

Agent-based Programming in 3APL

Intelligent Systems Group

Institute of Information and
Computing Sciences
Utrecht University
<http://www.cs.uu.nl/3apl/>

3APL is an abstract agent-based programming language for implementing cognitive agents and multi-agent systems [2]. 3APL programs are executed by an interpreter in Java, which makes use of an embedded Prolog reasoning engine. The language has a nice formal semantics. It is intended to bridge the gap between agent theory and agent programming.

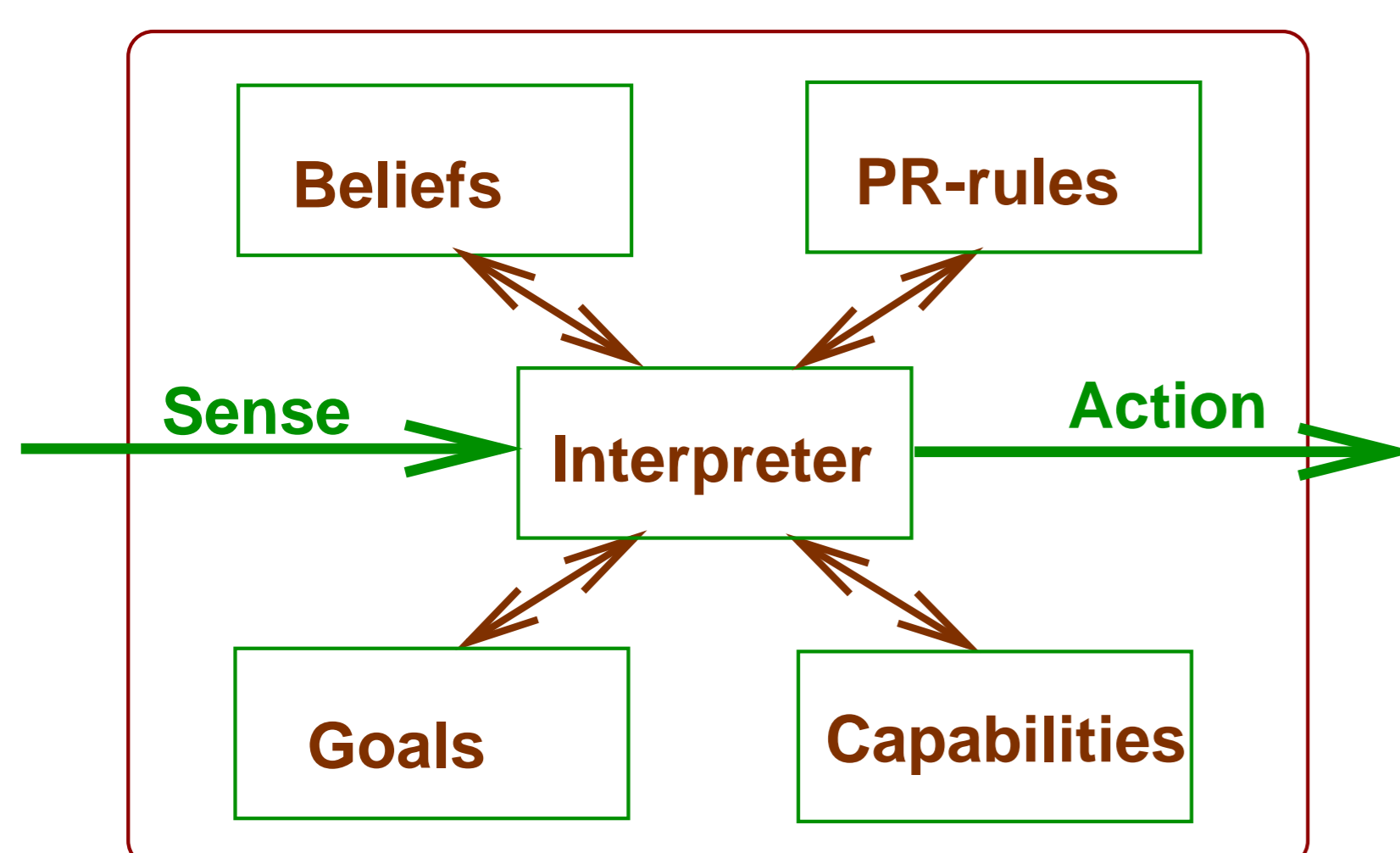


Figure 1: 3APL Architecture

Agent Architecture

Each agent consists of four modules with a central interpreter for control (figure 1). Triggered by sensing actions, the agent updates and revises its beliefs and goals. Using practical reasoning rules it selects appropriate actions from its capabilities to achieve its goals. Capabilities are of the form $\{Pre\}Action\{Post\}$. The belief base contains a set of facts. Rules are of the form $Goal \leftarrow Guard | SubGoals$, where subgoals are selected provided the guard condition is satisfied. In this way agents can be programmed that adjust their program to the circumstances. Goals in 3APL are procedural (goals-to-do). The language will be extended with declarative goals (goals-to-be).

```
PROGRAM "patrol_agent.3apl"  
CAPABILITIES:  
  {east(me)} GoWest() { west(me),  
                        NOT east(me) },  
  {west(me)} GoEast() { east(me),  
                       NOT west(me) }  
BELIEFBASE: west(me)  
GOALBASE: patrol()  
RULEBASE:  
  patrol() ← east(me) | GoWest();patrol(),  
  patrol() ← west(me) | GoEast();patrol().
```

Figure 2: Example Program

Robotic Control

3APL agents can be used to control the behaviour of physical robots. The results of sensing actions are translated into beliefs; the actions of the robot are carried out in an environment. Using reinforcement learning the robot is able to and adjust its program to the type of environment [3].

Agent Deliberation

In the current version agents apply a fixed deliberation cycle (figure 3). New research is concerned with a programmable deliberation cycle, where these steps can be taken in any order [1]. This allows the design of different agent types such as reactive agents or deliberative agents. Agent programming methodologies are developed to help the designer select an agent type that is most suited.

REPEAT

1. Find practical reasoning rules that match goals.
2. Find practical reasoning rules that match beliefs
3. Select practical reasoning rule to apply to goals.
4. Apply practical reasoning rule to goals.
5. Find goals to execute.
6. Select goal to execute.
7. Execute goal.

UNTIL goalbase is empty.

Figure 3: Deliberation Cycle

Multi-Agent Platform

We are developing a multi-agent platform, which supports several programming tools and multi-agent services (figure 4). These include an agent management system for life-cycle management, name registration, name lookup and authentication, a directory facilitator to query agent attributes and the services they provide, and the ability to set up agent communication channels for sending and receiving messages. To allow message passing, the architecture of figure 1 is extended with the capability to receive incoming messages and send messages to other agents. The platform is based on FIPA standards.

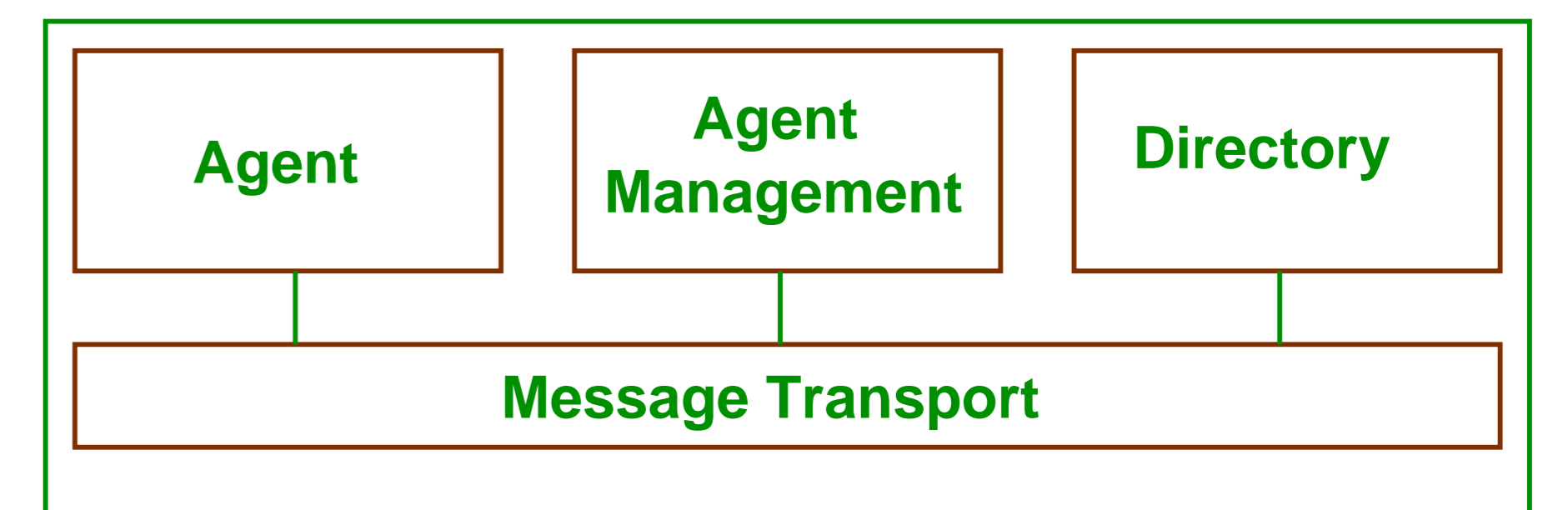


Figure 4: Multi-Agent Platform

Agent Communication

We are developing a set of communication primitives for different types of interaction, such as negotiation, information exchange or dispute. Using communicative acts like `propose(i, j, A)`, `accept(j, i, B)` or `reject(j, i, B)` agents can construct protocols for coherent interaction [4].

Conclusion

3APL is a powerful and elegant language for agent-based programming. It has been applied to robotic control, agent communication, multi-agent systems and theoretical research on the agent deliberation process.

Contributors

Frank de Boer, Mehdi Dastani, Frank Dignum, Joris Hulstijn, Meindert Kroese, John-Jules Meyer, Birna van Riemsdijk, Marco Wiering, Steven Anker, Koen Hindriks, Wiebe van der Hoek, Jeroen van der Ham, Eric ten Hove, Marko Verbeek

References

1. M. Dastani and F. de Boer and F. Dignum and W. van der Hoek and M. Kroese and J.-J. Ch. Meyer, Programming the Deliberation Cycle of Cognitive Robots, *Cognitive Robotics Workshop*, 2002.
2. K.V. Hindriks, F.S. de Boer, W. van der Hoek, and J.-J. Ch. Meyer, Agent programming in 3APL, *Autonomous Agents and Multi-Agent Systems*, 2(4):357–401, 1999.
3. M. Verbeek, 3APL as Programming Language for Cognitive Robots, Master's thesis, ICS, 2002.
4. M. Dastani, J. van der Ham, and F. Dignum, Communication for Goal Directed Agents, *AAMAS'02 Workshop on Agent Communication Languages and Conversation Policies*, Bologna Italy, 2002.

