 91509-900
[3] Pós-Graduação em Engenharia de Automação e Sistemas
Universidade Federal de Santa Catarina
Bairro Trindade, Caixa Postal 476, Florianópolis - SC, 88040-900
[4] Pós-Graduação em Engenharia de Software
Instituto de Gestão e Tecnologia da Informação
Complexo Paisagem - Rua Roma, 26 - Santa Lucia, Belo Horizonte - MG, 30360-680

Resumo Este texto relata trabalhos desenvolvidos e em desenvolvimento no Laboratório de Sistemas de Computação (LSC) da Universidade Federal de Santa Maria (UFSM) para a implementação de soluções para o contexto de Cidades Inteligentes. Os projetos tem sido desenvolvidos desde 2010 pelos alunos dos cursos de bacharelado em Ciência da Computação e Sistemas de Informação, e de mestrado do Programa de Pós-Graduação em Informática da UFSM. Mais precisamente, neste texto são enfocados projetos nos eixos de sistemas de transportes inteligentes, infraestruturas para cidades inteligentes e automação residencial. Os eixos são explorados de duas formas: simulação computacional e prototipação. Os trabalhos que envolvem prototipação foram desenvolvidos na plataforma de *software* e *hardware* Arduino, e em plataformas de programação abertas. O ambiente de simulação usado foi o SUMO. A principal contribuição deste texto é mostrar que uma coleção de trabalhos pontuais pode colaborar no avanço do estado da arte.

Palavras-chave: simulação, prototipação, sistemas de transporte inteligentes.

1 Introdução

Este texto relata trabalhos desenvolvidos e em desenvolvimento por pesquisadores no Laboratório de Sistemas de Computação (LSC) da Universidade Federal

de Santa Maria (UFSM) para a implementação de soluções para o contexto de Cidades Inteligentes. Os projetos tem sido desenvolvidos desde 2010 pelos alunos dos cursos de bacharelado em Ciência da Computação e Sistemas de Informação, e de mestrado do Programa de Pós-Graduação em Informática da UFSM.

Mais precisamente, neste texto são enfocados projetos nos eixos de sistemas de transportes inteligentes, infraestruturas para cidades inteligentes e automação residencial. Os eixos são explorados de duas formas: simulação computacional e prototipação. Para a simulação de sistemas de transporte, foi usado o simulador de código aberto SUMO [1]. Os trabalhos que envolvem prototipação estão em em duas frentes: trabalhos desenvolvidos na plataforma de *software* e *hardware* Arduino[5] e trabalhos desenvolvidos em plataformas de programação abertas. Plataformas de programação abertas foram usadas para simular serviços de base gerenciais para infraestruturas computacionais, como serviços de gerenciamento autonômicos e um serviço para atualização de *software*. Um objetivo claro e que está presente em todos os trabalhos é o uso do suporte de plataformas de baixo custo e de sistemas de implementação aberta.

O conceito de *smart cities*, ou cidades inteligentes, emprega tecnologias de informação e comunicação (TICs) para melhorar a infraestrutura urbana, buscando tornar os centros urbanos mais eficientes e melhores de se viver. De forma análoga, o conceito de sistemas de transportes inteligentes também faz uso das TICs para melhorar a eficiência e a segurança de redes de transporte.

Sabe-se também que o conceito cidades inteligentes e de sistemas de transportes inteligentes dificilmente serão completamente implementados. Entretanto, estes conceitos direcionam e impulsionam as aplicações executam em plataformas inteligentes, o que abre uma infinidade de oportunidades para pesquisa e desenvolvimento de novas soluções.

2 Simulação no controle de interseções de trânsito

No futuro, com a implantação das tecnologias de redes veiculares, o trânsito irá se beneficiar da comunicação intra-veicular para o controle semafórico. Semáforos serão virtualizados, e não haverá mais a necessidade do uso de sinaleiras para o controle do fluxo de veículos e pedestres. Até lá, novos algoritmos devem ser projetados e avaliados.

2.1 Distribuição do fluxo do trânsito

Em Dinanga e Pasin [2], foi apresentada uma implementação de uma simulação no contexto de redes de transporte. Diferentes políticas para o gerenciamento de interseções foram avaliadas. A simulação conta com duas vias concorrentes: uma com sentido sul ao norte (s), e outra com sentido oeste ao leste (w), as duas se encontrando uma interseção, conforme mostrado na Figura 1. Os veículos que estão dentro do alcance de transmissão podem colaborar para o algoritmo de controle semafórico.

[5] https://www.arduino.cc

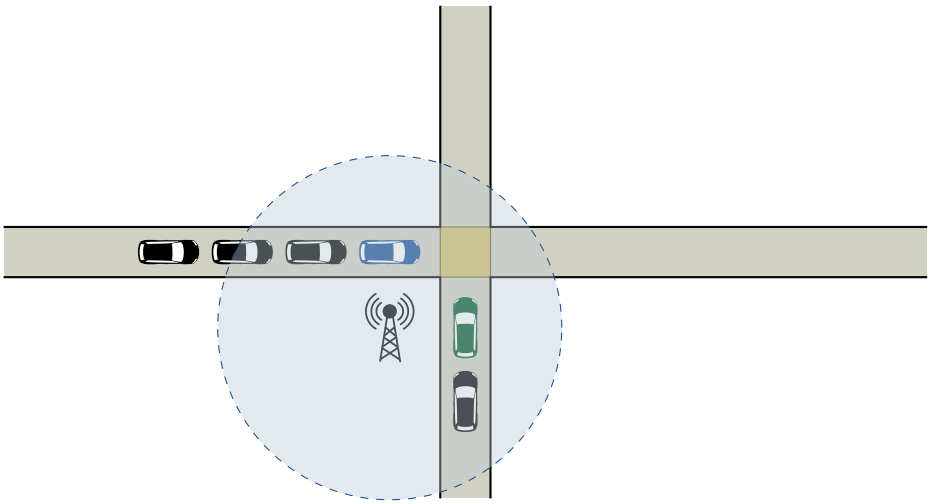

Figura 1: Cenário alvo com uma interseção e duas vias.

Foram criados três estados de fluxos do trânsitos: E1, E2, e E3, baseados na preferência da s sobre a w na passagem dos veículos por uma interseção, no período do acesso dos veículos nas duas vias, e na velocidade máxima das duas vias. Depois foram implementadas cinco políticas para controlar a passagem dos veículos de s e w pela interseção. Por fim, foram gerados 15 cenários decorrentes da aplicação das políticas implementadas aos estados de fluxos do trânsito definidos, classificados em três grupos de cenários de acordo com cada estado do trânsito.

Experimentos com estes três cenários foram executados para o controle da interseção com duas vias concorrentes, para avaliar a distribuição do fluxo. Os resultados de mostraram que quando o fluxo é idêntico nas duas vias (E1), a política da maior fila primeiro (LQF) é a mais adequada. Quando a frequência do fluxo da via preferencial é superior à via concorrente (E2), a política temporizada do semáforo mostrou-se a mais adequada. Por fim, quando a frequência do fluxo da via preferencial é inferior à via concorrente (E3), o algoritmo *zipper merge* (ZM) foi a mais adequado. O algoritmo ZM alterna entre os fluxos, permitindo a passagem de x veículos (isto é, um comboio) a cada *round*. Sempre que ocorre a troca do fluxo, é adicionada uma penalidade pois uma fila precisa parar e outra precisa iniciar o escoamento. Se o comboio é formado por apenas um carro, isto é, se $x = 1$, haverá sempre uma alternância de passagem de fluxo cada vez que um único carro passa a interseção.

Com os experimentos, concluiu-se que aplicar apenas uma política não adaptativa para controlar o trânsito urbano não é a solução mais adequada. Como o trânsito é dinâmico, políticas adaptáveis ao dado contexto podem ser mais interessantes.

2.2 Eficiência no escoamento do trânsito

Na sequencia, no trabalho de Fao de Moura *et al.* [3] foi comparada a eficiência no escoamento do trânsito (vazão) de algoritmos para o controle semafórico virtuais em redes veiculares. A eficiência implica na redução dos tempos de viagens e, possivelmente, auxilia na redução de congestionamentos. Os algoritmos foram implementados no simulador SUMO. Resultados demonstram que soluções adaptativas oferecem ganhos significativos.

Foram implementados três algoritmos que possibilitam operação em semáforos virtuais: ZM, fila mais velha primeiro (OQF) e a LQF. A métrica usada para avaliar os diferentes algoritmos foi a vazão.

O algoritmo OQF avalia o tempo de geração do veículo no topo das filas concorrentes w e s, uma para cada fluxo. Se o veículo no topo da fila w é saiu antes da garagem que o veículo no topo da fila s, a fila de w passa primeiro. De outra forma, a fila s passa primeiro. Finalmente, o algoritmo LQF avalia o tamanho das filas em w e s. Se o tamanho da fila w é maior que o tamanho da fila em s, a fila em w passa primeiro. De outra forma, a fila s passa primeiro.

Os três algoritmos foram avaliados levando em conta diferentes fluxos de trânsito. O algoritmo ZM foi avaliado usando diferentes tamanhos de comboios: $x = 1$, 10 e 20 veículos. Este algoritmo foi contrastado com os demais. O que se pode observar, é que algoritmos baseados em comboios, como ZM para $x = 20$ e LQF, apresentam a melhor vazão mesmo com diferentes volumes de tráfego. Quando $x < 10$, ocorrem muitas trocas nas filas, e o algoritmo ZM perde a eficiência em termos de escoamento. A vazão, portanto, é baixa. O Algoritmo OQF, voltado para a justiça, dado que a fila mais velha nunca espera, também apresentou resultados interessantes e similares ao LQF. Portanto, conforme o esperado, algoritmos com comboios grandes e, portanto, com menos trocas, são as melhores soluções para escoamento eficiente de veículos.

Finalmente, uma exploração mais ampla de algoritmos para o controle de uma interseção foi realizada em Pasin *et al.* [4]. Nesse trabalho, foram avaliados três algoritmos usando duas métricas opostas: justiça e vazão. Os algoritmos que avaliam cada demanda de cada veículo individualmente, considerando aspectos do fluxo, são tipicamente mais adequados para a justiça. Algoritmos que priorizam comboios são mais adequados para a vazão. Um novo algoritmo foi proposto, eficiente em termos de justiça e vazão.

2.3 Eficiência em termos de baixa emissão de gases poluentes

Com o aumento de veículos nas vias, há também outros problemas associados, como o efeito estufa causado pela emissão de gases poluentes provenientes dos veículos e a alta demanda por combustível. Esses gases tóxicos são prejudiciais para a saúde e o meio ambiente. Dessa forma, a otimização de tráfego se torna importante, não somente para a redeção do tempo de viagem dos veículos, como também para minimizar a emissão dos gases poluentes.

Em Liza e Pasin [5] são utilizados diferentes algoritmos para controlar as interseções em uma rede de transporte. Os autores, testam o comportamento

dos algoritmos: ZM, LQF e *First In, First Out* (FIFO). O objetivo do trabalho é investigar qual das abordagens minimiza a emissão de gases poluentes (CO_2), o consumo de combustível e a duração média da viagem, e aumenta a vazão de veículos nas interseções.

Para avaliação dos algoritmos, experimentos foram conduzidos no simulador SUMO utilizando uma rede de transporte com quatro interseções. A rede de transporte em questão possui uma via arterial e quatro vias coletoras, as quais cruzam a via arterial. Todas as interseções possuem um semáforo virtual, que adota uma política (ZM, LQF ou FIFO). Em cada execução, todos os semáforos virtuais executam a mesma política. Além disso, três níveis de fluxo de tráfego foram testados (alto, médio e baixo). Para cada nível foram alteradas as variáveis velocidade das vias e o período com que os veículos são inseridos na simulação. Os resultados obtidos pelos algoritmos foram comparados entre eles.

Para as métricas emissão de CO_2, consumo de combustível e tempo médio de viagem, quando menor o valor, melhor. Para a métrica vazão, quanto maior o valor, melhor é o desempenho do algoritmo. A Figura 2 apresenta os gráficos dos resultados obtidos pelos algoritmos para cada uma das métricas. O algoritmo orientado a comboios LQF é o que obtém os melhores resultados se comparado com os algoritmos ZM e FIFO, em todos os níveis de fluxo de tráfego. Portando, entre as políticas testadas, o algoritmo LQF é o que emite menos poluentes e aumenta a vazão na rede de transporte avaliada.

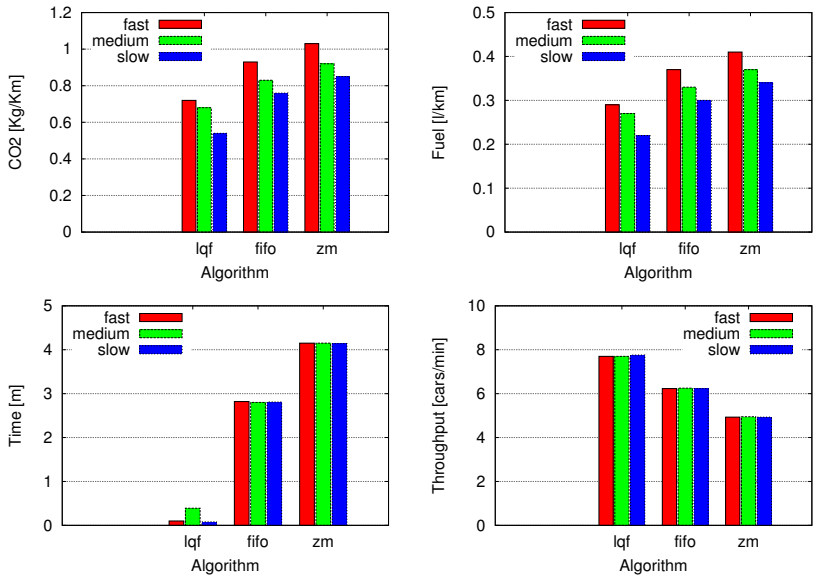

Figura 2: Gráficos dos resultados obtidos pelos algoritmos para diferentes métricas e fluxos de tráfego.

3 Prototipação: infraestruturas de suporte para cidades inteligentes

Uma vez implantado o conceito de cidades inteligentes e de sistemas de transportes inteligentes, sabe-que que estas aplicações executarão sobre infraestruturas também inteligentes. As infrastruturas terão demandas elásticas, onde o sistema está sujeito a picos de carga na chamada hora do *rush*, quando recebe muitas requisições de usuários, e momentos de ociosidade, por exemplo, durante a madrugada. Um sistema de informação ao usuário de transporte é uma aplicação típica de demanda elástica. Sabe-se também que este sistema precisa estar continuamente disponível e que desativá-lo para reparar falhas ou realizar atualização, é algo inaceitável. Os trabalhos que seguem contribuem nesta direção.

3.1 Infraestruturas para aplicações com demandas elásticas

Em Perini *et al.* [6] [7] foi proposto um sistema de auto-gerenciamento para uma infraestrutura computacional que suporta aplicações Web com demandas elásticas. O esquema desta arquitetura é apresentado na Figura 3.

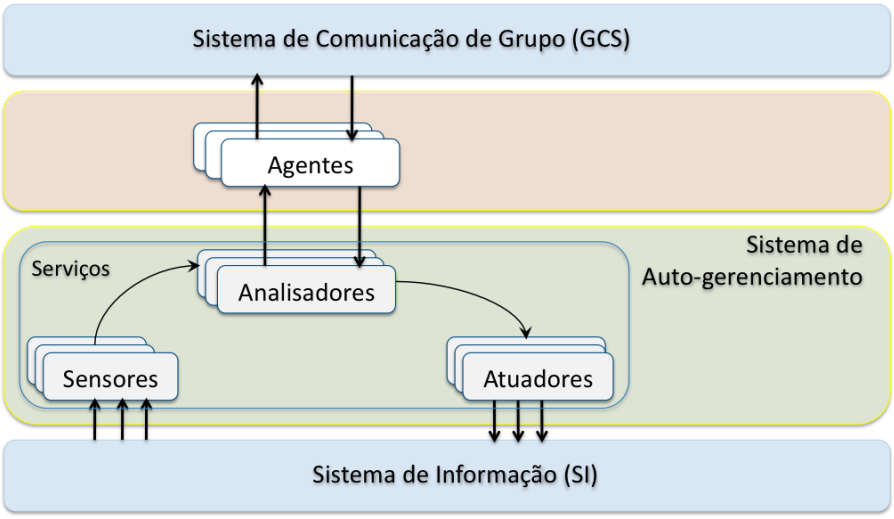

Figura 3: Arquitetura em múltiplas camadas para um sistema com auto-gerenciamento que suporta uma aplicação provedora de informação ao usuário de transportes.

O sistema de auto-gerenciamento é implementado como uma coleção de serviços descentralizados. Baseado na informação coletada por sensores embutidos na aplicação (Sistema de Informação ou SI) que executa em uma infraestrutura

computacional, os serviços de auto-gerenciamento desencadeiam adaptação dinâmica sempre que necessário. Por exemplo, o serviço de reconfiguração adapta a quantidade de servidores disponíveis na infraestrutura com a finalidade de manter a qualidade do serviço.

Esta proposta destaca as seguintes características: (*i*) suporte ao tratamento de decisões conflitantes, (*ii*) economia de recursos: em períodos de ociosidade, servidores excedentes são logicamente removidos da infraestrutura e podem ser alocados a outras aplicações ou podem operar em *standby*, e (*iii*) suporte estratégico para reconfiguração de servidores devido à demanda elástica e manutenção de configuração (número de servidores) em virtude de picos de carga momentâneos.

O suporte à tomada de decisão é realizado através de um sistema multiagente. As decisões não são somente tomadas com base em métricas atuais, mas com base em dados históricos. O suporte à reconfiguração é oferecido através de primitivas implementadas por um sistema de comunicação de grupo (SCG)[6].

3.2 Atualização dinâmica em infraestruturas computacionais

Apesar de sistemas que oferecem informação ao usuário não possuírem demanda de tempo real crítica, eles precisam estar continuamente disponíveis. Uma questão importante é que os sistemas providos por terceiros (bancos de dados, servidores de aplicação) estão em contínua atualização. Novas versões são lançadas no mercado a cada ano e é importante mantê-las atualizadas pois a cada nova versão, novas melhorias são adicionadas. O sistema de gerenciamento de banco de dados (SGBD) é parte fundamental do *software* corporativo e sua atualização dinâmica não é simples devido à complexidade inerente. Frequentemente, atualização de *software* é uma tarefa executada manualmente e está associada à indisponibilidade de serviço e uso de *hardware* adicional.

Em De Gasperi [8] foi proposto um passo na direção da atualização dinâmica de *software* (ADS) para SGBDs. Nesta solução, o modelo de componentes de *software* é usado como um alternativa de suporte para a ADS. O modelo de componentes apresenta um nível de abstração que favorece a troca de componentes pois oculta detalhes de implementação. Em uma infraestrutura de *software*, cada SGBD, e cada servidor de aplicação pode ser encapsulado em um componente e uma versão defasada pode ser substituída por uma nova versão daquele componente, se houver compatibilidade. A atualização ocorre de forma incremental, sem interromper o serviço. Para validação da proposta foi desenvolvido um protótipo utilizando o modelo de componentes Fractal[7]. Experimentos em ambiente controlado confirmam a viabilidade da solução se as versões forem compatíveis. Porém, destaca-se que o próprio modelo de componentes ainda é um gargalo.

[6] http://jgroups.org
[7] http://fractal.ow2.org/tutorials/adl

4 Prototipação em sistemas de transporte: localização de veículos

A plataforma Arduino oferece em uma excelente alternativa para a prototipação de pequenos projetos de computação embarcada. O LSC adquiriu em 2015 quatro carros-robôs Arduino, e diferentes tecnologias de comunicação, que são usados em experimentos no contexto de redes veiculares. Esta plataforma está sendo usada como base para a implementação de projetos no contexto de redes veiculares no LSC.

Silveira Rodrigues *et al.* [9] apresentaram um serviço de auto-localização para veículos aproveitando a infraestrutura de redes veiculares. Em um esquema baseado em trilateração que comunica sensores e veículo Arduino, um algortimo de auto-localização possibilita que o veículo se localize no ambiente.

A solução proposta possui dois tipos de componentes principais: sensores e unidade central de processamento. Os sensores captam informações sobre a localização de um veículo v e são implementados por SONARes (*Sound Navigation and Ranging*). A informação coletada pelo sonar consiste na distância entre o sonar e um veículo v. Através desta distância, a unidade central de processamento computa as coordenadas (x, y) de um dado veículo. As coordenadas são válidas apenas para o ambiente e que são conhecidas através pontos de referências, podendo ser um mapa do ambiente ou posições preestabelecidas.

Um sonar é um instrumento que detecta a presença de um objeto medindo o tempo de refração de uma onda sonora. Neste trabalho, um conjunto de sonares é apresentado como uma tecnologia para localização veículos. Esta solução, além de oferecer precisão significativa, representa uma opção de baixo custo em comparação ao GPS.

O sonar implementado neste projeto é composto por um sensor ultrassônico HC-SR04 e um servo motor 9g SG90, que são conectados à uma placa Arduino UNO. O sensor ultrassônico detecta e calcula a distância de obstáculos na sua frente. Ele emite uma onda sonora que é refletida quando colide com um obstáculo. A distância é calculada através da medição do intervalo de tempo que a onda sonora é emitida até o momento que ela retorna refletida.

Para que o sistema de localização calcule a posição do veículo, o sensor é acoplado ao servo motor. O servo motor rotaciona o sensor em seu próprio eixo, fazendo com que aumente a área de detecção e permitindo calcular a posição do veículo. Através da plataforma Arduino, é possível controlar precisamente qual o grau de rotação que o eixo do servo motor está rotacionado e, consequentemente, a direção para qual o sensor está apontando.

Além de controlar o sensor e o motor, o Arduino é responsável por juntar essas informações com a posição do sonar no ambiente, e enviá-las para a computador, que opera como central de controle. A cada rotação do servo motor, é enviada para a central uma mensagem contendo essas informações. De posse das informações coletadas por três sonares, a central infere a posição de um veículo por trilateração, estando ele em movimento ou parado. Experimentos foram realizados em um ambiente controlado e apresentaram erro de $\approx$ 8cm, entre a

posição obtida pelos sonares e aposição indicada na pista, para uma pista circular com 5,5m de comprimento. Dado que o carro tem tamanho de 20cm^2, este erro é aceitável. A Figura 4 mostra o esquema do cenário onde os experimentos foram realizados e foto real do cenário com a pista circular.

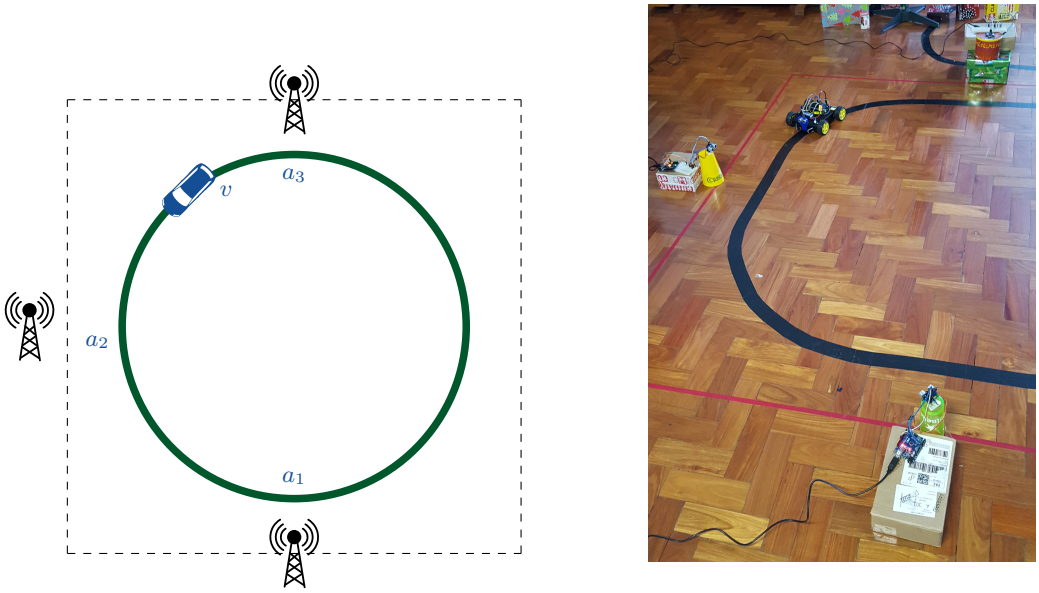

Figura 4: Esquema do cenário dos experimentos e foto real do cenário com a pista circular.

5 Prototipação em automação residencial

Tecnologias para automatização de ambientes e residências buscam prover controle automático e segurança enquanto diminuem o consumo de energia. Uma *smart house* consiste em uma casa com sensores inteligentes e atuadores operando em diferentes plataformas com objetivos diferentes. Em Mota da Silva [10] foi desenvolvida uma arquitetura de serviços para provimento de informações e gerenciamento de uma *smart house*.

Uma maquete foi construída para atuar como sistema protótipo. A maquete possui uma placa Arduíno UNO, que é responsável pelo controle de sensores e atuadores. O processamento dos dados é realizado em um computador convencional. A ênfase deste projeto é no sistema de iluminação e no sistema de alarme, que conta com sensores de presença, controlados através de uma interface Web. A interface *Web* foi desenvolvida com o uso do *framework* Django[8]. Através da

[8] https://www.djangoproject.com

interface, pode-se visualizar os dados coletados, atualizados frequentemente com o uso da tecnologia AJAX[9], em forma de gráficos e relatórios por período informado, informando quanto tempo as luzes ficaram ligadas, desligadas, quantas vezes foram acesas e se o alarme foi ativado ou não. O usuário também pode ligar ou desligar as luzes manualmente. Também foram desenvolvidos um módulo para controlar o portão da garagem e outro módulo de alarme contra intrusão.

Em duas peças da casa foram instaladas lâmpadas LED e sensores de presença. Foram necessários o uso de resistores para permitir o acionamento das lâmpadas de LED na casa. A aparência da maquete com a placa Arduino é apresentada na Figura 5.

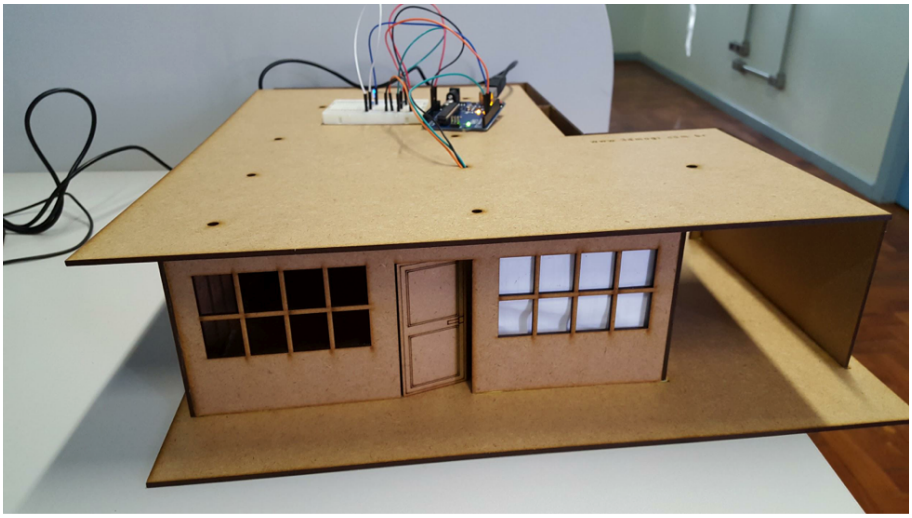

Figura 5: Maquete usada no projeto.

Os próximos passos deste trabalho incluem (*i*) a troca da placa Arduino UNO por uma placa mais robusta, para permitir a adição do módulo Wi-Fi para estabelecer a comunicação entre a casa e um computador remoto, (*ii*) melhoramento da interface gráfica, com a adição de novas funções e (*iii*) melhoramento do módulo de relatórios.

6 Trabalhos em desenvolvimento e pesquisa futura

Finalmente, esta seção apresenta trabalhos que estão sendo desenvolvidos no LSC. Destaca-se a avaliação do serviço intra-campus da UFSM e a avaliação da comunicação veicular utilizando diferentes tecnologias.

[9] http://glm-ajax.sourceforge.net

6.1 Avaliação do serviço intra-campus da UFSM

A UFSM conta com um vasto *campus* e um serviço gratuito de transporte interno para facilitar a circulação de alunos, professores e funcionários. Basicamente, neste serviço, existe um ônibus circular que parte a cada 30 minutos do terminal Centro Comercial do Campus, percorrendo um percurso pré-estabelecido. O serviço opera das 7h30min até as 22 horas, de segunda a sexta.

A tabela detalhada com os horários (de partida no ponto inicial e de chegada no ponto final), está disponível no *site* da UFSM. A informação sobre os horários oferecida no *site* é estática e não reflete o fato de que o trânsito está sujeito a transtornos. Por exemplo, em horários de pico, atrasos são recorrentes.

Outro problema é a ausência de informação sobre paradas deste ônibus na UFSM. Com exceção da parada inicial, não se tem informações sobre a localização das demais paradas e sobre os horários que o ônibus serve tais paradas. Também não se sabe qual horário o ônibus chega em qual parada.

O trabalho em andamento, desenvolvido pelo aluno Ezequiel Rodrigues Ribeiro propõe uma avaliação deste serviço. A avaliação é composta de duas partes: (*i*) coleta de dados através de um aplicativo, e (*ii*) descoberta de conhecimento pela associação dos dados coletados.

Como resultado principal do desenvolvimento deste trabalho, espera-se avaliar o serviço e detectar onde o serviço pode ser melhorado. Perguntas que devem ser respondidas: qual é o horário mais sobrecarregado? Qual é o horário no qual o ônibus está mais vazio? Quais são as paradas com maior demanda? Como a demanda está distribuída pelos dias da semana? Onde estão as paradas? Espera-se, de forma geral, colaborar eventualmente para a qualidade do serviço do transporte intra-campus.

6.2 Avaliação da comunicação veicular utilizando diferentes tecnologias

Outro projeto de pesquisa em andamento busca avaliar a possibilidade de implementar a comunicação veículo-infraestrutura usando diferentes tecnologias. A plataforma Arduino é uma plataforma bem restrita, em termos computacionais. Entretanto, possui um conjunto amplo de módulos de comunicação, com diferentes tecnologias (Bluetooth, Bluetooth LE, Wi-Fi, LoRA[10]). Estas tecnologias já foram adquiridas pelo LSC e estão sendo avaliadas pelo aluno Leonardo de Abreu Schmidt.

7 Conclusões

Este texto apresentou os projetos desenvolvidos e em desenvolvimento pela equipe do LSC da UFSM no contexto de cidade inteligentes, ao longo dos últimos 8 anos. Destacam-se dois eixos: transportes e automação residencial, e duas abordagens: prototipação e simulação. Os projetos contam ou contaram com

[10] https://www.lora-alliance.org

colaborações externas, onde destaca-se a Universidade Federal do Rio Grande do Sul (UFRGS, Porto Alegre), Institut Polytechnique des Sciences Avancées (IPSA, Ivry-sur-Seine, França) e Universidade Humboldt (Berlim, Alemanha). É importante ressaltar que as soluções apresentadas pelo grupo de pesquisa focam sempre no uso de plataformas de baixo custo e de especificação aberta.

Referências

1. M. Behrisch, L. Bieker, J. Erdmann, D. Krajzewicz. SUMO – Simulation of Urban MObility: an overview, In: Proceedings of the 3rd SIMUL: The Third International Conference on Advances in System Simulation - ThinkMind (SIMUL 2011). pp. 55-60.
2. E. K. Dinanga, M. Pasin. Toward equitable vehicle-based intersection control in transportation networks. In: Proceedings of the 8th International Workshop on Agents in Traffic and Transportation (ATT-2014), Vizzari, Kluegl, Vokrinek (eds.), joint with the 13th Int. Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014). Paris, França. pp. 48-55.
3. R. Fao de Moura, E. T. Zancanaro, L. Lunardi-Lemos, M. Pasin. Avaliação da eficiência de algoritmos para semáforos virtuais em redes veiculares. In: ERAD 2015, 2015, Gramado - RS. Anais da XV ERAD, 2015. pp. 185-188.
4. M. Pasin, B. Scheuermann, R. Fao de Moura. VANET-based intersection control with a throughput/fairness tradeoff. In: Proceedings of the 8th IFIP Wireless and Mobile Networking Conference (WMNC 2015), 5-7 Outubro 2015, Munique, Alemanha. pp. 208-215.
5. L. Lunardi Lemos, M. Pasin. Intersection control in transportation networks: opportunities to minimize air pollution emissions. In: IEEE 19th International Conference on Intelligent Transportation Systems (ITSC 2016), Novembro 2016, Rio de Janeiro, Brasil. pp. 1616-1621.
6. M. Pasin, F. S. Perini, Ana L. C. Bazzan. Self-management as support to an advanced traveler information system. In: Proceedings of the SBSI 2012 - São Paulo - SP: Sociedade Brasileira de Computação, 2012. pp. 138-143.
7. M. Pasin, Ana L. C. Bazzan, F. S. Perini. Toward a self-managed distributed infrastructure to an advanced traveler information system. In: Proceedings of the Autosoft 2011, em conjunto com CBSoft 2011, São Paulo - SP. pp. 33-39.
8. C. F. De Gasperi, M. Pasin. Proposta para atualização de SGBDs para aplicações com demanda de contínua disponibilidade usando suporte do modelo de componentes de software. In: Proceedings of the Workshop de Testes e Tolerância a Falhas 2013 - WTF 2013, em conjunto com Simpósio Brasileiro de Redes de Computadores 2013 (SBRC 2013), Brasilia - DF, Brazil. Porto Alegre: Sociedade Brasileira de Computação, 2013. v. 1. pp. 3-16.
9. R. Silveira Rodrigues, M. Pasin, R. Machado. Indoor position tracking: an application using the Arduino mobile platform. In: Proceedings of the 10th IFIP Wireless and Mobile Networking Conference (WMNC 2017), 25-27 Setembro 2017, Valencia, Espanha.
10. M. L. Mota da Silva. Serviço para gerenciar uma Smart House: uma implementação usando a plataforma Arduino. Trabalho de Graduação, Curso de Sistemas de Informação, Universidade Federal de Santa Maria, Santa Maria - RS, Julho de 2016. 33p.