

# Autonomy Development for Unmanned Aerial Vehicles with Jason Agents

Marcelo T. Hama<sup>1</sup>, Rafael H. Bordini<sup>2</sup>

<sup>1</sup>Instituto de Informatica – Universidade Federal do Rio Grande do Sul (UFRGS)  
Caixa Postal 15.064 – CEP 91501970 – Porto Alegre – RS – Brazil

mthama@inf.ufrgs.br, r.bordini@inf.ufrgs.br

## 1. Introduction

In many situations, the development of autonomy addressing particular types of systems is not an easy task due to the abstract gap between low-level instructions (more close to hardware level) and high-level operations (more close to application level). In these types of situations, we cannot just assume a direct implementation in view of the fact that it can make the overall system inflexible and hard to extend. In this scenario, a usual plan is to apply or construct interfaces that serve as technologic bridges or use some kind of middleware to mediate the communication between the systems. In the context of this work, we face an abstract gap in an intelligent behaviour development for unmanned aerial vehicles that use an open API firmware which can send and receive data through a universal asynchronous receiver/transmitter (UART) following a particular protocol, and being controlled by a professional through radio transmitters. Generally, to save memory and processing, UAVs firmware is programmed with the use of languages that are close to hardware level (e.g., C and Assembly), but as drawback it makes the autonomy development technically expensive: autonomy involves high-level operations like intelligent team cooperation and inter-agent communication management; Input/Output management, bit-conversions, and system interruptions are not issues that this type of development should be concerned with.

## 2. Theoretical Basis

The application of intelligent agents in the context of UAV is not in itself a novelty and there are several research projects in this subject, but as far as we have surveyed, our research is the first one that proposes an open-source framework with use of a BDI agent-oriented programming language in the scope of unmanned aerial vehicles. The agent design can be seen as an interesting way to implement autonomy for UAVs due to the conceptual modeling similarities. The main approach is the application of multiple Jason [Bordini and Hubner 2007] agents as auto-pilots for a UAV, in a conception that is reminiscent of human pilots inside a manned aircrafts. Jason is an AgentSpeak, an extended version of AgentSpeak(L) interpreter, and runtime IDE where it is possible to develop and simulate Belief-Desire-Intention [Bratman 1987] agents and multi-agent systems.

## 3. Our Model

The model is based on an analogy between a UAV controlled by agents and common manned aircraft controlled by humans. In this analogy, an agent has capabilities that allow it to communicate with others agents or with remote services/systems while the manned aircraft pilots and tripulations can speak with each other or send/receive messages to remote personnel. The Jason agents have .ASL Files (a kind of source file) that describe its behaviour and plans, while the manned tripulation has its human intelligence and follows a flight plan. The manned aerial vehicle commands (i.e., accelerate, initiate take-off, activate landing gear, turn on ailerons, change flap angles, and so on) are modeled as a set of capabilities in an action protocol. The final concept is to see the manned aerial vehicle as the UAV with Jason virtual environment as the pilot cabin. Table 1 shows the modelling analogies.

Unmanned Aerial Vehicle		Manned Aerial Vehicle
Agents	models	Manned Aircraft Tripulation
AgentSpeak .ASL File	models	Tripulation Intelligence and Flight Plans
Action Protocol	models	Manned Aircraft Commands
Virtual Environment	models	Pilot Cabin
UAV	models	Manned Aircraft

Table 1. Modelling analogies.

## 4. Architecture Conceptual Model

The framework runs on a light operating system within an upgraded UAV prototype. The UAV can send/receive data bytes through the UART which make calls directly to the UAV firmware API and its low-level implementation. These data bytes are converted into readable stream by

an **Input/Output Manager**, and processed by a **Converter** which uses a **Firmware Model** to construct valid values of arguments from incoming parameters, defining the framework Engine. These values are sent/received by the **Action Protocol** instance, and called by the implemented **Agent Architecture**. The **Agents** run within a **Jason MAS** deliberative reasoner, performing the interpretation of **.ASL Files** that describe the intelligent behaviours of each of these agents. The conceptual model is shown in Figure 1.

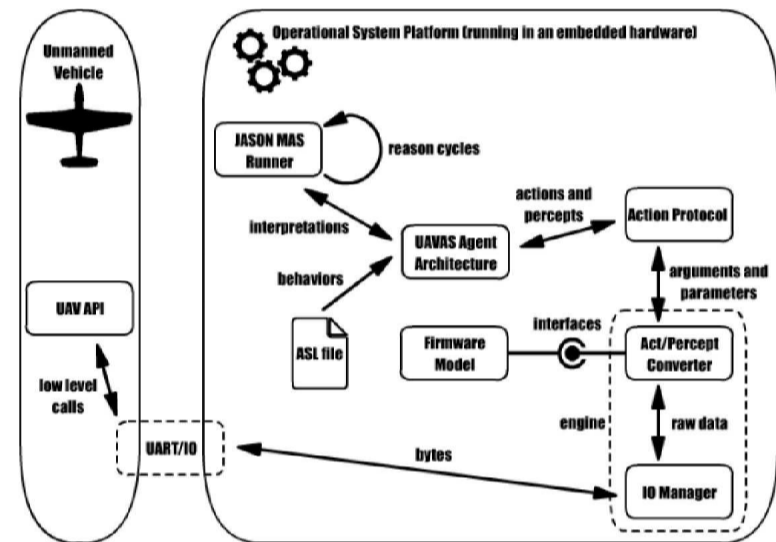


Figure 1. Conceptual model [Hama 2012].

## 5. Preliminary Results

As preliminary results, we are able to simulate the model infrastructure using Flight Model Simulator (FMS). FMS is an RC-Sim used to receive signals from radio transmitters and simulate them in a virtual UAV through a 3D graphic world. Currently, by encoding the generated streams in the same sequence created in transmitters we can simulate a set of operations. In Figure 2, we show a take-off simulation screenshot using FMS. The image shows a quadrotor rising vertically. For now, we are seeking to implement a specialised simulator that employs multiple virtual UAV instantiations, inter-UAV and intra-UAV communication, and an API to perceive environment state through query capabilities. Another research direction is towards solving hardware issues that have so far prevented us from implementing the model in real UAV prototypes.



Figure 2. Take-off simulation screenshot, using FMS [Hama et al. 2011].

## 6. References

Bordini, R. H. and Hubner, J. F. (2007). A Java-based interpreter for an extended version of AgentSpeak. University of Durham, Universidade Regional de Blumenau; Bratman, M. E. (1987). Intention, Plans, and Practical Reason. Harvard University Press, Harvard University; Hama, M. T., Allgayer, R. S., Pereira, C. E., and Bordini, R. H. (2011). Uavas: Agentspeak agents for unmanned aerial vehicles. In II Workshop on Autonomous Software Systems, page 7, Sao Paulo, SP, Brazil. Autosoft;