# Chapter 1.5
# Agent-Oriented Methodologies:
## An Introduction

**Marie-Luce Bourguet**
*Queen Mary University of London, UK*

## ABSTRACT

As an introduction to agent-oriented (AO) methodologies, we first describe the characteristics of both agents and multi-agent systems (MASs). This leads to a discussion of what makes an AO methodology that can be used to build an MAS. Finally, we briefly introduce the ten methodologies that are described in the remaining chapters in this book.

## INTRODUCTION

A methodology aims to prescribe all the elements necessary for the development of a software system, especially in the context of commercial applications. Prior to industry adoption, however, it is necessary for researchers to create that methodology. This has led to academic and industry researchers creating a large number of methodological approaches. A decade ago, there were estimated to be over a thousand methodological approaches to software development (Jayaratna, 1994), although these can be grouped into a much smaller number (around five) of software development *approaches* (Iivari, Hirschheim, & Klein, 1999). To these can be added a sixth: agent-oriented (AO) methodologies; that is, methodological approaches suitable for the development of agent-oriented or agent-based software.[1]

In parallel to the growth and availability of object-oriented (OO) systems development methodologies in the nineties, we are now seeing the burgeoning of a number of innovative AO methodologies, several of which form the core of this book. However, in contrast to OO methodologies, the field is not industry-driven—most AO methodologies are supported by small teams of academic researchers. Based on an observation that the coalescence of groups of OO methodologies in the late 1990s led to an increased take-up by industry of the object-oriented paradigm for system development and project management, this book aims to encourage first the coalescence and collaboration between research groups and then, hopefully, more rapid industry adoption of AO methodological approaches. In other words, most AO methodologies are (at the time of writing) in an early stage and still in the first context of mostly "academic" methodologies for agent-oriented systems development, albeit that many of these methodologies have been tested in small, industrial applications. One purpose of this book is to identify those predominant and tested AO methodologies, characterize them, analyse them, and seek some method of unification and consolidation with the hope that, in so doing, the community of scholars supporting AO methodologies will soon be able to transfer those innovative ideas into industry acceptance. This means mimicking the OO transition curve by seeking consolidation. One means of such consolidation is discussed in the last chapter of the book: the use of a method engineering framework (e.g., Martin & Odell, 1995) to create a repository of agent-oriented method fragments.

## AGENTS AND MULTI-AGENT SYSTEMS

Defining agents is not straightforward. There are many opinions, some of which you will see reflected in later chapters of this book (see also discussions in, for example, Luck, Ashri & D'Inverno, 2004). The key characteristics of agents are widely understood to be highly autonomous, proactive, situated, and directed software entities. Other characteristics such as mobility are optional and create a special subtype of agent; whereas some characteristics cannot be used as determining factors since they are really grey shades of a scale that encompasses both objects and agents.

In this context, an autonomous agent is one that is totally independent and can decide its own behaviour, particularly how it will respond to incoming communications from other agents. A proactive agent is one that can act without any external prompts. However, it should be noted that this introduces some problems, since there is also a large literature on purely *reactive* agents that would not be classified as agents with this categorization. Although reactive agents dominate in some domains, in reality, most agents being designed today have both proactive and reactive behaviour. Balancing the two is the key challenge for designers of agent-oriented software systems.

The characteristic of situatedness means that agents are contained totally within some specific environment. They are able to perceive this environment, be acted upon by the environment, and, in turn, affect the environment. Finally, the directedness means that agents possess some well-defined goal and their behaviour is seen as being directed towards effecting or achieving that goal.

Comparison with objects is often made. Some see agents as "clever objects" or "objects that can say no." This means that a hybrid agent+object system is entirely feasible. Others see agents at a much higher level of abstraction (e.g., Milgrom et al., 2001), much in the same way that OO specialists view components at a similar more granular level. Indeed, it is still unresolved as to how the scale of objects, components, and agents are matched and to what extent hybrid object/component/agent systems are feasible.

Some consequences of these high-level definitions are that agents participate in decision-making cycles, sometimes labelled as "perceive-decide-act" cycles. To achieve this, we have to consider other lower-level characteristics such as the roles that agents play, the metaphor of the agents having a mental state, including the possession of skills and responsibilities, aptitudes, and capabilities. When considering their interactions via perceptions and actions with other agents and the environment, we introduce notions of perceptions, actions, and agent communication languages. Negotiating skills involve the consideration of contract nets, auction strategies, and the issues of competition versus cooperation.