Autonomy Development for Unmanned Aerial Vehicles with *Jason* Agents

Marcelo T. Hama[1] **, Rafael H. Bordini**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

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

Abstract. *Aerospace technology takes a very important role in several subjects and, in this context, unmanned aerial vehicles (UAV) are evolving into a noticeable theme where the application of agent designs, for autonomy development, has become a much used approach over the last decade. This paper presents an agent-oriented model to develop autonomy and to program UAVs with the use of Jason agents.*

Keywords: Intelligent Agents, Unmanned Aerial Vehicle, Autonomy Development, Framework

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 languagens 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 [Wooldridge 2009] in the context of UAV is not in itself a novelty and there are several research projects in this subject, as we can see in [Jakob et al. 2010, Scerri et al. 2008, Sislak et al. 2008], 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[1] in the scope of unmanned aerial vehicles. The

[1]JACK Intelligent Agents has been used before, but it is a commercial and Java-oriented tool.

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, Bordini et al. 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*[2] interpretator and runtime IDE where it is possible to develop and simulate Belief-Desire-Intention [Bratman 1987] agents and multi-agent systems.

3. The Model

In an attempt to reduce the *abstraction gap* in software development for UAVs, we proposed an agent-oriented framework using *AgentSpeak* to control a UAV. The proposed model uses a technology bridge between the agent layer and UAV firmware layer, with an action protocol that defines a set of agent capabilities. The protocol actions are treated as capabilities by the agents within the UAV and, as a result, the UAV can implement partial or full autonomy using a BDI approach.

3.1. Abstraction Analogy

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.

3.2. 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 streams 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**

[2]An extended version of *AgentSpeak(L)* [Rao 1996].

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.

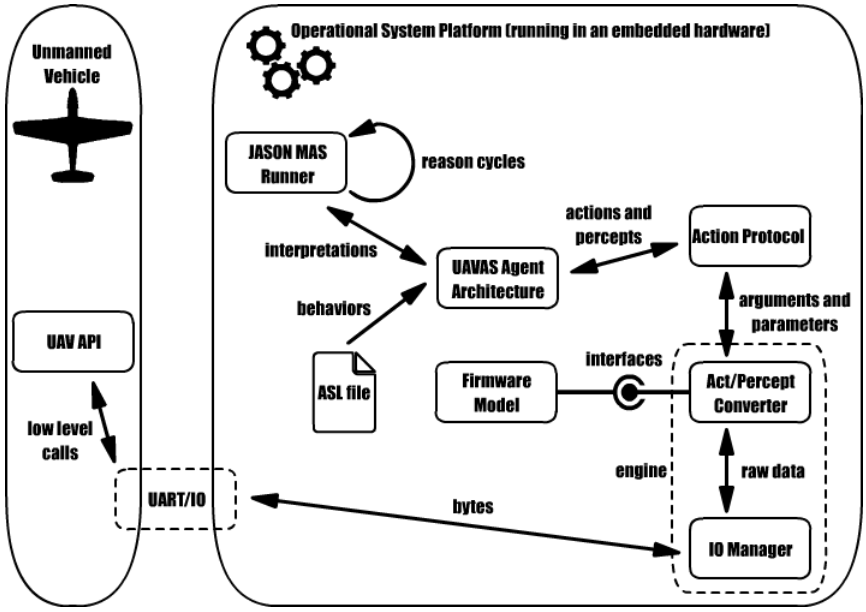

Figure 1. Conceptual model [Hama 2012].

4. Preliminary Results

As preliminary results, we are able to simulate the model infrastructure using Flight Model Simulator[3] (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[4]. In Figure 2, we show a take-off simulation screenshot using FMS. The image shows a quadrotor rising vertically.

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

[3]Project and download page at `http://n.ethz.ch/~mmoeller/fms/`.
[4]We could not simulate percepts because FMS does not provide an open API

5. Current and Future Works

In previous work, [Hama et al. 2011, Hama 2012], a technology bridge has been defined to address the *abstraction gap* in autonomy development for unmanned aerial vehicles. For now, we are seeking to implemen 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.

References

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

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. (2012). Uavas abstraction model - jason agents as pilots and tripulation for unmanned aerial vehicles (to appear). In *Simpósio Brasileiro de Inteligência Artificial*, Curitiba, PR, Brazil.

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, São Paulo, SP, Brazil. Autosoft.

Jakob, M., Semsch, E., Pavlicek, D., and Pechoucek, M. (2010). Occlusion-aware multi-uav surveillance of multiple uraban areas. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 1407 – 1408, Toronto, Canada. ACM Press.

Rao, A. S. (1996). Agentspeak(l): Bdi agents speak out in a logical computable language. In *Proc. 7th European workshop on Modelling Autonomous Agents in a Multi-Agent World*, pages 42 – 55, Secaucus, NJ, USA. Springer-Verlag.

Scerri, P., Gonten, T. V., Fudge, G., Owens, S., and Sycara, K. (2008). Transitioning multiagent technology to uav applications. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, pages 89 – 96, Carnegie Mellon University. International Foundation for Autonomous Agents and Multiagent Systems.

Sislak, D., Pechoucek, M., Volf, P., Pavlicek, D., Samek, J., Marik, V., and Losiewicz, P. (2008). Agentfly: Towards multi-agent technology in free flight air traffic control. *Defense Industry Applications of Autonomous Agents and Multi-Agent Systems*, pages 73 – 97.

Wooldridge, M. (2009). *An Introduction to MultiAgent Systems*. John Wiley and Sons Ltd, United Kingdom.