Chapter 73
Agent-Based Software Engineering, Paradigm Shift, or Research Program Evolution

Yves Wautelet
KU Leuven, Belgium

Christophe Schinckus
Royal Melbourne Institute of Technology, Australia

Manuel Kolp
Université catholique de Lovuain, Belgium

ABSTRACT

Information systems are deeply linked to human activities. Unfortunately, development methodologies have been traditionally inspired by programming concepts and not by organizational and human ones. This leads to ontological and semantic gaps between the systems and their environments. The adoption of agent orientation and multi-agent systems (MAS) helps to reduce these gaps by offering modeling tools based on organizational concepts (actors, agents, goals, objectives, responsibilities, social dependencies, etc.) as fundamentals to conceive systems through all the development process. Moreover, software development is becoming increasingly complex. Stakeholders' expectations are growing higher while the development agendas have to be as short as possible. Project managers, business analysts, and software developers need adequate processes and models to specify the organizational context, capture requirements, and build efficient and flexible systems.

INTRODUCTION

Information systems are deeply linked to human activities. Unfortunately, development methodologies have been traditionally inspired by programming concepts and not by organizational and human ones. This leads to ontological and semantic gaps between the systems and their environments. The adoption

DOI: 10.4018/978-1-7998-3016-0.ch073

of agent orientation and Multi-Agent Systems (MAS) helps to reduce these gaps by offering modeling tools based on organizational concepts (actors, agents, goals, objectives, responsibilities, social dependencies, etc.) as fundamentals to conceive systems through all the development process. Moreover, software development is becoming increasingly complex. Stakeholders' expectations are growing higher while the development agendas have to be as short as possible. Project managers, business analysts and software developers need adequate processes and models to specify the organizational context, capture requirements and build efficient and flexible systems.

We propose, in this paper, a modern epistemological validation of the emergence of Object-Orientation (OO) and Agent-Orientation (AO). The latter will be put into perspective through the Lakatosian approach. Related work and contributions to the epistemological position of OO and AO will first be explicated. The emerging context of the conceptual frameworks of OO and AO is then briefly described. The validation of our epistemological reading will be done on the basis of OO and AO operationalization of some critical theoretical concepts derived from the Kuhnian and Lakatosian theories. We finally discuss the adoption of the Lakatosian research programme concept to characterize both OO and AO. Implications of this epistemological position on everyday work have been distinguished both for software engineering researchers and practitioners. For researchers, it mostly has an implication on how agent ontologies are built and for practitioners it has an implication on how software problems are envisaged.

This paper is organized as follows. Section 2 presents the contributions as well as the research context. We point out the emergence of OO and AO. Section 3 focuses on our epistemological approach: AO is successively considered as a paradigm and a research programme. On the basis of how some relevant concepts of the Kuhnian and Lakatosian frameworks are operationalized by OO and AO, we provide a Lakatosian reading of those modeling concepts. Conclusions are summarized in Section 4.

STATE OF THE ART

This section presents the contributions of an epistemological reading for the computer science researcher.

Related Work and Contributions

Basili (1992) defines Software Engineering (SE) as "the disciplined development and evolution of software systems based upon a set of principles, technologies and processes". These theoretical frameworks are expected to solve practical problems by proposing software solutions. SE is a practice-oriented field (where empiricism often plays an important role) and constantly evolving; however, one must dispose of a framework to build common (and preferably best) practices improvement. Kaisler (2005) points out that "We develop more experience, we not only continue to learn new practices, but we refine and hone the practices that we have already learned". SE is the genuine discipline that emerged from this interconnection between practices and software solutions. Today's software development has become a very complex task and no one has the required skills or time to resolve a sophisticated problem on his or her own. Software development phases need the input from lots of people having to use concepts and ideas for which they share a common understanding. This can be referred as SE's key role: providing some common theoretical entities to allow specialists to develop software solutions.

Few papers in specialized literature point to an in depth questioning of SE knowledge evolution. As (Kaisler, 2005) emphasizes, the literature is mainly technical or practical and focused on the software