AGERE! (Actors and aGEnts REloaded)

SPLASH 2011 Workshop on Programming Systems, Languages and Applications based on Actors, Agents, and Decentralized Control

Gul Agha
University of Illinois at Urbana-Champaign, USA
agha@cs.uiuc.edu

Rafael H. Bordini
Institute of Informatics, Federal University of Rio Grande do Sul, Brazil
R.Bordini@inf.ufrgs.br

Alessandro Ricci
University of Bologna, Italy
a.ricci@unibo.it

Abstract

The fundamental turn of software into concurrency and distribution is not only a matter of performance, but also of appropriate design and abstraction. This calls for programming paradigms that would allow developers to think, design, develop, execute, debug, and profile programs exhibiting different degrees of concurrency, reactiveness, autonomy, decentralization of control, and distribution in ways that are more natural than that supported the current paradigms. This workshop aims at exploring programming approaches explicitly providing a level of abstraction that promotes a *decentralized mindset* in solving problems and programming systems exhibiting such features. To this end, the abstractions of *actors* and *agents* (and systems of actors / systems of agents) are taken as a natural reference: the objective of the workshop is then to foster the research in all aspects of *actor-oriented programming* and *agent-oriented programming* and other *decentralized approaches* as evolution of mainstream paradigms (such as OOP), including the theory and the practice of design and programming, bringing together researchers working on the models, languages, and technologies, as well as practitioners developing real-world systems and applications.

Categories and Subject Descriptors D.1 [*Programming Techniques*]; D.2 [*Programming Languages*]; D.3 [*Software Engineering*]; I.2.5 [*Artificial Intelligence*]: Programming Languages and Software; I.2.11 [*Artificial Intelligence*]: Distributed Artificial Intelligence

Keywords agent-oriented programming, actor-oriented programming

1. Main Theme and Goals

The fundamental turn of software into concurrency, interactivity, distribution is not only a matter of performance, but also design and abstraction [19]. "The free lunch is over" calls for the devising of new programming paradigms — whether they are evolutions of existing ones or not — that would allow programmers to naturally think, design, develop, execute, debug and profile programs exhibiting different degrees of concurrency, reactiveness, autonomy, decentralization of control, distribution. Almost any application today includes the need of programming software components that actively — pro-actively and re-actively — do concurrently some jobs, react to various kind of events, communicate with each other by means of some interaction model. How to properly program such entities and systems of entities, what kinds of programming abstractions can help in systematically structuring complex reactive and proactive behavior, what kinds of programming abstractions can be effective in organizing applications as ensembles of relatively autonomous entities working together, and many other related issues are important open research questions.

The focus of this workshop is to investigate the definition of suitable levels of abstraction, programming languages, and platforms to support and promote a *decentralized mindset* [15] in solving problems, designing systems, programming applications, including the teaching of computer programming. That is, the question is how to think about problems and programs taking decentralization of control and interaction as the most essential features. To this end, we start from *agents* (and multi-agent systems) and *actors*, which can be recognized as two main broad families of concepts, abstractions and programming tools described in literature that explicitly promote such a decentralized thinking — even if assuming different facets depending on the context in which they are discussed, being it concurrent programming or distributed artificial intelligence. Accordingly, in this workshop we aim at promoting the discussion about agent-oriented and actor-oriented programming languages (models, theories, applications, systems), so as to explore agents and actors as a general-purpose computing paradigm explicitly supporting a decentralized mindset in solving problems and computer programming. Although we start from these two well known approaches, the workshop aims to promote discussion of any other approaches that also propose to contribute to the essential aspects of autonomous behavior and decentralized control. Any stage of software development is interesting for the workshop, including requirements, modeling, prototyping, design, implementation, testing, and any other means of producing running software based on actors and agents as first-class abstractions.

Overall, the workshop aims at fostering the development of the research in agent and actor oriented programming in the same vein that OOPSLA did for OOP at the beginning of the 80's. We hope to promote the investigation of all the features that would make agent-oriented or actor-oriented programming languages effective tools for developing software systems, as an evolution of the OO paradigm. Including aspects that concern both the *theory* and the *practice* of design and programming using such paradigms, so as to bring together researchers working on the models, languages, and technologies, and practitioners using such technologies to develop real-world systems and applications.

This overall perspective — which is oriented to impact on mainstream programming paradigms and software development — is

SPLASH'11 Companion, October 22–27, 2011, Portland, Oregon, USA.
ACM 978-1-4503-0940-0/11/10.

what distinguishes this event from related venues about agents and actors, organized in different contexts. Nevertheless, the event aims at being a good forum for collecting, discussing, and confronting related research that typically appears in different communities in the context of (distributed) artificial intelligence, distributed computing, computer programming and software engineering. Examples include: research on agent oriented programming and multi-agent programming [5, 6], either rooted in distributed artificial intelligence [7, 9, 12, 18] or computer programming contexts [14, 16, 17, 20]; research on actor-oriented programming [1, 11], including well-known programming languages/systems providing directly or indirectly support for actor oriented programming: examples are Erlang [4], Scala [10], Axum [21]; research on concurrent object-oriented programming [2, 3, 8], and on the extension of OO programming languages towards actor or agent-like levels of abstraction; research on new programming paradigms and reinvention of programming [13].

References

[1] G. Agha. *Actors: a model of concurrent computation in distributed systems*. MIT Press, Cambridge, MA, USA, 1986.

[2] G. Agha. Concurrent object-oriented programming. *Commun. ACM*, 33:125–141, September 1990.

[3] G. Agha, P. Wegner, and A. Yonezawa, editors. *Research directions in concurrent object-oriented programming*. MIT Press, Cambridge, MA, USA, 1993.

[4] J. Armstrong. Erlang. *Commun. ACM*, 53(9):68–75, 2010.

[5] R. Bordini, M. Dastani, J. Dix, and A. El Fallah Seghrouchni, editors. *Multi-Agent Programming Languages, Platforms and Applications - Vol. 1*. Springer, 2005.

[6] R. Bordini, M. Dastani, J. Dix, and A. El Fallah Seghrouchni, editors. *Multi-Agent Programming Languages, Platforms and Applications - Vol. 2*. Springer, 2009.

[7] R. Bordini, J. Hübner, and M. Wooldridge. *Programming Multi-Agent Systems in AgentSpeak Using Jason*. John Wiley & Sons, Ltd, 2007.

[8] J.-P. Briot, R. Guerraoui, and K.-P. Lohr. Concurrency and distribution in object-oriented programming. *ACM Comput. Surv.*, 30(3):291–329, 1998.

[9] M. Dastani. 2apl: a practical agent programming language. *Autonomous Agents and Multi-Agent Systems*, 16(3):214–248, 2008.

[10] P. Haller and M. Odersky. Scala actors: Unifying thread-based and event-based programming. *Theoretical Computer Science*, 2008.

[11] C. Hewitt. Viewing control structures as patterns of passing messages. *Artif. Intell.*, 8(3):323–364, 1977.

[12] K. V. Hindriks. Programming rational agents in GOAL. In Bordini et al. [6], pages 3–37.

[13] A. Kay. Programming and programming languages, 2010. VPRI Research Note RN-2010-001.

[14] J. J. Odell. Objects and agents compared. *Journal of Object Technology*, 1(1):41–53, 2002.

[15] M. Resnick. *Turtles, Termites and Traffic Jams. Explorations in Massively Parallel Microworlds*. MIT Press, 1994.

[16] A. Ricci and A. Santi. Agent-oriented computing: Agents as a paradigm for computer programming and software development. In *Proc. of the 3rd Int. Conf. on Future Computational Technologies and Applications – Future Computing 2011*, Rome, Italy, 2011. IARIA.

[17] A. Ricci, M. Viroli, and G. Piancastelli. simpA: An agent-oriented approach for programming concurrent applications on top of java. *Science of Computer Programming*, 76(1):37 – 62, 2011.

[18] Y. Shoham. Agent-oriented programming. *Artificial Intelligence*, 60(1):51–92, 1993.

[19] H. Sutter and J. Larus. Software and the concurrency revolution. *ACM Queue: Tomorrow's Computing Today*, 3(7):54–62, Sept. 2005.

[20] M. D. Travers. *Programming with Agents: New metaphors for thinking about computation*. Massachusetts Institute of Technology, 1996. PhD Thesis.

[21] Axum project, 2011. `http://msdn.microsoft.com/en-us/devlabs/dd795202`.