Coordination for Situated MAS: Towards an Event-driven Architecture

Andrea Omicini and Stefano Mariani

Dipartimento di Informatica–Scienza e Ingegneria (DISI)
Alma Mater Studiorum–Università di Bologna, Italy
{andrea.omicini, s.mariani}@unibo.it

Abstract Complex software systems modelled as multi-agent systems (MAS) are characterised by activities that are generated either by agents, or by the environment in its most general acceptation—that is, environmental resources and the spatio-temporal fabric. Modelling and engineering complex multi-agent systems (MAS) – such as pervasive, adaptive, and situated MAS – requires then to properly handle diverse classes of events: agent operations, resource events, spatio-temporal situation.
In the following, first we devise out the requirements for a software architecture for an agent-based middleware based on boundary artefacts, then we sketch a concrete architecture based on the TuCSoN middleware for MAS coordination.

1 Motivation

Today's complex computational systems more and more require strict coupling with the environment: pervasive, adaptive, self-organising systems need to work as situated systems, able to react to relevant changes in the environment, and to possibly act over it appropriately and timely. Interaction with the *environment* is then one of the main issue in complex computational systems nowadays [1].

On the other hand, agent-oriented abstractions and technologies provide a solid ground for complex system modelling and engineering: in particular, meta-models like A&A [2], middlewares like CArtAgO [3], JADE [4], TuCSoN [5], agent-oriented methodologies like Gaia [6], PASSI [7] and SODA [8] already proved their effectiveness in dealing with the engineering of complex software systems [9]. The *reactive* nature of situated systems, however, does not cope well with the *proactive* nature of agency, at least not with no compromise: in particular, the event-driven computational model pushed by system situation does not match straightforwardly the typical high-level programming model of agent-oriented languages—in particular those for intelligent agents.

While such issues are typically faced with more articulated agent languages and architectures – like hybrid agents architectures –, their increasing complexity (in particular in size and number of components and events) mandates for principled solutions, possibly at system level rather than at single-component level. Accordingly, in the following we sketch an *event-driven architecture* for agent middleware exploiting coordination abstractions for event handling, discuss its abstract features, and describe a possible reification as a concrete architecture based on the TuCSoN middleware for multi-agent system (MAS) coordination.

2 MAS as Event-driven Systems

Situated systems have to deal with the environment as the main source of activity, as well as the foremost target for their own activity. Environment activity is typically modelled in terms of *events*, whose interaction with computational systems is articulated in a number of stages: at least, selection of potentially-relevant events, perception of selected events, delivering of perceived events to designed components, elaboration of events by components. Moreover, situatedness also means reactiveness to the spatio-temporal fabric: perceiving and reacting to events related to location and motion in space, and to the passage of time, are essential features of mobile and pervasive computing applications. In the overall, dealing with situatedness basically requires an *event-driven programming model*, along with a suitable choice of the representation language for environment events.

On the other hand, modelling a complex computational system as a MAS basically accounts to encapsulating system activities within agents. Whereas the notion of environment as a sort of external source of event is more or less easy to accept, the same does not hold for agents. However, agents in an open MAS are possibly not designed and controlled by the MAS designer: so, their activity should be in principle handled again as an unpredictable source of events: either for openness, or for the intrinsic complexity that an agent behaviour may in principle encapsulate. Accordingly, both organisation and security issues require modelling agents, too, as (possibly unpredictable) event sources within MAS, to be possibly handled via event-driven engineering techniques.

As a result, an event-driven view of MAS is possible, where agents and the environment are the sources of all activities, and the overall behaviour of the MAS is obtained by suitably modelling activities as events, and governing them through suitable event-driven models and technologies.

3 Boundary & Coordination Artefacts

Whereas agents and environment are the most suitable abstractions to handle activities in a MAS, artefacts – being reactive by definition – are the most suitable abstractions to encapsulate reactive behaviours—so, the most suitable way to handle events in a complex MAS, according to the A&A meta-model [2].

The first issue is to map activities of any sort – even possibly unpredictable ones – upon a set of *admissible events*—that is, those events that are accepted and handled by the MAS. Apart from an appropriate model, this requires suitably-defined architectural abstractions embedding such a mapping. This is in fact the role of *boundary artefacts*, which mediate between agents and the MAS, as well as between the MAS and its environment.

In particular, we envision a principled MAS architecture where each agent and each resource in the environment is associated to its own boundary artefact, working on the one hand as a proxy for the agent / resource within the MAS, on the other hand as a sort of interface for the agent / resource towards the MAS. Known examples of boundary artefacts are Agent Communication Contexts [10] and the abstractions of Law-Governed Linda [11].

However, once brought within a MAS by a boundary artefact, an admissible event has to be handled to possibly generate other events and / or computational activities, defining the overall behaviour of a MAS: for instance, to aggregate events from resources, like a bunch of sensors. This is the role of *coordination artefacts* [12], which capture admissible MAS events, and associate them to computational activities implementing coordination laws, possibly generating further events, and giving raise to *event chains*.

With respect to the classification of artefacts introduced by the A&A meta-model [13], *individual* and *resource artefacts* are basically represented here by boundary artefacts, whereas *social artefacts* play roughly the role of coordination artefacts, here. In principle, however, boundary artefacts have a much more limited function with respect to individual and resource artefacts, which are devoted also to contain the basic coordination policies related to individual agents and resources. Then, a more precise architectural mapping would require *individual coordination artefacts* to be associated to boundary artefacts in order to achieve the same sort of architectural functionality provided by A&A individual artefacts.

4 A Concrete Event-driven Architecture in TuCSoN

The abstract architecture sketched above essentially models complex MAS as composed of *proactive entities* (agents, environment resources, space-time fabric) and *reactive entities* (boundary and coordination artefacts), connected together by a net of co-ordinated events. Quite unsurprisingly, a possible reification of such an abstract architecture can be designed upon the TuCSoN middleware for MAS coordination [5].

First of all, it is quite easy to map coordination artefacts upon ReSpecT tuple centres [14], which are the coordination abstraction provided by TuCSoN. There, computational activities devoted to MAS coordination can be represented in terms of the ReSpecT logic-based specification language [15], allowing admissible events to be associated to *reactions*, possibly generating further events within a MAS.

Then, two middleware abstractions play the role of boundary artefacts in TuCSoN: *agent coordination contexts* (ACC) [16], for agents, and *transducers* [17], for resources. On the one hand, ACC play the role of security and organisation abstractions [18]: each agent has an associated ACC that mediates all the agent interactions with the TuCSoN system, working both as its representative within the TuCSoN-coordinated MAS, and as its interface towards the MAS itself, providing the agent with available operations. On the other hand, transducers [17] are in charge of representing individual resources, along with their own peculiar ways of interacting: each portion of the MAS environment represented by a resource is associated to its specific transducer, capable of two-way interaction to map meaningful resource events upon admissible MAS events.

Mapping our abstract event-driven architecture upon the TuCSoN middleware obviously mandates for a complete event driven model. In TuCSoN, this is achieved by *(i)* generalising the TuCSoN notion of admissible event, and *(ii)* extending ReSpecT as a full-fledge event-driven language, capable of dealing with general-purpose events, enabling ReSpecT tuple centres to work as event-driven abstractions for MAS coordination—as discussed in [19].

In order to test the effectiveness of the abstract architecture depicted above, as well as of the corresponding TuCSoN-based concrete architecture for event-driven engineering of complex MAS, experiments were conducted, by exploiting the TuCSoN technology in complex application scenarios. In particular, TuCSoN is currently adopted for the implementation of the Molecules of Knowledge (MoK for short) model for knowledge self-organisation [20], and for the testing of the SAPERE middleware for pervasive adaptive services [21].

The TuCSoN middleware is available as an open source project [22], and in its current stage of development features ACC in its main distribution. The most general notion of transducers (with transducer managers for middleware lifecycle) and the complete situated version of ReSpecT are instead currently under testing.

Acknowledgements

The authors would like to thank the organisers of PNSE'13 and ModBE'13 – and in particular Daniel Moldt – for inviting our contribution.

This work has been supported by the EU-FP7-FET Proactive project SAPERE – Self-Aware PERvasive service Ecosystems, under contract no. 256873.

References

1. Weyns, D., Omicini, A., Odell, J.J.: Environment as a first-class abstraction in multi-agent systems. Autonomous Agents and Multi-Agent Systems **14**(1) (February 2007) 5–30 Special Issue on Environments for Multi-agent Systems.
2. Omicini, A., Ricci, A., Viroli, M.: Artifacts in the A&A meta-model for multi-agent systems. Autonomous Agents and Multi-Agent Systems **17**(3) (December 2008) 432–456 Special Issue on Foundations, Advanced Topics and Industrial Perspectives of Multi-Agent Systems.
3. Ricci, A., Viroli, M., Omicini, A.: CArtAgO: A framework for prototyping artifact-based environments in MAS. In Weyns, D., Parunak, H.V.D., Michel, F., eds.: Environments for MultiAgent Systems III. Volume 4389 of LNAI. Springer (May 2007) 67–86 3rd International Workshop (E4MAS 2006), Hakodate, Japan, 8 May 2006. Selected Revised and Invited Papers.

4. Bellifemine, F.L., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE. Wiley (February 2007)
5. Omicini, A., Zambonelli, F.: Coordination for Internet application development. Autonomous Agents and Multi-Agent Systems **2**(3) (September 1999) 251–269 Special Issue: Coordination Mechanisms for Web Agents.
6. Zambonelli, F., Jennings, N.R., Wooldridge, M.J.: Developing multiagent systems: The Gaia methodology. ACM Transactions on Software Engeneering Methodologies **12**(3) (2003) 317–370
7. Cossentino, M., Gaglio, S., Sabatucci, L., Seidita, V.: The PASSI and agile PASSI MAS meta-models compared with a unifying proposal. In Pechoucek, M., Petta, P., Varga, L.Z., eds.: Multi-Agent Systems and Applications IV. Volume 3690 of LNCS., Springer (2005) 183–192 4th International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2005, Budapest, Hungary, September 15-17, 2005, Proceedings.
8. Molesini, A., Omicini, A., Denti, E., Ricci, A.: SODA: A roadmap to artefacts. In Dikenelli, O., Gleizes, M.P., Ricci, A., eds.: Engineering Societies in the Agents World VI. Volume 3963 of LNAI. Springer (June 2006) 49–62 6th International Workshop (ESAW 2005), Kuşadası, Aydın, Turkey, 26–28 October 2005. Revised, Selected & Invited Papers.
9. Zambonelli, F., Omicini, A.: Challenges and research directions in agent-oriented software engineering. Autonomous Agents and Multi-Agent Systems **9**(3) (November 2004) 253–283 Special Issue: Challenges for Agent-Based Computing.
10. Di Stefano, A., Santoro, C.: Modeling multi-agent communication contexts. In: 1st International Joint Conference on Autonomous Agents & Multiagent Systems (AAMAS 2002), Bologna, Italy, ACM Press (15–19 July 2002) 174–175 Proceedings.
11. Minsky, N.H., Leichter, J.: Law-Governed Linda as a coordination model. In Ciancarini, P., Nierstrasz, O., Yonezawa, A., eds.: Object-based Models and Languages for Concurrent Systems. Volume 924 of LNCS. Springer (1995) 125–146
12. Omicini, A., Ricci, A., Viroli, M., Castelfranchi, C., Tummolini, L.: Coordination artifacts: Environment-based coordination for intelligent agents. In Jennings, N.R., Sierra, C., Sonenberg, L., Tambe, M., eds.: 3rd international Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004). Volume 1., New York, USA, ACM (19–23 July 2004) 286–293
13. Omicini, A., Ricci, A., Viroli, M.: *Agens Faber*: Toward a theory of artefacts for MAS. Electronic Notes in Theoretical Computer Science **150**(3) (29 May 2006) 21–36 1st International Workshop "Coordination and Organization" (CoOrg 2005), COORDINATION 2005, Namur, Belgium, 22 April 2005. Proceedings.
14. Omicini, A., Denti, E.: From tuple spaces to tuple centres. Science of Computer Programming **41**(3) (November 2001) 277–294
15. Omicini, A.: Formal ReSpecT in the A&A perspective. Electronic Notes in Theoretical Computer Science **175**(2) (June 2007) 97–117 5th International Workshop on Foundations of Coordination Languages and Software Architectures (FOCLASA'06), CONCUR'06, Bonn, Germany, 31 August 2006. Post-proceedings.
16. Omicini, A.: Towards a notion of agent coordination context. In Marinescu, D.C., Lee, C., eds.: Process Coordination and Ubiquitous Computing. CRC Press, Boca Raton, FL, USA (October 2002) 187–200
17. Casadei, M., Omicini, A.: Situated tuple centres in ReSpecT. In Shin, S.Y., Ossowski, S., Menezes, R., Viroli, M., eds.: 24th Annual ACM Symposium on Applied Computing (SAC 2009). Volume III., Honolulu, Hawai'i, USA, ACM (8–12 March 2009) 1361–1368

18. Omicini, A., Ricci, A., Viroli, M.: Agent Coordination Contexts for the formal specification and enactment of coordination and security policies. Science of Computer Programming **63**(1) (November 2006) 88–107 Special Issue on Security Issues in Coordination Models, Languages, and Systems.
19. Mariani, S., Omicini, A.: Event-driven programming for situated MAS with ReSpecT tuple centres. In: Workshop on Agent Based Computing: From Model to Implementation X (ABC:MI 2013), Koblenz, Germany (September 2013) Proceedings.
20. Mariani, S., Omicini, A.: Molecules of Knowledge: Self-organisation in knowledge-intensive environments. In Fortino, G., Bădică, C., Malgeri, M., Unland, R., eds.: Intelligent Distributed Computing VI. Volume 446 of Studies in Computational Intelligence., Springer (2013) 17–22 6th International Symposium on Intelligent Distributed Computing (IDC 2012), Calabria, Italy, 24-26 September 2012. Proceedings.
21. Zambonelli, F., Castelli, G., Ferrari, L., Mamei, M., Rosi, A., Di Marzo Serugendo, G., Risoldi, M., Tchao, A.E., Dobson, S., Stevenson, G., Ye, Y., Nardini, E., Omicini, A., Montagna, S., Viroli, M., Ferscha, A., Maschek, S., Wally, B.: Self-aware pervasive service ecosystems. Procedia Computer Science **7** (December 2011) 197–199 Proceedings of the 2nd European Future Technologies Conference and Exhibition 2011 (FET 11).
22. TuCSoN: Home page. `http://tucson.apice.unibo.it`