Implementando uma IDE para SMA Embarcados*

Vinicius Souza de Jesus[1], Nilson Mori Lazarin[12], Carlos Eduardo Pantoja[12], Gleifer Vaz Alves[3], Jose Viterbo Filho[1]

[1]Universidade Federal Fluminense (UFF)
Niterói, RJ – Brazil

[2]Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (Cefet/RJ)
Rio de Janeiro, RJ – Brazil

[3]Universidade Tecnológica Federal do Paraná (UTFPR)
Ponta Grossa, PR – Brazil

vsjesus@id.uff.br, {nilson.lazarin,carlos.pantoja}@cefet-rj.br

gleifer@utfpr.edu.br, viterbo@ic.uff.br

Abstract. *Developing an Embedded MAS is a task that requires the domain of several areas of knowledge; therefore, the Embedded MAS architecture used in this work is divided into four layers: reasoning, serial, firmware, and hardware. Different knowledge is required at each project layer: electronics, low-level programming, object-oriented and agent-oriented programming. This work seeks to present the first results of a specialized web Integrated Development Environment (IDE) for Embedded MAS to simplify their development.*

Resumo. *Desenvolver um SMA Embarcado é uma tarefa que exige o domínio de diversas áreas de conhecimentos, sendo assim, a arquitetura de SMA Embarcado utilizada neste trabalho está dividida em quatro camadas: reasoning, serial, firmware e hardware. Em cada camada do projeto são necessários conhecimentos distintos: eletrônica, programação de baixo nível, programação orientada objetos e orientada a agentes. Este trabalho busca apresentar os primeiros resultados de um Ambiente de Desenvolvimento Integrado (IDE) web especializado para SMA Embarcados visando simplificar seu desenvolvimento.*

1. Introdução

Sistemas Embarcados são sistemas encapsulados em dispositivos físicos funcionando de maneira exclusiva [Heath 2003]. Sistemas Multi-Agentes (SMA) são sistemas compostos por múltiplos agentes autônomos e pró-ativos, com capacidade de tomada de decisão. Esses agentes podem interagir entre si para que coletivamente alcancem um objetivo comum ao SMA [Wooldridge 2000]. Os SMA podem ser aplicados em ambientes virtuais, ou em ambientes físicos. Os SMA aplicados em ambientes físicos utilizam dispositivos físicos com sensores, atuadores e microcontroladores para interagirem com o ambiente, denominado SMA Embarcado [Souza de Castro et al. 2020].

Comumente utiliza-se uma arquitetura em quatro camadas (*raciocínio; serial; firmware e hardware*), para facilitar a embarcação de SMA. A camada de raciocínio é

*This short paper describes a demonstration submitted to WESAAC.

a camada onde está situado o SMA e é responsável pela cognição. A camada serial faz o interfaceamento entre o SMA e o microcontrolador do dispositivo físico. A camada de *firmware* recebe as mensagens do SMA para atuar no ambiente e envia as informações do ambiente para o SMA. Por fim, a camada de *hardware* é composta pelos sensores e atuadores do dispositivo [Pantoja et al. 2016].

O desenvolvimento de um SMA Embarcado é uma tarefa que exige o domínio de conceitos de diversas áreas, tais como: eletrônica, na camada de *hardware*; programação de baixo nível, na camada de *firmware*; programação orientada a objetos, na camada serial; por fim, programação orientada a agentes, na camada de raciocínio. Portanto, o objetivo deste trabalho é apresentar os resultados iniciais da proposta de um Ambiente de Desenvolvimento Integrado (IDE) web para o desenvolvimento de SMA Embarcados que busca auxiliar na simplificação do processo de embarcação de SMA. Com isso, o desenvolvimento fica centralizado e o projetista não necessita utilizar diferentes ambientes de desenvolvimento.

Este trabalho está organizado da seguinte forma: na Seção 2 são apresentados os trabalhos relacionados, suas limitações e um comparativo com este trabalho; na Seção 3 é apresentada a proposta da IDE para desenvolvimento de SMA Embarcados; por fim, na Seção 4 são apresentadas as considerações finais e as indicações de trabalhos futuros.

2. Trabalhos Relacionados

Nesta seção são apresentados os trabalhos relacionados, as principais limitações de cada um e uma comparação com as contribuições do trabalho proposto. Considerando a literatura de SMA embarcado alguns trabalhos podem ser destacados como: O *Lego Agent* [Jensen 2010] que é uma extensão dos agentes do *Framework Jason* [Bordini et al. 2007] com ações internas capazes de enviar comandos via *Bluetooth* para uma plataforma *Lego Mindstorm NXT*. Porém, como o SMA não fica dentro da plataforma, não caracteriza uma embarcação, mas sim um controle remoto.

O agente *ARGO* [Pantoja et al. 2016], é uma extensão da arquitetura de agentes do *Framework Jason* capaz de captar percepções e atuar no ambiente físico. Além disso, o trabalho possibilita o uso de diferentes opções de microcontroladores na camada de *firmware*, como *Arduino* (via *Javino* [Lazarin and Pantoja 2015]) ou *PIC* (via *Javic* [Guinelli and Pantoja 2016]). Todavia, é necessário utilizar a *Arduino IDE* ou o *software Proteus* para o desenvolvimento na camada de *firmware*.

Com isso, é possível observar que nos trabalhos anteriores, o projetista de SMA Embarcado deve necessariamente utilizar diferentes IDE para o desenvolvimento de cada camada do SMA Embarcado. Portanto, este trabalho busca centralizar o desenvolvimento de SMA Embarcado em uma única IDE que dê suporte para todas as camadas.

3. IDE para o Desenvolvimento de SMA Embarcado

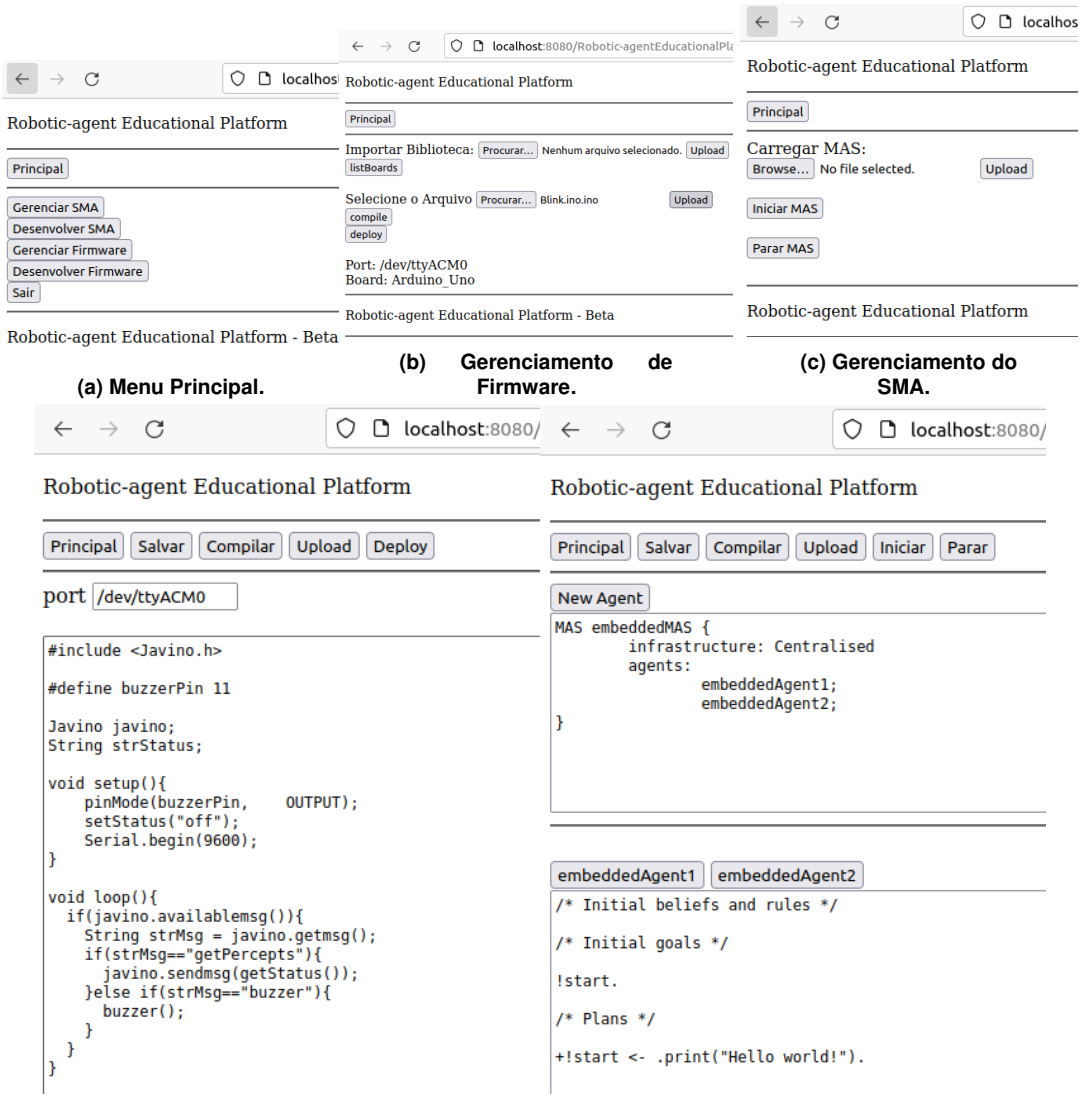
Nesta seção é apresentada uma IDE que busca facilitar o desenvolvimento de SMA Embarcados, através da centralização da codificação das camadas de *firmware* e raciocínio em um único ambiente de desenvolvimento. A IDE proposta é uma aplicação web que dispõe de dois módulos para o projetista de SMA Embarcado:

- **Desenvolvimento**: módulo destinado à codificação do raciocínio dos agentes no

SMA Embarcado e do *firmware* no microcontrolador responsável pelo controle do hardware.

- **Gerenciamento**: módulo destinado a se conectar com o ChonOS[1], um sistema operacional para SMA Embarcado, em execução no dispositivo físico, permitindo ao projetista executar *upload* e *deploy* de *firmware* do microcontrolador; além de realizar *upload*, iniciar ou parar um SMA.

Para o desenvolvimento de um SMA Embarcado utilizando a IDE proposta, primeiramente o projetista deve inserir as informações de login do dispositivo físico a qual deseja embarcar um SMA. Após isso, será direcionado para a tela de Menu principal (Figura 1a) onde o mesmo terá acesso a todas as funcionalidades da IDE, tais como: fazer o gerenciamento do *firmware* (Figura 1b) onde pode-se compilar e realizar o *deploy*; fazer o gerenciamento do SMA (Figura 1c) onde pode-se realizar *upload*, parar ou iniciar.



(a) Menu Principal.

(b) Gerenciamento de Firmware.

(c) Gerenciamento do SMA.

(d) Tela Desenvolvimento Firmware.

(e) Tela Desenvolvimento SMA.

Figura 1. Telas da IDE

[1]http://chonos.sf.net

Ainda na tela de Menu Principal (Figura 1a), a IDE possui botões para acessar as funcionalidades do módulo de desenvolvimento. Na tela de desenvolvimento de SMA (Figura 1d) é possível compilar, salvar, realizar *upload*, *deploy* do SMA. Além disso, a tela permite desenvolver o código fonte dos agentes, inserir novos agentes e editar a estrutura do SMA. Na tela de desenvolvimento de *firmware* (Figura 1e) é possível compilar, editar, salvar, fazer *upload*, *deploy* do código fonte do *firmware*.

4. Considerações Finais

Este trabalho apresenta as principais dificuldades na embarcação de SMA evidenciando a necessidade de domínio em diferentes áreas de conhecimentos (Eletrônica, Programação de baixo nível, Programação orientada a objetos e orientada a agentes). Além disso, são apresentados os resultados iniciais de uma IDE que se encontra em estágio de construção, a qual objetiva a simplificação e auxílio no desenvolvimento de SMA Embarcados.

Como trabalhos futuros pretende-se adicionar um verificador de sintaxe em tempo real para o auxílio no desenvolvimento. Além disso, utilizar o *Cascading Style Sheets* (CSS) para estilizar a interface gráfica da IDE. Por fim, deseja-se viabilizar o desenvolvimento de SMA Embarcados em diferentes linguagens de programação de agentes. Sendo assim, pretende-se implementar um módulo para a linguagem GWENDOLEN [Dennis 2017], que é uma linguagem usada para programação de agentes racionais e ainda possibilita a verificação formal do comportamento dos agentes.

Referências

Bordini, R. H., Hübner, J. F., and Wooldridge, M. (2007). *Programming Multi-Agent Systems in AgentSpeak using Jason*. John Wiley & Sons Ltd.

Dennis, L. A. (2017). Gwendolen semantics: 2017. Technical Report ULCS-17-001, University of Liverpool, Department of Computer Science.

Guinelli, J. V. and Pantoja, C. (2016). A Middleware for Using PIC Microcontrollers and Jason Framework for Programming Multi-Agent Systems. In *Anais do Workshop de Pesquisa em Computação dos Campos Gerais WPCCG*, volume 1, Ponta Grossa.

Heath, S. (2003). *Embedded systems design*. Newnes, Oxford ; Boston, 2nd ed edition.

Jensen, A. S. (2010). Implementing Lego Agents Using Jason. *CoRR*, abs/1010.0150.

Lazarin, N. M. and Pantoja, C. E. (2015). A Robotic-agent Platform for Embedding Software Agents Using Raspberry Pi and Arduino Boards. In *Proceedings of the 9th Software Agents, Environments and Applications School (WESAAC)*, pages 13–20, Niterói.

Pantoja, C., Junior, M., Lazarin, N. M., and Sichman, J. (2016). ARGO: A Customized Jason Architecture for Programming Embedded Robotic Agents. In *Fourth International Workshop on Engineering Multi Agent Systems (EMAS 2016)*, Singapore.

Souza de Castro, L., Manoel, F., Souza de Jesus, V., Pantoja, C., Borges, A., and Vaz Alves, G. (2020). Integrando sistemas multi-agentes embarcados, simulação urbana e aplicações de iot. In *XIV Workshop Escola de Sistemas de Agentes, seus Ambientes e Aplicações (WESAAC 2020)*.

Wooldridge, M. J. (2000). *Reasoning about rational agents*. Intelligent robotics and autonomous agents. MIT Press, Cambridge, Mass.