Category: Artificial Intelligence

Intelligent Software Agents and Multi-Agent Systems

Milan Stankovic
University of Belgrade, Serbia

Uros Krcadinac
University of Belgrade, Serbia

Vitomir Kovanovic
University of Belgrade, Serbia

Jelena Jovanovic
University of Belgrade, Serbia

INTRODUCTION

Agent-based systems are one of the most important and exciting areas of research and development that emerged in information technology (IT) in the past two decades. In a nutshell, an agent is a computer program that is capable of performing a flexible, autonomous action in typically dynamic and unpredictable domains (Luck, McBurney, Shehory, & Willmott, 2005). Agents emerged as a response of the IT research community to the new data-processing requirements that traditional computing models and paradigms were increasingly incapable to deal with (e.g., the huge and ever-increasing quantities of available data). Many IT researchers believe that agents represent one of the most important software paradigms that have emerged since the object orientation.

From the historic point of view, the agent-oriented research and development (R&D) originates from different disciplines. Undoubtedly, the main contribution to the field of autonomous agents came from artificial intelligence (AI). Ultimately, AI is all about building intelligent artifacts and if these artifacts sense and act in some environment, then they can be considered agents (Russell & Norvig, 1995). Also, object-oriented programming (Booch, 2004), concurrent object-based systems (Agha, Wegner, & Yonezawa, 1993), and human-computer interaction (Maes, 1994) are fields that constantly drive forward the agent R&D in the last few decades.

In addition, the concept of an agent has become important in a diverse range of sub-disciplines of IT, including software engineering, computer networks, mobile systems, control systems, decision support, information retrieval and management, electronic commerce, and many others. Agents are being used in an increasingly wide variety of applications—ranging from comparatively small systems such as personalized email filters to large, complex, mission critical systems such as air-traffic control.

BACKGROUND

Even though there is no universal consensus over some key definitions in the field, it is intuitively clear what an "agent" is. One of the most widely used definitions states that "*an agent is a computer system, situated in some environment, that is capable of flexible autonomous action in order to meet its design objectives"* (Jennings, Sycara, & Wooldridge, 1998, p. 8).

There are three key concepts in this definition: *situatedness*, *autonomy*, and *flexibility*. *Situatedness* means that an agent receives sensory input from its environment and that it can perform actions which change the environment in some way. *Autonomy* is seen as the ability of an agent to act without the direct intervention of humans and that it has control over its own actions and internal state. In addition, the autonomy implies the capability of learning from experience. By *flexibility*, we mean the agent's ability to perceive its environment and respond to changes in a timely fashion; it should be able to exhibit opportunistic, goal-directed behaviour and take the initiative whenever appropriate. Also, an agent should be able to interact with other agents and humans, thus to be *social*. Some authors emphasize

the importance of the concept of *rationality*, which will be discussed in the next section.

Moreover, agent technologies can be considered from three perspectives (Luck, McBurney, Shehory, & Willmott, 2005):

- As a *design metaphor*, agents offer designers a way of structuring an application around autonomous, communicative elements;
- As a *source of technologies*, agent-based computing spans a range of specific techniques aimed at supporting interactions among entities in dynamic and open environments; and
- As a *simulation tool*, multi-agent systems offer robust models for representing complex and dynamic real-world environments, such as economies, societies and bio-systems.

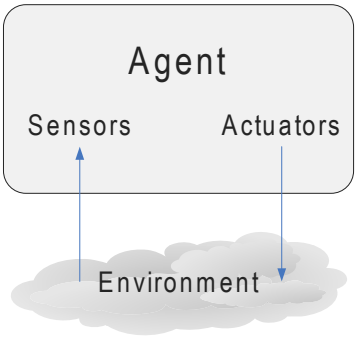
INTELLIGENT SOFTWARE AGENTS

Agents and Environments

Agents can be viewed as software entities that perceive their *environment* through *sensors* and act upon that environment through *actuators* (Russell & Norvig, 1995). There is an obvious analogy with a human agent who has ears, eyes, and other organs as sensors, and arms, legs, and other organs as actuators.

When we refer to an agent's perceptual inputs, we refer to the *agent's percepts*. An agent typically collects its percepts during the time, so its action in any moment generally depends on the whole sequence of percepts up to that moment. If we could generate a decision tree for every possible percept sequence of an agent, we could completely define the agent's behavior. Strictly speaking, we would say that we have defined the *agent function* that maps any sequence of percepts to the concrete action. The program that defines the agent function is called the *agent program*. These two concepts are different; the agent function is a formal description of the agent's behavior. The agent program is a concrete implementation of that formalism.

Figure 1. Agent and environment



As Russell and Norvig (1995) stipulate, one of the most desirable properties of an agent is its *rationality.* We say that an agent is rational if it always does the action that will cause the agent to be the most successful.

The rationality of an agent depends on:

- The performance measure that defines what is a good action and what is a bad action;
- The agent's knowledge about the environment;
- The agent's available actions;
- The agent's percept history.

One of the most cited definitions of a rational agent is:

for each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has. (Russell & Norvig, 1995, p. 36)

The main challenge in the field of intelligent software agents is to develop an agent program that implements the desired functionalities. Since it is a computer program, we need to have some computing device with appropriate sensors and actuators on which the agent program will run. We call this *agent architecture*. So an agent is essentially made of two components: the agent architecture and the agent program.

The Types of Agents

When we deal with the structure of an agent, we consider various implementation models for agent development. There are several basic types of agents with respect to their structure (Russell & Norvig, 1995).

The simplest kind of agents are the *simple reflex agents*. Such an agent only reacts to its current percept, totally ignoring its percept history. When a new percept is received, a rule that maps that percept to an action is fired. Such rules are known as *condition-action rules*.

Model-based reflex agents are more powerful agents, because they maintain some sort of internal state of the environment that depends on the percept history. For maintaining this sort of information, an agent must have two types of knowledge: (1) how the environment evolves, and (2) how its actions affect the environment.

Goal-based agents have some sort of goal information that describes desirable states of the world. Such an agent's decision making process is fundamentally different, because when a goal-based agent is considering performing an action it is asking itself "would this action make me happy?" along