# Chapter 5.2
# Multi-Agent Systems as Computational Organizations:
## The Gaia Methodology

**Franco Zambonelli**
*Università di Modena e Reggio Emilia, Italy*

**Nicholas R. Jennings**
*University of Southampton, UK*

**Michael Wooldridge**
*University of Liverpool, UK*

## ABSTRACT

The multi-agent system paradigm introduces a number of new design/development issues when compared with more traditional approaches to software development and calls for the adoption of new software engineering abstractions. To this end, in this chapter, we elaborate on the potential of analyzing and architecting complex multi-agent systems in terms of computational organizations. Specifically, we identify the appropriate organizational abstractions that are central to the analysis and design of such systems, discuss their role and importance, and show how such abstractions are exploited in the context of the Gaia methodology for multi-agent systems development.

## INTRODUCTION

In the past few years, *multi-agent systems* (MASs) have been recognized as an effective software engineering paradigm for designing and developing complex software systems (Jennings, 2001; Wooldridge & Jennings, 1995). In fact, the key characteristics of MASs (i.e., autonomy, situatedness, proactivity, and sociality [Wooldridge & Jennings, 1995]) are well suited to tackling the emerging complexities of modern software scenarios for a number of reasons. Firstly, the autonomy of the application components (i.e., the ability for an agent to decide what actions it should take at what time [Wooldridge & Jennings, 1995]) reflects the decentralized nature of modern distributed systems (Tennenhouse, 2000) and can be considered as the natural extension

to the notions of modularity and encapsulation for systems that are owned by different stakeholders (Parunak, 1997). Secondly, the flexible way in which agents operate (balancing reactive behaviour in response to the environment in which they are situated, with proactive behaviour towards the achievement of their design objectives [Wooldridge & Jennings, 1995]) is suited to the dynamic and unpredictable situations in which software is now expected to operate (Zambonelli, Jennings, Omicini, & Wooldridge, 2001). Finally, the high-level, dynamic, and social nature of multi-agent interactions is appropriate to open systems in which the constituent components and their interaction patterns constantly change (Estrin, Culler, Pister, & Sukjatme, 2002).

In this chapter, we firstly elaborate on the organizational metaphor and present and discuss the various organizational abstractions that come into play in the design and development of a complex MAS. In particular, we show that an organization is more than simply a collection of roles (as most methodologies assume) and that, in order to effectively build an MAS in organizational terms, a variety of additional organization-oriented abstractions needs to be devised and placed in the context of a methodology. In addition, we discuss how the presented organizational abstractions are effectively and coherently exploited in the Gaia methodology. Specifically, the Gaia methodology (fully described in Zambonelli, Jennings, & Wooldridge, 2003) both promotes a simple and clear to manage the development process and exploits the appropriate organizational abstractions in the process of developing MASs.

The chapter is organized as follows. The next section introduces the organizational metaphor and describes the organizational abstractions that are to be exploited in agent-oriented software engineering. Following that, we detail how these abstractions are exploited in Gaia in order to provide a methodology for the analysis and design of MASs. The next section clarifies these concepts with the use of a simple case study, followed by a discussion of the weaknesses and strengths of the Gaia methodology. The chapter ends with our conclusions.

## COMPUTATIONAL ORGANIZATIONS AND ORGANIZATIONAL ABSTRACTIONS

From here on, we take it as given that modern software scenarios suit a modeling approach based on autonomous, situated agents that interact in flexible ways (an argument already well developed in Chapter I and in the literature [Jennings, 2001; Zambonelli et al., 2003]). Given this position, the next challenge is to understand which further abstractions complete the agent-oriented mindset and may lead to the definition of a useful methodology for agent-oriented software development.

### Characterization and Motivations

In recent years, researchers in the area of MASs have proposed a number of different approaches for modeling systems based on different metaphors, none of which can reasonably claim to be general purpose. For instance: the ant algorithms metaphor (Bonabeau, Dorigo, & Theraulaz, 1999) has shown to be useful in efficiently solving complex distributed problems such as routing and distributed sorting; physical metaphors (Abelson et al., 2000, Mamei, Zambonelli, & Leonardi, 2003), focusing on the spontaneous reshaping of a system's structure, may have useful applications in pervasive and mobile computing; and societal metaphors have been effectively applied in robotics applications (Collinot, Drogoul, & Benhamou, 1996; Moses & Tennenholtz, 1995) and in the understanding and control of highly-decentralized systems (Hattori, Ohguro, Yokoo, Matsubara, & Yoshida, 1999).

Our approach focuses on the development of medium- to large-size systems, possibly situated