# An Analysis of Javino Middleware for Robotic Platforms Using Jason and JADE Frameworks

**Dayana Junger[1], João Victor Guinelli da Silva[1], Carlos Eduardo Pantoja[1]**

[1]CEFET/RJ – UnED Nova Friburgo – Av. Gov. Roberto da Silveira, 1900 – Nova Friburgo – RJ – Brasil

dayanacomputer@hotmail.com, {joao.silva, pantoja}@cefet-rj.br

***Abstract.*** *Robotics is the field of knowledge in which it is possible to use agent-oriented programming languages to develop robots capable of interacting with their environment by using sensors and actuators. Some situations require that agents try to achieve tasks in the physical world by both sending commands to actuators and perceiving data from sensors. Based on this, Javino middleware can be employed to interconnect Java-based MAS and hardware devices. Thus, our main objective in this paper is to prove the usefulness of using Javino to perform the communication between MAS applications and Arduino boards. For this, we conduct experiments to analyze the Javino's performance in Jason and JADE frameworks and discover if there is any condition in which this middleware should not be used.*

## 1. Introduction

Intelligent agents are computational systems that can both perceive and act, by themselves, into their environment. They can work together with other agents, forming a Multi-Agent System (MAS) [Wooldridge, 2009]. Similarly, robots are physical agents that use their sensors and actuators to get information about the environment and act on it based on their own decisions [Matarić, 2007]. Because of the similarity between these definitions, the idea of embedding MAS in robots to implement the decision-making mechanism is widely used; however, it is not a simple task, and some works try to achieve this using different approaches.

In [Soriano et al., 2013], for example, for each robot there is a computer running a simulated agent implemented using the Java Agent Development Framework (JADE) [Bellifemini et al., 2004]. In this work, the robots and its related computers communicate through Bluetooth. On the other hand, in [Lazarin and Pantoja, 2015], the authors embed software agents programmed using Jason framework [Bordini et al., 2007] into a Raspberry Pi[1]. The MAS embedded can control Arduino boards through the serial port using Javino middleware.

Considering this, our main objective of this paper is to prove the usefulness of using Javino to perform the communication between MAS applications and Arduino[2] boards. For this, we perform experiments to analyze the Javino's performance in the frameworks mentioned above to discover if there is any condition in which this middleware should not be used. Since Javino employs serial ports, if different agents compete for the same port at the same time, conflicts may occur.

[1] www.raspberrypi.org/

[2] www.arduino.cc/

The rest of this paper is structured as follows: in Section 2, we introduce concepts from Jason, JADE and Javino. We analyze the middleware in Section 3. In Section 4, some related works are discussed and in Section 5, the conclusion is seen.

## 2. Development of MAS

Taking either Jason or JADE along with Javino is a procedure which can be used to allow the development of robots using Arduino boards being controlled by the MAS. However, there are some divergences between Jason and JADE that influence the way each multi-agent system is embedded into robotic devices. This section presents some basic concepts of Jason, JADE and Javino.

### 2.1. Jason Framework

Jason is composed of agents which have plans, desires, intentions and beliefs. These agents can be situated in a simulated environment, in which it is possible to program which perceptions they can access. In the environment, it is programmed the external actions that agents can execute to respond to a stimulus. Furthermore, Jason includes some extensions, such as facilities for communication.

Jason agents implement a reasoning where they perceive the environment and updates their belief base with the perceptions received. Any update in the belief base generates an event which is added to an event list. After that, the agent checks for acceptable messages in its mailbox. Next, an event is chosen, and all the agent's plans are selected to be analyzed. Then a function gets one of these plans to be executed. Finally, if there is more than one ready-to-use intention, a function takes one of them, and the first non-executed action from this intention is performed.

### 2.2. JADE Framework

JADE [Bellifemini et al., 2004] is a framework for MAS development which is based on the programming language Java, and that fully implements the Foundation for Intelligent Physical Agents (FIPA) specification. Its main objective is to provide usable and accessible services for different-knowledge-level developers, which do not need to know FIPA.

In this framework, the agent platform is composed of one or more containers, in which the agents live, that are linked to the main container. The main container has some special responsibilities in the platform, such as managing information about containers and agents. Furthermore, it also hosts two special sorts of agents, one that provides the white page services, and another one that provides the yellow page services.

### 2.3. Javino Middleware

Javino [Lazarin and Pantoja, 2015] is a middleware used for performing the communication between an Arduino and a Java program that implements a protocol specified to provide an error-detection mechanism for data exchange through the serial port. That middleware is composed of a double-sided library, in which one acts on the Java side while another one acts on the Arduino side.

The main objective of the protocol is to provide a communication bridge between the hardware and the MAS by ensuring that there is no error in the information received from any side during data communication. The authors suggest that this error detection may be particularly useful for embedded MAS because a wrong message can stop the effectors or/and even the agent, or, even worse, let the agent behave wrongly by taking damaged data.

## 3. Middleware Analysis

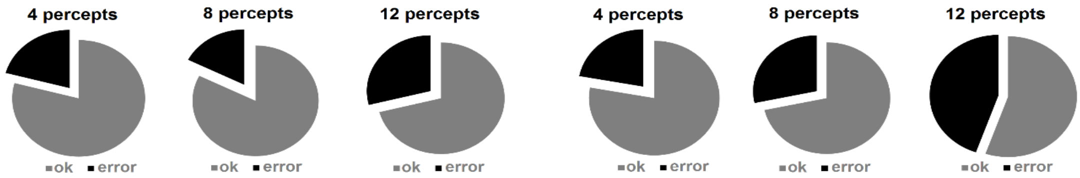
In order to evaluate Javino usage along with robotic agents, we executed experiments using Jason and JADE to verify if the middleware can transmit data with an acceptable loss rate; if it is possible to use Javino when there is competition for the same serial port; if the delaying time or the perception sizes have any influence on data transmission; and if the framework employed influences the middleware usage.

To perform the tests, we employed a robot with 12 ultrasonic sensors, which could send perceptions for the agent-oriented software by using our approach. We divided the experiments into two types: *Listen* mode and *Request* mode. In *Listen* mode, the controller sends data from sensors continuously, and, in the *Request* mode, the agent requests data by sending a serial message to the controller. For each one of these experiments, the robot sent 4, 8 and 12 different perceptions to the MAS. Furthermore, in the *Listen* mode, the delaying time-tested between each sent message was 100, 300, 500 and 700 milliseconds (ms). We also varied the number of agents *(n)* which were competing for the same serial port by starting with *n=1*. For each experiment performed, we monitored the messages sent to MAS for 5 minutes.

After the experiments execution with the *Listen* mode, it was possible to verify the percentage of correct messages sent for 4, 8 and 12 perceptions (Figure 2a) in Jason. The error rates were, respectively, 21%, 18% and 29% of 2271, 2122 and 1900 messages received from Javino. Firstly, we perceived that while the size of the message increased, the number of received messages decreased, since it needs more computational time for processing and verifying the correctness of the message.

In JADE, the error rates for 4, 8 and 12 perceptions were 22%, 28% and 45% of 2409, 2118 and 1983 messages sent (Figure 2b). Javino presented the same behaviour of Jason when the size of messages increased. JADE presented error rates higher than Jason, especially in messages with 12 perceptions. It could have occurred because the reactive architecture of JADE is slightly faster than the BDI one, so the agent tries to get more messages from sensors. However, it was necessary to identify which was causing the bottleneck of errors in communication. In both cases, we considered all messages sent without considering the delaying time.

For this, we analyzed the messages sent with a delaying time of 100, 300, 500 and 700 milliseconds. In Jason, the correct message rates with delaying time of 100 ms for 4, 8 and 12 perceptions were, respectively, 2%, 33% and 45% (Figure 3a). When the delaying time increased to 300 ms, the rates of correct messages were 100%, 99% and 99%. We noticed that while the delaying time decreased, the error rate increased. It makes sense since there were several messages arriving at almost the same time. Then, the agent was trying to get messages with errors.




(a) **Percentage of correct messages sent with *n* perceptions using Jason.**

(b) **Percentage of correct messages sent with *n* perceptions using Jason.**

**Figure 2: Percentage of correct messages sent from Javino to Jason (a) and Javino to JADE (b).**

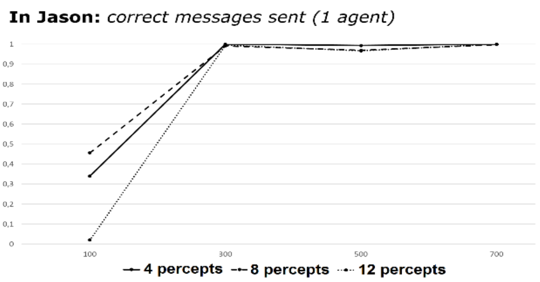
In JADE, the rate of correct messages with a delaying time of 100 ms for 4, 8 and 12 perceptions were 99%, 99% and 0.1%. When the delaying time increased to 300 ms, the rates were 99%, 99% and 88% (Figure 3b). In this case, the JADE lost some messages, but it was not significantly different from Jason. Until this point, all the experiments used only one agent. It was important to verify the behaviour of Javino and the MAS when n agents were competing for the same serial port.

Then, in this experiment, which uses Jason, it is possible to observe the rate of correct messages in which the delaying time was related to more than one agent (Figure 3c). When one or more agents competed for the same serial port, the error rates increased, significantly, since only one agent should manipulate a serial port. The MAS stopped running after 2 or 3 minutes. In JADE, the rates of correct messages sent when more than one agent was competing for the serial port were 14% and 0% for 2 and 3 agents, respectively (Figure 3d). Therefore, we noticed that it is not recommended to use Javino with 2 or more agents accessing the same serial port since the agent competes for the port. However, if the agents access different serial ports, there is no problem in using Javino.

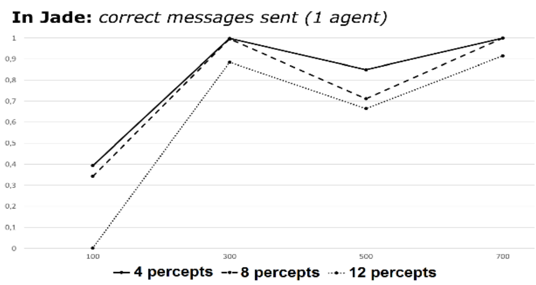
After the experiments execution with the *Request* mode, by using the same parameters of the former experiments, we noticed that the rate of correct messages sent was 100% for 4, 8 and 12 perceptions. When it was considered the delaying time, the rate of correct messages was 100% for all parameters. It happened because of the characteristic of the Request mode since the agent is responsible for requesting the information from the controller, which response by transmitting the perceptions from sensors. Consequently, the delaying time (used in the controller programming), in this situation, does not influence the middleware performance, since the requirement depends on the agent and do not depend on the controller.

## 4. Related Work

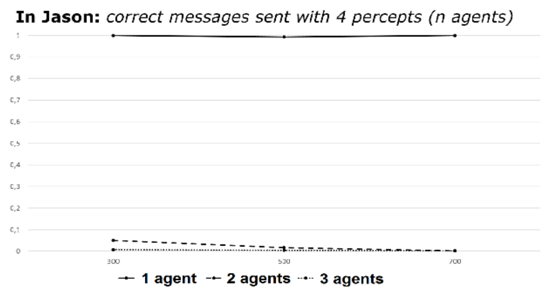
In this section, we discuss some related works, which employ robotic agents. In [Jensen, 2010], the author presents an extended version of Jason that combines this framework along with Lego robots. Furthermore, it proposes, as an extension, several internal actions to establish the communication between physical agents and the Lego Mindstorm NXT toolkit. It states that a Lego agent can be represented as a Jason agent. It uses Bluetooth as a communication channel and LeJOS as the middleware between Jason and Lego agents. Additionally, it presents a communication process between Lego agents. Despite being able to communicate, the author says that the robots performance is slow, and the robotic platforms are tied to Lego Mindstorm NXT. Differently, we propose an architecture which provides the possibility of developing the same functionalities by using any robotic platform and, for instance, Jason and JADE frameworks.



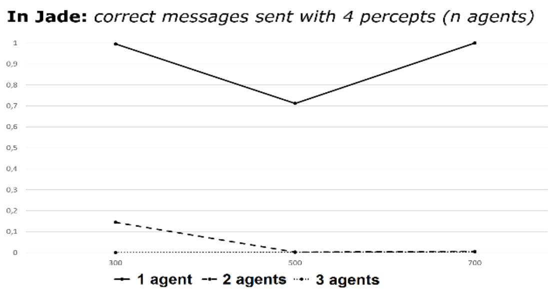

**(a) Serial port usage by 1 agent in Jason.**




**(b) Serial port usage by 1 agent in JADE.**




**(c) Serial port usage by *n* agent in Jason.**




**(d) Serial port usage by *n* agent in Jason.**

**Figure 3: Messages received *per* delaying time by Jason (a) and by JADE (b). Agents competing for the same serial port in Jason (c) and JADE (d).**

Describing the implementation of MAS into embedded systems with restricted resources is a procedure discussed in [Soriano et al, 2013] and it suggests that it is possible to develop different robotic applications by using JADE. As the communication time is taken into account, in the sense that it is an important aspect to be considered to avoid the system uncertainty, a kinematic control is locally employed to avert this sort of problem, and it uses two hardware platforms (Lego Mindstorms NXT and E-puck) to effect practical tests.

In one of these tests, the robots are supposed to follow points by starting from the closest one to a well-known position. In the tests, each robot is connected via Bluetooth to a computer in which the correspondent software agent is running, and these computers are part of a network that forms the MAS by using JADE. Then, these robots can perform cooperative actions into the environment. However, the methodology approach does not address the issue of embedding directly an agent-oriented programming in the hardware. In this approach is necessary a computer to implement the agent's functionality of the robot.

## 5. Conclusions

This paper presented an analysis of Javino, a middleware that implements a communication protocol with error detection that can be employed in robotics and MAS. To perform the tests, it was used a 12-sensor robot to analyze Javino's performance along with a MAS, which was programmed through Jason and JADE. In the experiments performed by using the Listen mode, we showed that it is not possible to use Javino with a delaying time smaller than 100 ms. It is recommended to use it by setting up a waiting time greater than 300 ms, independently of message sizes (which should be up to 256 bytes). In the Request mode, there were no errors related to communication in both agent-oriented frameworks. We stated that if more than one agent competes for the same serial port in which the controller is connected, the Javino does not work properly.

For further works, we aim to provide a mechanism to allow the communication between robotic agents through Javino. For this, we aim to extend the Javino's protocol to address this issue and test it in different MAS frameworks. In this protocol, it must be provided a way to enable robotic agents (which are embedded along with individual MAS) communicate with each other. So, the agent's' architecture must be modified to support this new communication protocol.

## References


Bellifemini, F., Caire, G., Greenwood, D., (2004). Developing Multi-Agent Systems with JADE. John Wiley and Sons, London.

Bordini, R.H., Hubner, J.F., Wooldridge, W. (2007). Programming Multi-Agent Systems in AgentSpeak Using Jason. John Wiley and Sons, London.

Jensen, A.S., (2010). Implementing Lego Agents Using Jason, http://arxiv.org/pdf/1010.0150.pdf, November.

Lazarin, N.M., Pantoja, C.E. (2015). A Robotic-agent Platform for Embedding Software Agents using Raspberry Pi and Arduino Boards. In Proceedings of 9th Software Agents, Environments and Applications School: WESAAC'15, Niterói, Brazil.

Matarić, M. J. (2007). The Robotics Primer. MIT Press.

Soriano, A., Marín, L., Valera, Á., Vallés M. (2013). Multi-Agent Systems Integration in Embedded Systems with Limited Resources to Perform Tasks of Coordination and Cooperation. In: Proceedings of 10th International Conference on Informatics in Control, Automation and Robotics: p. 140 - 147, ICINCO' 13, Reykjavik, Iceland.

Wooldridge, M. (2009). An Introduction to Multi-Agent Systems. John Wiley and Sons, London.