

## Towards an Agent-Oriented Paradigm

Jorge J. Gomez-Sanz\*, Fabien Michel†, Eric Platon‡, Alessandro Ricci§

In this position statement, software paradigms are understood as fundamental styles of programming or engineering regarding how solutions to problems are to be formulated in terms of fundamental abstractions. The creation of a new paradigm is mainly motivated by the aim to build better software that can address increasingly complex requirements. One paradigm can be better than another for solving a concrete problem in different ways. For instance, it may be easier to understand, or cheaper to produce/maintain, or more robust.

We argue that a main step essential to advance the research in SE and MAS and, in particular, for a widespread adoption of AOSE in both industrial and academic contexts, concerns the identification and development of an *agent-oriented paradigm*, focussing on the theory and practice of using agents for software development. This paradigm is to be contrasted with mainstream paradigms, such as OO, and its advantages and limitations evidenced.

Agent research elaborates numerous facts and results, but we argue that there is no paradigm yet, well recognised and accepted by SE and computing programming communities, and this hampers progress in SE with agents. A simple evidence for this is given by the very low number of references to AOSE, agent-oriented programming and agent research in general that can be found in mainstream SE and computer programming literature, including papers, surveys, books, textbooks. Conversely, the notion of agent is everywhere in AI and related literature (Russell and Norvig's book is a main example).

When a paradigm is adopted, different developers produce similar solutions for the same problem. Despite the progress in AOSE, this is something that it not happening within our community. So, the question now is why there is no consensus in the use of agents. We think this is the result of several factors:

---

\*Departamento de Sistemas Informáticos y Programación, Universidad Complutense de Madrid, 28040 Madrid, España. Email: [jjgomez@sip.ucm.es](mailto:jjgomez@sip.ucm.es)

†CRESTIC / LERI, IUT de Reims-Châlons-CharlevilleRue des crayères BP 1035, 51 687 Reims CEDEX 2, France. Email: [fabien.michel@univ-reims.fr](mailto:fabien.michel@univ-reims.fr)

‡National Institute of Informatics, Honiden Laboratory, 2-1-2 Hitotsubashi, Chiyoda-ku, 101-8430 Tokyo, Japan. Email: [platon@nii.ac.jp](mailto:platon@nii.ac.jp)

§DEIS, Alma Mater Studiorum Università di Bologna, Via Venezia 52, 47023 Cesena (FC), Italy. Email: [a.ricci@unibo.it](mailto:a.ricci@unibo.it)

- Too little concern with theory and practice of computer programming and SE, compared to a strong concern with AI. Agents could find a future also outside (Distributed/Classic) AI, if a focus was put on how to use agents to effectively build software systems, in particular complex software systems. There are precedents of a similar evolution in the LISP and Prolog programming languages, which led to the functional and logic paradigms.
- An overly strong emphasis on theory. Even though there is an important effort to change this situation, implementation and deployment is still considered (too) secondary to (pure) theoretical results.
- Not illustrating sufficiently the value of application of agent technology to problems; failure of disseminating evidence for improvements and cost savings obtained. Agent technology needs to be assessed and audited, and to do so, there must exist public benchmarks and code illustrating good practice. These examples should regard real-world problems.
- Weak involvement of industrial players. Differently from what happened for other paradigms (e.g., object and service), the agent community seems not having stimulated the participation and, often, the interest of industrial players in the effort of making the agent paradigm a valuable addition to the current programming and engineering practices.

Devising an agent paradigm means injecting in the land of SE and programming theory/practice a new abstraction layer based on first-class agent concepts for organizing and programming applications and systems, and *making it clearly acknowledged by mainstream SE and computer programming communities and literature*. This implies defining suitable programming languages, and methodologies based on shared and accepted computational models and theories. And these elements need to be based on the experience of the development of systems exhibiting, for instance, adaptability, robustness, scalability, and autonomy. Hence, it will be necessary to gather and organize existing knowledge on the construction of MAS and promote the dissemination of this knowledge.

## Acknowledgments

Many people contributed directly or indirectly to this position statement - which actually summarizes the discussion developed in a forum on agent-oriented computing. In particular we would like to thank: Paolo Petta, Michael Luck, Brian Henderson-Sellers, Paolo Giorgini, Yves Demazeaux, James Odell, Juan Pavon, Amal El Fallah Seghrouchni, Peter McBurney, Van H. Dyke Parunak, Alexis Drogoul, Massimo Cossentino.