Workshop Preview of the 2015 Workshop on Programming based on Actors, Agents, and Decentralized Control (AGERE! 2015)

Elisa Gonzalez Boix
Vrije Universiteit Brussel, Belgium
egonzale@vub.ac.be

Philipp Haller
KTH Royal Institute of Technology, Sweden
phaller@kth.se

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

Carlos Varela
Rensselaer Polytechnic Institute, USA
cvarela@cs.rpi.edu

Abstract

The AGERE! workshop focuses on programming systems, languages and applications based on actors, active/concurrent objects, agents and – more generally – high-level programming paradigms promoting a mindset of decentralized control in solving problems and developing software. The workshop is designed to cover both the theory and the practice of design and programming, bringing together researchers working on models, languages and technologies, and practitioners developing real-world systems and applications.

Categories and Subject Descriptors D.1.3 [*Programming Techniques*]: Concurrent Programming; D.2 [*Software Engineering*]; D.3 [*Programming Languages*]

Keywords actors, agent-oriented programming, asynchronous programming, concurrent programming, event-driven programming, decentralized control

1. Concurrent and Decentralized Thinking

Nowadays concurrency is part of every-day programming, including aspects that are directly or indirectly related, such as asynchronous/event-driven/reactive programming and distributed programming. On the one hand, this calls for introducing fine-grained mechanisms, libraries and frameworks that make it possible to harness the power of concurrency in mainstream programming languages, and finally ease concurrent programming, which is a notoriously difficult task. On the other hand, it is not only a matter of performance, but also of conceptual modeling and design, devising proper *abstractions* that allow for developing modular, extensible, reusable concurrent/distributed/reactive programs. Like never before, we need to investigate programming paradigms – either novel ones or those which have evolved from existing ones – that allow one to naturally *think concurrent*, and provide abstractions for modeling and programming a concurrent and distributed world.

For this purpose – following the successful path set in previous editions (2011, 2012, 2013, 2014) – AGERE![1] aims to be the premier forum to explore these issues by drawing on and contributing to work on actors/active/concurrent objects and agents – as well as any programming paradigm embracing a *decentralized mindset* [17] in solving problems and designing systems.

2. Actors, Agents and High-Level Abstractions

On the one hand, actors and object-oriented concurrent programming [1–3, 7, 11, 13–15, 19, 22, 23] unify object-oriented programming with concurrency, providing a clean and powerful computation model which is being increasingly adopted in languages, frameworks and libraries used in the mainstream [8]. On the other hand, agents and agent-oriented programming [4–6, 12, 16, 18, 20] – even if developed in the AI and distributed AI contexts – provide a rich abstraction layer that could be effective for tackling the main complexities that concern the development of complex concurrent programs, featuring degrees of autonomy, proactivity, and reactivity. The objective of the workshop is to

SPLASH Companion'15, October 25–30, 2015, Pittsburgh, PA, USA
ACM. 978-1-4503-3722-9/15/10
http://dx.doi.org/10.1145/2814189.2833203

[1] *ago, agis, egi, actum,* **agere**—latin verb meaning to act, to lead, to do, common root for actors and agents

promote the investigation of all features that would make actor-based and agent-based programming approaches effective general-purpose tools for developing software systems as an evolution of the OO paradigm. Additionally, the workshop is meant to be a venue for all those approaches and paradigms that embrace concurrency in thinking and programming, providing proper abstractions for tackling important concerns, e.g., asynchronous programming, event-driven programming and reactive systems [9, 10].

In particular, a relevant issue today – especially in practice [8, 21] – concerns the adoption of actors/agents/etc. together with other existing concurrency models, as well as with mainstream programming models and languages. Thus, one of the main objectives of the workshop this year is to investigate how to unify actors/agents and object-oriented programming. Moreover, adoption of actors and agents on a large scale requires integration with a variety of runtime platforms, such as JavaScript execution engines, native platforms, and heterogeneous many-core machines. Several contributions this year investigate this important aspect.

3. Actors and Agents as a Software Development Paradigm

All stages of software development are considered interesting for the workshop, including requirements, modeling, formalization, prototyping, design, implementation, tooling, testing, and any other means of producing running software based on actors and agents as first-class abstractions. The scope of the workshop includes aspects that concern both the theory and the practice of design and programming using such paradigms, so as to bring together researchers working on models, languages and technologies, as well as practitioners using such technologies to develop real-world systems and applications.

References

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

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

[3] G. A. Agha, I. A. Mason, S. F. Smith, and C. L. Talcott. A foundation for actor computation. *J. Funct. Program.*, 7(1): 1–72, 1997.

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

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

[6] R. H. Bordini, M. Dastani, J. Dix, and A. El Fallah Seghrouchni. Special issue on multi-agent programming. *Autonomous Agents and Multi-Agent Systems*, 23 (2), 2011.

[7] P. Haller and M. Odersky. Scala actors: Unifying thread-based and event-based programming. *Theoretical Computer Science*, 410(2):202–220, 2009.

[8] P. Haller. On the integration of the actor model in mainstream technologies: the Scala perspective. In *AGERE! 2012*, pages 1–6. ACM, 2012.

[9] D. Harel and A. Pnueli. On the development of reactive systems. Springer, 1985.

[10] D. Harel, A. Marron, and G. Weiss. Behavioral programming. *Commun. ACM*, 55(7):90–100, 2012.

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

[12] N. R. Jennings. An agent-based approach for building complex software systems. *Commun. ACM*, 44(4):35–41, 2001.

[13] E. B. Johnsen and O. Owe. An asynchronous communication model for distributed concurrent objects. *Software & Systems Modeling*, 6(1):39–58, 2007.

[14] E. B. Johnsen, R. Hähnle, J. Schäfer, R. Schlatte, and M. Steffen. ABS: A core language for abstract behavioral specification. In *Formal Methods for Components and Objects*, pages 142–164. Springer, 2012.

[15] M. Miller, E. Tribble, and J. Shapiro. Concurrency among strangers: programming in E as plan coordination. In *Trustworthy Global Computing*, pages 195–229. Springer, 2005.

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

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

[18] A. Ricci and A. Santi. Designing a general-purpose programming language based on agent-oriented abstractions: The simpal project. In *Proceedings of the Compilation of the Co-located Workshops*, SPLASH '11 Workshops, pages 159–170. ACM, 2011.

[19] J. Schäfer and A. Poetzsch-Heffter. JCoBox: generalizing active objects to concurrent components. In *ECOOP 2010*, pages 275–299. Springer, 2010.

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

[21] S. Tasharofi, P. Dinges, and R. Johnson. Why do Scala developers mix the actor model with other concurrency models? In *ECOOP 2013*, pages 302–326. Springer, 2013.

[22] T. Van Cutsem, S. Mostinckx, E. Gonzalez Boix, J. Dedecker, and W. De Meuter. AmbientTalk: Object-oriented event-driven programming in mobile ad hoc networks. In *SCCC 2007*, pages 3–12. IEEE, 2007.

[23] A. Yonezawa and M. Tokoro. *Object-oriented concurrent programming*. MIT Press, 1987.