Parallelising Multi-agent Systems for High Performance Computing

Paulo Leitão
Polytechnic Institute of Bragança
Bragança, Portugal

E-mail: pleitao@ipb.pt

Udo Inden
Cologne University of Applied Sciences
Köln, Germany

E-mail: udo.inden@fh-koeln.de

Claus-Peter Rückemann
Leibniz Universität Hannover / Westfälische Wilhelms-Universität Münster (WWU) / North-German Supercomputing Alliance (HLRN),
Münster, Hannover, Germany
E-mail: ruckema@uni-muenster.de

***Abstract*— Multi-Agent Systems (MAS) are seen as a promising technology to face the current requirements of large-scale distributed and complex systems, e.g., autonomous traffic systems or risk management. The application of MAS to such large scale systems, characterised by millions of distributed nodes, imposes special demanding requirements in terms of fast computation. The paper discusses the parallelisation of MAS solutions using larger-scale distributed High End Computing platforms as well as High Performance Computing as a suitable approach to handle the complexity associated to collaborative solutions for large-scale systems.**

Keywords- Multi-Agent Systems; High Performance Computing; Parallel systems; Risk Management.

I. INTRODUCTION

The dynamics of business environments and technological advancements are driving forces of an increasing demand for High-End and High- Performance Computing (HEC, HPC) for designing and operating highly customised and complex products and services like networked manufacturing systems (e.g., Nokia mobile phones), autonomous traffic systems (e.g., SESAR aviation), computational risk management (e.g., financial models), complex design (e.g., new aircrafts) and complex service operations (e.g., a large airport, harbour), oil and gas seismic imaging, climate and weather forecasting, bacteria modelling and estimated coverage of epidemics, considering huge population sizes. Therefore, more flexible, agile and reconfigurable systems are required, in opposite to the typical centralised solutions that are not enough to address the demanding requirements.

Multi-Agent Systems (MAS) [1], [2], [3] are a suitable approach to face this challenge, based on the decentralisation of functions by distributed nodes. The application of MAS principles to large scale systems (e.g., 6 million parts spanning across a supply-chain) imposes demanding requirements in terms of fast computation that significantly exceed any current MAS platform. The use of reliable, secure and “on demand” utilisation of high computing capacities to handle all or part of such complexity associated to large-scale systems within short periods of time is required.

High End Computing (HEC) and High Performance Computing (HPC) are widely perceived as only affordable to large corporations. However, a recent communication from the European Commission predicts that “*Exa-scale computers (machines capable of 1e+18 operations / second) will be in existence by 2020*” and suggests that “*97 % of industrial companies that employ HPC consider it indispensable for their ability to innovate, compete, and survive*” [4].

In this paper we discuss the parallelisation of MAS using larger-scale HEC or HPC platforms to handle the complexity associated to collaborative solutions for large-scale systems. The computational power of these platforms is useful for large-scale modelling and simulation of data and compute intensive applications like those mentioned above that often require the use of simulations because real-world testing would be too expensive, dangerous, environmentally damaging, or even physically impossible. For this purpose, agent-based simulation is a suitable approach, but its computing requirements can be high and demanding distributed and non-distributed highly performant computing platforms.

The rest of this paper is organised as follows: Section II overviews the MAS and HPC concepts and Section III discusses the combination of these technologies, namely the associated challenges and the existing approaches. Section IV presents some scenarios that benefit from the use of MAS solutions running in high end platforms. Section V describes the REPAST HPC tool, as an example of an agent-based modelling framework for large-scale distributed computing platforms. At last, Section VI rounds up the paper with the conclusions and points out the future work.

II. BACKGROUND ON MULTI-AGENT SYSTEMS AND HIGH PERFORMANCE COMPUTING

A. *Multi-agent Systems*

MAS are a computational paradigm derived from the distributed artificial intelligence field, characterised by the decentralisation and parallel execution of activities based on autonomous agents. Since agents have limited knowledge and skills, they need to interact, e.g., through negotiation, to achieve their individual goals, as illustrated in Figure 1. The high-level of autonomy and cooperation exhibited by such solutions, allow them to fast respond to perturbations. MAS can be used to solve problems that are difficult or impossible for a monolithic system to solve, offering models for representing complex and dynamic real-world environments [5]. Examples of application areas are electronic commerce, manufacturing, robotics and telecommunications.

ISBN: 978-1-61208-310-0

The development of multi-agent system solutions is strongly simplified if an agent development platform is used, taking advantage of the useful features and services provided, such as registry and management services. In some cases, they follow the specifications established by the Foundation for Intelligent Physical Agents (FIPA). Examples of such agent-development platforms are the Java Agent Development Framework (JADE), AGlobe and JACK. For example, JADE [6] is a Java based architecture that uses the Remote Method Invocation (RMI) to support the creation of distributed Java based applications. It is compliant with FIPA specifications, providing low programming effort and features to support the management of agent-based solutions, delivering an easy integration with other tools, namely Protégé and Java Expert System Shell (JESS).

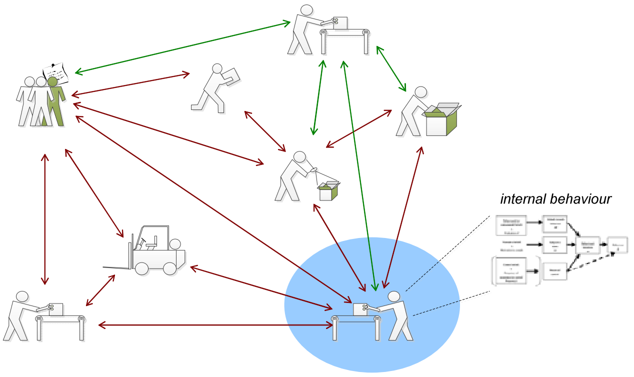

Figure 1. Example of a Multi-agent system.

Another level of using multi-agent systems principles is known as Agent-Based Modelling (ABM), which is a paradigm to create, analyse, experiment and simulate systems populated by cooperative agents. The dynamics of the simulation are specified as agent behaviour rules, and the purpose of the simulation is to reveal population-level structure that is the outcome of this individual-level behaviour. In particular, it supports the reproduction of a variety of patterns and complex phenomena observed in real world, such as evolution and self-organisation. Examples are the simulation of economic, biological and physical dynamics. Several ABM computational frameworks are currently available, such as Netlogo, Mason and Repast (for a comparative survey, see [7] and [8]). Applications of ABMs can be found in [9].

Note that there is a clear difference between multi-agent systems and agent-based simulation: MAS frameworks allow building agent-based systems, but they haven’t a simulation infrastructure (e.g., misses a scheduler and the notion of a "clock"). On contrary, ABM frameworks allow agent-based simulation but they have not the purpose of developing agent systems (nor FIPA compliant).

B. *High Performance Computing*

In the 21st century, highly performant computing, especially HEC and HPC, is a key enabling technology, with many countries world-wide investing huge sums of public money on related infrastructure, energy, and services. “High performance” is considered in its broadest definition covering demanding applications that are compute and / or data-centric, which can support the exploitation of parallelism on a large scale.

HEC systems can range from a desktop computer, through clusters of servers and data centres up to high-end custom supercomputers. Resources can be physically close to each other, e.g., in a highly performant compute cluster, or the compute power can be distributed on a large number of computers as with most Grid and Cloud computing concepts.

HPC systems are based on architectures with a large number of processors, for exploiting massive parallelism. Commonly used models are Massively Parallel Processing and Symmetric Multi-Processing, used with the concept of local islands. Due to physically shared memory usage and compute communication the physical architectures with these HPC systems are different.

Therefore, besides the compute power, the major challenges with the high end field of applications and economical resources usage are efficient communication and data locality. So, major requirements are the intelligent and autonomous distribution and scheduling of highly parallel and loosely coupled job tasks and processes as well as managing the automation of data distribution.

III. PARALLELISING MULTI-AGENT SYSTEMS OVER HIGH PERFORMANCE COMPUTING PLATFORMS

As previously described, HPC is nowadays a hot topic, with many high processing computing resources available through the organisation of grids or clusters of computers. The pertinent question arising in this context is how MAS can be integrated into HPC environments, allowing the development of solutions handling more efficiently complexity in large scale systems. This section discusses the challenges in this topic and particularly, the parallelisation of MAS solutions.

A. *Challenges, Requirements and Technical Issues*

The main issue related to this topic is to understand how to deploy societies of agents on a large scale computational platform. The challenges are mainly related to the parallelisation and HPC readiness of MAS applications.

Parallelising MAS applications means deploying and running these multiple intelligent agents on several computational resources nodes. As an example, if a simulation experiment plan defines 10000 simulations (e.g., each one carried out by one agent), and a computational Grid of 20 nodes is used, five hundred simulations can be assigned to each node (note that parallelising only one simulation is also possible by decomposing the simulation in independent agents). However, the distribution of the multi-agent system execution is a complex task, due to the technical particularities, and particularly, to the intensive communication among agents and the environment. Additionally, individual agents need to modify the environment during their decision-making processes, which in terms of parallelisation are translated to the need to share several layers of environmental data and agents, which is a complex process [10]. The access to these distributed nodes to collect and synchronise the state and actions of the agents, e.g., to support decision-making and monitoring processes, are also required.

Traditionally, HPC are placed in one site, being a challenge to combine several HPC sites that work in a transparent way for the MAS applications. Usually the agent-development platforms support the distribution within a single computer, for example using threads or simple shared-memory parallelism. For example, JADE agents are based in a behaviour structure that are implemented a simple threads.

Having these considerations in mind, important requirements and constraints for each HPC provider with respect to the MAS applications are related to:

- *Integration and storage of data*: Middleware for integrating flow of data from distributed, heterogeneous sources, and also the agent's endogenous data and the agent topologies.
- *Synchronisation of distributed data and behaviour*: HPC systems primarily synchronise the distribution of data and algorithms of massive parallel applications across massive parallel hardware and are efficient if processor primarily compute rather than communicate or wait for input from other applications.
- *Communication scale and latency*: ICT platform that ensure the scalability of many agents and the fulfilment of communication constraints, and particularly, time latency in the interaction between MAS nodes and also with legacy systems.
- *Security*: Infrastructure granting high-level security and privacy of data (probably implementing risk-aware capabilities as feature of intelligent applications).
- *Interaction and searching*: Proper mechanisms for the scale design of interactions among agents, e.g., negotiation protocols using Agent Communication Languages (ACL), and the searching and discovering features in large-scale systems.

In conclusion, these systems should exhibit a set of properties that include a time scheduler for coordinating and synchronising flow of agent-based events, and storage systems for endogenous data, such as agents' internal state information and agents' topologies, synchronised to form a single data storage framework.

At this stage, the question is if agent development frameworks, such as JADE, or ABM frameworks, such as Repast, already fulfil these requirements. Basically, agent-development frameworks let these issues open to the developer, being necessary to consider a proper architecture that allows the integration. The architecture may consider three layers, as illustrated in **Figure 2**.

The *lower part* is about the MAS based application, using an agent development framework or an ABM tool. At this application level, the agent-based model may integrate relevant sources of knowledge distributed across organisations, functions, space, but also time (different latency, etc.) and technically to be parallel computed on thread level. An example is a risk management application designed as a MAS solution and able of running on HPC resources (chiefly effectively distributed with multiple HPC islands).

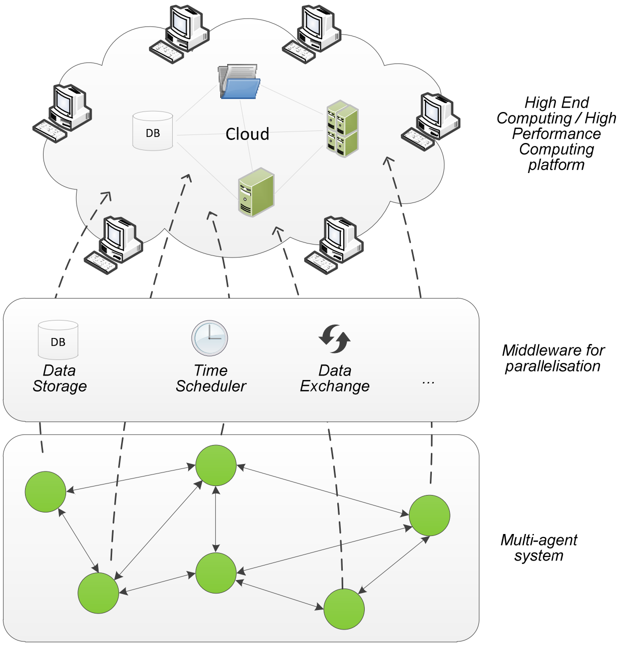

Figure 2. Architecture for an integrated MAS – HPC solution.

The *middle part* is related to the middleware that ensure the automatic parallelisation of the multi-agent system application, namely providing mechanisms for integration and storage of data, synchronisation of distributed data and behaviour and security of data.

The *upper part* is about the HPC/Cloud infrastructure. Access is provided by a service platform, including a "configurator" for adjusting demands of applications and HPC capacities, a "broker" system for access of configured applications to HPC capacity including commercial management functionality, an "application controller", e.g., for managing simulation, and "data exchange" functionality (which may include security systems).

The holonic principles may be considered to make more flexible the development of the multi-agent system applications, and also to reduce the communication across MAS nodes (note that holonic systems, following the Koestler observations, define intermediate stable control states [11]). This may simplify the achievement of the cross communication latency.

B. *Existing Approaches*

Multi-agent system technology is already being used in some HPC/Cloud projects as brokers for service search in the EU FP7 mOSIAC project [12] or as learning entities for energy efficiency in the EnerGy project [13].

However, the several available agent-development frameworks, as JADE is, are not completely aimed to run at extremely high-performance computing platforms, i.e., they are not HPC compliant. The same occur for the existing ABMs frameworks. Note that these frameworks can be used to distribute MAS applications across multiple computing resources, but this is a complex task that requires that developers codify by hand the HPC functionalities.

As explained by Dr. Michael North, deputy director of Argonne’s Center for Complex Adaptive Agent Systems Simulation, “*Before we can efficiently use high-performance computing, we need to do research to understand how to best use the architectures. Switching our model to run on 10,000 nodes instead of 10 is not just a linearly scalable process. Moving from a few interactions to a million interactions is not just moving to greater numbers. Entire interactions change. Also, we need to consider implementation issues such as partitioning of the name space, memory space, communicate on, and the effect of noise and jitter on these large platforms. Ultimately, we need to make changes to our model to deal with such issues.*” [14].

The advent of HPC had driven the extension of some of the ABM tools to support the deployment of agent-based models into HPC environments aiming to have simulation results in a fast way. Examples are Repast HPC [15], Flexible Large-Scale Agent Modelling Environment (FLAME) [16], Pandora [10] and GridABM [17]. These ABM frameworks support the design and execution of large-scale social simulations in HPC environments. Examples of services and mechanisms provided by these tools are the cross-process communication and synchronisation, parallel scheduling, data collection and random number generation.

The EPIS project provides a general and user-friendly approach to combine multi-agent based simulation and HPC platforms, based on the definition of workflows, allowing to deploy simulations on a cluster of computers without any prior parallelising work [18].

However, the same effort is missing for the agent development frameworks. Probably, the reason for the missing HPC extensions in the agent development frameworks, and only focused in ABM frameworks, is that HPC platforms are more useful for simulation purposes that requires more computational power, due to the typical huge population size and short time response.

IV. SCENARIOS

Though *swarms* of software agents imply parallel operations, the title of this paper indicates that processing and communication in typical MAS are at least not fully parallel organised. Thus, which kind of problems drives the need of porting technological swarm intelligence to the capacity and capability levels of typically parallel high-end computing?

The industrial use-cases motivating our work on parallel MAS are marked by large landscapes of interdependent risks to be managed. First ideas (example 1) to model such landscapes and to make them accessible for computing grew in the Cargo-Lifter project that aimed at the development of the technology, the production and the operations of a 265-metres airship for transporting heavy and oversized goods. The venture failed because of its overwhelming and, in key areas of technology and operations, ignored complexity [23].

In current industry-driven research, the concept of risk landscapes again became relevant: The ARUM project [24] (example 2) addresses problems of the ramp-up of production of small series, particularly in aviation industry. Factors like a high capital intensity of product and production development and specifics of small series lead to a “point of no return” where a decision is to be made to start production. After this decision has been made, immatureness and undependability of technology, of suppliers or production processes and resources are left to ramp-up management.

Problems in this phase delayed the market introduction of the Boeing 787 “Dreamliner” for about 3 years and returned in January 2013 when fire hazards of a new type of Lithium-Ion batteries gave reason to ground 50 operating aircrafts for, at the time this text is written, for about 4 months [25]. Comparing, failures delayed the production of the Airbus A380 [26]. A core problem of the case is that events may interact in hardly predictable ways. These events affect revenues, translate into early depreciations, drive costs by disrupting operations or by re-engineering, and finally penalties or a loss of reputation.

A third industrial case relies on work with scientists and industry experts on a risk model for a complex change program of another globally operating technology provider in aviation industry. In the next four years the firm will face a multi ramp-up scenario by exchanging old by new products and production technologies. A factory-wide “Readiness Review” delivered more than 2200 issues, each representing a risk identified by an expert and in further analysis melted down to about 1000 problems that, in further steps, are to be operationalized by respective control parameters and to be assigned to “cells” of managerial responsibility.

In the first example about 20 factors were identified influencing the incidence rate per risk and another 20 their impact. Yet under ramp-up conditions, both, event-risk and impact may be not known and the view is to be altered from handling material risk to understanding holistic capabilities of the system to adapt.

Particularly, in the latter approach the computational scale of the problem matters: In example 1, a MAS has been used to support the development of a common domain ontology across teams in product-, production and operations’ engineering and to integrate the knowledge from more than 60 large industrial lead-users worldwide. Yet the model just captured about 70 of about 1400 risks that had been identified in a high level analysis (comparing to example 3).

In ARUM semantic technologies will improve response to unplanned events as they occur by identifying, structuring and exploiting discretions to act in a scene and, in tactical planning, by enhancing capabilities to adapt to unplanned events. In order to demonstrate its value ARUM will focus exemplary cases and test scenarios like the scene of a work-station or of the production of a module of an aircraft. At the end, ARUM will come up with an intelligent enterprise bus integrating legacy with new intelligent planning systems.

In contrast, the third case puts the behaviour of the landscape into the foreground. This requires holistic modelling and, e.g., of running batches of Monte-Carlo simulations in reasonable response times. While in ARUM the operations’ network is beyond the scope the third example will need further research and experiments.

As a first approach similarities between existing and envisioned industrial HPC applications may help to demarcate the dimension of the scale: Such a similarity is provided by events propagating across a network specified by dimensions like space and time, temperature and humidity, organisation or technology. For example, event A with a likelihood of p(A) may occur in place 1 at time t_1 and with a likelihood of p(AB) will cause event B at place 2 and time t_2. The event may be *'increase of humidity of x%'* propagating with the direction of wind in a geographic area, or *'failure of resource X'* affecting subsequent operations in a factory.

In natural sciences, an event can be linked to geographic areas while in a business environment it is linked to areas of managerial responsibility. Although the domains materially differ, the concept of propagation applies to both and, besides laws of physics, is controlled by two arguments: The event risk and the impact of the event. And, as events in weather models may result in rain or sunshine, an unavailability of a resource may induce a downside (a production delay), or an upside if, e.g., the analysis of an unavailability or failure can help to pave the way for change of the process that reduces risk.

Based on such initial and rough deliberations, the computational scale of simulating interdependent risks organised by “cells” of managerial responsibility may be compared to the scale of a weather model with a similar number of geographic "cells". So MAS-goes-HPC approaches as taken here may also contribute to discussions in the HPC community about the needs for more autonomic software [27] to master an increasing complexity.

V. REPAST FOR HIGH PERFORMANCE COMPUTING

The combination of MAS principles and HPC platforms can be illustrated with the Repast HPC tool, which constitutes an example of an ABM system for massively distributed MAS solutions over large-scale distributed computing platforms.

The Recursive Porous Agent Simulation Toolkit (Repast) is one of several available ABMs toolkits, developed by the Argonne National Laboratory's Decision and Information Sciences Division. In spite of borrowing many concepts from the Swarm agent-based modelling toolkit [28], Repast differentiates from Swarm since Repast has multiple pure implementations in several languages and built-in adaptive features such as genetic algorithms and neural networks. Repast is a free open source toolkit, which supports the development of extremely flexible models of social agents, being possible to dynamically access and modify the agent’s model properties at run time. Repast includes a fully concurrent discrete event scheduler that supports both sequential and parallel discrete event operations, offering an automated Monte Carlo simulation framework.

Repast for High Performance Computing (Repast HPC), is written using C++ and makes use of the Message Passing Interface (MPI) [29] for parallel operations through distributed memory computing. Repast HPC is based on the development principles of the Repast Simphony toolkit (e.g., the Context and Projections concepts), and extended with functionalities to support the distribution of the agent-based model working in a parallel distributed environment. As an ABM toolkit, Repast HPC offers a suite of features that specifically facilitate creating, executing and evaluating agent-based models, but also provides the following key features that support the parallelisation of the agent-based model [30]:

- Synchronisation scheduling of events.
- Global data collection (similar to Repast Simphony).
- Automatic management of agent interactions across Processes.

Repast HPC is designed for a parallel environment in which many processes, each one containing several agents), are running in parallel and memory is not shared across processes. Since Repast HPC simulations are distributed across multiple processes, cross-process communication and synchronisation of the simulation state are required. This is performed automatically by Repast HPC.

A Repast HPC simulation consists of [30]:

- A set of agents, each one implemented as objects, i.e., as C++ classes (the agents’ state is represented by the attributes of the classes and the agents' behaviour represented by the methods in those classes).
- The Package type code, that are typically implemented as structures containing the minimal amount of agent state necessary to copy an agent from one process to another one.
- A model class, which is responsible to initialise the simulation, namely to create the SharedContext and populated it with agents, create projections and add them to the SharedContext, initialise data collection, schedule simulation actions and perform an initial synchronisation of agents and projections.
- A main function, which initialises the MPI and Repast HPC environments, creates the model class and triggers the Scheduler Runner.

After the development of the agent-based model, the model is ready to be simulated, requiring the configuration of the simulation, as well the code to make scalable some actions in the simulation. Note that in spite of Repast HPC does much of the parallel programming of an agent simulation, developers should provide details of how the simulation can be properly implemented to run in the parallel environment

Repast HPC was successfully tested for scalability on the Blue Gene/P computer of the Argonne National Laboratory's, which is a first signal of the framework maturity and reliability to run simulations in HPC environments. However, a long step is required to ensure the desired reliability and robustness of MAS parallelisation for critical large-scale applications.

Other aspect to be considered is the possibility to connect Repast HPC with other systems (e.g., legacy systems) into larger, heterogeneous application landscapes (e.g., for risk management spanning domains of engineering and business operations). In the literature, several connections to other systems, e.g., JADE platform that is widely used for the development of MAS solutions, are found. Since the API is written in C++, the connection to legacy systems could be simplified. Note that the Repast HPC version has two types of programming languages/environments: Logo-style C++ for less demanding applications and a C++ version for more complex ones.

VI. CONCLUSIONS AND FUTURE WORK

This paper discusses the combination of MAS and HPC concepts aiming to overcome the demanding computational requirements of MAS platforms when they are applied to very large-scale systems. For this purpose, several scenarios that benefit from the usage of MAS solutions running over high end platforms were presented, as well the emerging challenges of such combination.

In particular, it was noticed the importance of using such approach in the simulation of data considering huge population sizes. For this purpose, the Repast HPC framework was analysed, which is an agent-based modelling and simulation tool for handling massively distributed MAS solutions running over large-scale distributed platforms.

Future work is devoted to the application of mechanisms to parallelise MAS solutions, and particularly Repast HPC, for large-scale systems involving complex problems.

ACKNOWLEDGMENTS

This work is partially supported by the EU FP7 Programme, in the frame of the ARUM project, Grant Agreement No. 31405.

REFERENCES

[1] J. Ferber, Multi-Agent Systems, An Introduction to Distributed Artificial Intelligence. Addison-Wesley, 1999.

[2] M. Wooldridge, An Introduction to Multi-Agent Systems. John Wiley & Sons, 2002.

[3] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach. Prentice Hall, 1995.

[4] “High-Performance Computing: Europe's place in a Global Race”, Communication from the commission to the European parliament, the council, the European economic and social committee and the committee of the regions, Brussels, 15 Feb., 2012.

[5] M. Luck, “50 Facts about Agent-based Computing” AgentLink III.

[6] F. Bellifemine, G. Caire, and D. Greenwood, Developing Multi-Agent Systems with JADE. Wiley, 2007.

[7] N. Gilbert and S. Bankes, “Platforms and Methods for Agent-based Modeling”, Proc. of the National Academy of Sciences of the USA, vol. 99, suppl. 3, 2002, pp. 7197-7198.

[8] S. F. Railsback, S. L. Lytinen, and S. K. Jackson, “Agent-Based Simulation Platforms: Review and Development Recommendations”, Simulation, 82(9), 2006, pp. 609-623.

[9] P. Leitão, J. Barbosa, and D. Trentesaux, “Bio-inspired Multi-Agent Systems for Re-configurable Manufacturing Systems”, Engineering Applications of Artificial Intelligence, 25(5), 2012, pp. 934-944.

[10] P. Wittek and X. Rubio-Campillo, “Scalable Agent-based Modelling with Cloud HPC Resources for Social Simulations”, Proc. of the 4th IEEE Intern. Conf. on Cloud Computing Technology and Science, 2012, pp. 355-362.

[11] A. Koestler, “The Ghost in the Machine”, Arkana Books, London, 1969.

[12] mOSIAC project, http://www.mosaic-cloud.eu [accessed: 2013-04-12].

[13] EnerGy project, http://www.projectenergy.eu/flash.html [accessed: 2013-04-12].

[14] C. Macal, M. North, G. Pieper, and C. Drugan, "Agent-Based Modeling and Simulation for Exascale Computing", SciDAC Review, Issue 8, 2008, pp. 34-41.

[15] N. Collier and M. North, “Repast HPC: A Platform for Large-scale Agent-based Modeling”, Large-Scale Computing Techniques for Complex System Simulations, W. Dubitzky, K. Kurowski, and B. Schott (eds.), Wiley, 2011, pp. 81-110.

[16] S. Chin et al., “FLAME-II: A Redesign of the Flexible Large-scale Agent-based Modelling Environment”, RAL Technical Reports, RAL-TR-2012-019, November 2012.

[17] L. Gulýas et al., “Tools for Large Scale (distributed) Agent-based Computational Experiments”, Proc. of Computational Social Science Society of America Annual Conf. (CSSSA’11).

[18] E. Blanchart et al., “EPIS: A Grid Platform to Ease and Optimize Multi-agent Simulators Running”, Advances on Practical Applications of Agents and Multiagent Systems Advances in Intelligent and Soft Computing, vol. 88, 2011, pp. 129-134.

[19] http://cordis.europa.eu/fp7/ict/ssai/ [accessed: 2013-03-14].

[20] http://www.cvisproject.org/ [accessed: 2013-03-14].

[21] J. M. Mendes, F. Restivo, A. W. Colombo, and P. Leitão, “Behaviour and Integration of Service-oriented Automation and Production Devices at the Shop-Floor”, International Journal of Computer Aided Engineering and Technology, vol. 3, n. 3/4, 2011, pp. 281-291.

[22] http://www.eurocontrol.int/sesar/public/standard_page/overview.html [accessed: 2013-03-14].

[23] P. Hermanns, “Organizational Hubris, the Rise and Fall of the Celebrity Firm CargoLifter AG”, PhD Thesis, Kölner Wissenschaftsverlag Albers, Peters & Reihlen GbR, 2012, accessible online: http://www.diss.fu-berlin.de/diss/receive/FUDISS_thesis_000000039985?lang=en [accessed: 2013-07-02].

[24] http://www.eads.com/eads/int/en/news/press.20121105_eads_arum.html [accessed: 2013-03-14].

[25] --, “Dreamliner Grounded Until At Least End Of May”, Sky News, 25th February, 2013, accessible online: http://news.sky.com/story/1056514/dreamliner-grounded-until-at-least-end-of-may [accessed: 2013-07-02].

[26] J. M. Kingsley, “Airbus Slows A380 Final Assembly Ramp-up”, Flight global, 14th May 2009, accessible online: www.flightglobal.com/news/articles/airbus-slows-a380-final-assembly-ramp-up-326416/ [accessed: 2013-02-25].

[27] C. J. Earl, D. Chirag, and S. Conway, “World wide Technical Computing 2013 - Top 10 Predictions”, Filing Information: February 2013, IDC #239421, vol. 1, accessible online: http://www.idc.com/getdoc.jsp?containerId=239421 [accessed 2013-04-18].

[28] G. Madey, V. Freeh, and R. Tynan, “Agent-based Modelling of Open Source using Swarm”, Proceedings of the 8th Americas Conference on Information Systems, 2002, pp. 1472-1475.

[29] W. Gropp, E. Lusk, and A. Skjellum, “Using MPI: Portable Parallel Programming with the Message-Passing Interface”, 2nd Edition, MIT Press, 2000.

[30] N. Collier, “Repast HPC Manual”, 6th February, 2012.