UDC 004.4:004.8

S. Lebid
M. Olenin, Cand. Of. Sci. (Physico-Mathematical)
(National Aviation University, Ukraine)

Ontology-driven distributed multi-agent systems in aviation

Nowadays issues of managing various processes in aviation are rather relevant. Whether it is a system of tickets search and booking, an air traffic control system, airborne sensors etc. Most of them are multi-agent systems. Such systems solve certain problems, by knowledge-based decision-making. While ontologies are used to represent information in such systems.

Definition of multi-agent systems

Multi-agent system are systems composed of multiple interacting computing elements, known as agents. Agents are computer systems with two important capabilities. First they are at least to some extent capable of autonomous action – of deciding for themselves what they need to do in order to satisfy their design objectives. Second, they are capable of interacting with other agents – not simply by exchanging data, but by engaging in analogues of the kind of social activity that we all engage in every day of our lives: cooperation, coordination, negotiation, and the like.

Multi-agent systems development tools review

One of the best known approaches to the development of cognitive agents is BDI (beliefs-desires-intentions) architecture. In the area of agent-oriented programming languages in particular, AgentSpeak has been one of the most influential abstract languages based on the BDI architecture. The type of agents programmed with AgentSpeak are sometimes referred to as reactive planning systems.

We usually find it convenient to make a distinction between an agent program and an agent architecture. The agent architecture is the software framework within which an agent program runs. The PRS is an example of an agent architecture; the plans are the program that inhabits this architecture. We write the program that will direct the agent behaviour, but much of what the agent effectively does is determined by the architecture itself, without the programmer having to worry about it. The language interpreted by Jason is an extension of AgentSpeak, which is based on the BDI architecture. Thus, one of the components of the agent architecture is a belief base, and an example of one of the things that the interpreter will be doing constantly,

without it being explicitly programmed − even though this can be customized to perceive the environment and update the belief base accordingly.
Besides interpreting the original AgentSpeak language, Jason also features:

1) strong negation, closed-world and open-world assumption;
2) handling of plan failures;
3) speech-act based inter-agent communication (and belief annotations on information sources);
4) support for developing Environments (in this case Java);
5) the possibility to run a multi-agent system distributed over a network (using SACI or JADE);
6) annotations in beliefs used for meta-level information and annotations in plan labels that can be used by elaborate (e.g., decision theoretic) selection functions;
7) meta events, declarative goal annotations, higher order variables and treating plans as terms, imperative style commands in plan bodies, and various other language extensions;

JADE (Java Agent DEvelopment Framework) is a software Framework fully implemented in the Java language. It simplifies the implementation of multi-agent systems through a middle-ware that complies with the FIPA specifications and through a set of graphical tools that support the debugging and deployment phases. A JADE-based system can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another, as and when required. JADE is completely implemented in Java language and the minimal system requirement is the version 5 of JAVA (the run time environment or the JDK).

Besides the agent abstraction, JADE provides a simple yet powerful task execution and composition model, peer to peer agent communication based on the asynchronous message passing paradigm, a yellow pages service supporting publish subscribe discovery mechanism and many other advanced features that facilitates the development of a distributed system.

Apache Jena is a Java framework to construct Semantic Web Applications. It provides a programmatic environment for RDF, RDFS and OWL, SPARQL, and includes a rule-based inference engine.

Apache Jena TDB provides a lightweight, scalable transactional storage and SPARQL query layer. TDB is a component for RDF storage and query. It support the full range of Jena APIs. TDB can be used as a high performance RDF store on a single machine.

SDB is a SPARQL database subsystem for Jena. It provides for large scale storage and query of RDF data sets using conventional SQL databases. The database tools for load balancing, security, clustering, backup and administration can all be used to manage the installation. It offers an RDF Triple Store facility with SPARQL interface on top of other database systems.

At its core, Jena stores information as RDF triples in directed graphs, and allows your code to add, remove, manipulate, store and publish that information.

The collection of standards that define semantic web technologies includes SPARQL - the query language for RDF. Jena conforms to all of the published standards, and tracks the revisions and updates in the under-development areas of the standard. Handling SPARQL, both for query and update, is the responsibility of the SPARQL API.

Conslusion

The technologies and tools reviewed are aimed at facilitating the task of developing robust ontology-based agent-oriented intellectual systems which can be applied in modern aviation.

References

1) Jason java-based AgentSpeak extension http://jason.sourceforge.net/wp/description/ Accessed August 25, 2018.
2) Programming Multi-Agent Systems in AgentSpeak using Jason, Rafael H. Bordini, Jomi Fred Hübner, Michael Wooldridge, WILEY 2007.
3) JAVA Agent DEvelopment Framework http://jade.tilab.com/ Accessed august 25, 2018.
4) Developing multi-agent systems with JADE, Fabio Bellifemine, Giovanni Caire, Dominic Greenwood, WILEY 2007.
5) Apache Jena Semantic Web framework https://jena.apache.org/ Accessed September 2, 2018.