24th International Conference on Robotics in Alpe-Adria-Danube Region, RAAD 2015
May 27-29, Bucharest, Romania

The final publication is available on link.springer.com

Case Studies for Education in Robotics: From Serious Games to "Technology to Teach Technology" Platforms

Monica Drăgoicea and Theodor Borangiu

Politehnica University of Bucharest
Faculty of Automatic Control and Computers
313, Splaiul Independentei, 060042-Bucharest, Romania

monica.dragoicea@acse.pub.ro, borangiu@cimr.pub.ro

Abstract. This paper describes a specific perspective on developing case studies related to education in robotics. The proposed framework intends to support students learning how to develop distributed software applications through functionality composition. The multi-agent approach is used as a test bed for case studies development. It tries to stress the role of the application development platform in creating rich simulations, giving students the possibility to express their goals with clarity, and creating suitable application architectures to achieve their goals. The presented roadmap describes three "Technology to Teach Technology" platforms that support distributed application development. The last section of the paper gives the structure of a specific case study in mobile robotics, along with the Presage2 multi-agent platform.

Keywords: multi agent systems, mobile robotics, agent directed simulation

1 Introduction

Many companies are starting to develop specific software tools on a larger scale to help their users to interact with real world problems and to better develop skills for their new jobs facing a larger exposure to the Information Technology in different industry sectors. Even at the European Commission level it is recognized that ICT literacy should be encouraged and the education of tomorrow's ICT professionals should create that body of knowledge that supports creation and implementation of specific solutions to meet customers' needs and realize business opportunities [1].

The "serious games" movement is oriented today towards more educational purposes, in which players are involved in real-time interactions. These interactions are created in virtual worlds supporting decision making processes. *Innov8* [2, 3] and *CityOne* [4] are only two examples of educational games that help students evaluate their business skills and the consequences of their decisions, also addressing complex issues of the energy, water, banking and retail industries. Other examples of immersive educational simulation platforms, which are built as serious games, are *Volvo Car UK* [5], a learning solution that teaches sales people on essential topics of legislation

adfa, p. 197, 2011.

for automobile sales, or *Darfur is Dying* [6] which teaches people about the social and economic situation in Darfur.

Besides these above mentioned names, there are many other examples of games available on the market, and several types of creators and vendors are actively playing in it. Among them there are traditional video games companies, serious games vendors, redoubtable software giants like IBM and Microsoft, or multimedia companies. Using any of these solutions implies that users have a virtual interaction with software applications or platforms that let them either entertain or solve problems, teach, investigate, and advertise in a supervised way, according to general pre-programmed game rules.

However, this paper intends to express a different perspective on specific aspects of this type of dedicated virtual worlds, both on academic perspective and personal skills' development. This perspective tries to stress the role of the application development platform, not only in creating rich simulations (see for example [7]) but also in giving students the possibility to express their goals with clarity and to create suitable application architectures to achieve these goals.

In this respect, section 2 of the paper briefly describes three TTT-based platforms (*Technology to Teach Technology*) to support case studies in robotics education. The concept of self-organization in distributed software applications is further depicted in section 2 along with Presage2, a programming platform dedicated to the design of multi-agent systems, and a case study in robot soccer is formulated in section 3. Section 4 concludes the paper.

2 Platforms and Education in Robotics

2.1 Robocode and Object Oriented Technology

Robocode is an open source programming game in which two or more agents (embodying real world mobile robots) compete in a battle-type task [8, 9]. The simulator runs across all platforms supporting Java, being easily integrated with the Eclipse development platform [10]. Robocode evolved lately as a robot programming platform where modern AI programming and machine learning techniques can be evaluated to further education in robotics, for example [11, 12, 13].

2.2 Microsoft Robotics Developers Studio and Service-Oriented Computing

Over the last decades, new styles of writing software applications have been proposed, advancing as distributed information processing and functionality integration approaches. Service-Oriented Architecture (SOA) is being used today not only for e-commerce or business applications, but also emerged as a powerful software architecture design approach in its form of Service Oriented Computing (SOC) addressing more technologically based developments, like embedded applications [14, 15, 16].

The SOA development cycle requires specific approaches for developing service-oriented solutions like Service-Oriented Modelling and Architecture (SOMA) [17],

Service-Oriented System Engineering (SOSE) [18], and dedicated environments, like SOMA-SE, the platform supporting the model-driven design of SOA solutions [19].

These specific research and development directions raised also a new evolution perspective on the educational body of knowledge necessary to acquire these new IT skills, which is different from traditional software development processes. As SOA paradigm explicitly separates software engineering tasks from programming tasks, the well trained service-oriented system engineers focus on reusable components, which they utilize to solve real-world complex problems. Therefore, the educational goal has to be shifted towards teaching students the overall application architecture and how to compose large applications using existing functionalities (software services) [20].

In the mobile robotics domain, these skills can be exposed along with the Microsoft Robotics Developer Studio (MRDS), a reference programming platform that can be used as a basis for real mobile robots integration. MRDS was proposed as a choice of expressing service-oriented computing (SOC) principles such as modularity and reusability related to applying SOA to embedded systems development with direct application to mobile robot control architecture design [21].

2.3 Presage2 and self-organizing multi-agent systems

Considering the distributed nature of information processing in Multi-Agent Systems (MAS), Agent Oriented Computing (AOC) is used today to implement complex distributed applications and distributed intelligent systems, facing aspects like coordination [22], self-organization [23], and functionality composition [24]. At the same time, Agent Oriented Computing (AOC) and Service Oriented Computing (SOC) are proposed to be used today from an integrated exploitation perspective in which they contribute to distributed system design [25]. Agent Directed Simulation (ADS) [26] integrates different forms of agent-related research, such as agent simulation (using of simulation for agents) and agent-based simulation (using agents for simulation). Specifically, these research directions above described focus on improving distributed system design, with a strong accent on the aggregation of software components and their associated interactions in the model development. Software application functionalities are composed through interaction and they are embedded in the system model design. In a natural way, the role of the platform is also emphasised along with these new research developments, expressing *emergent phenomena* and *interactive system design* [27].

Presage2 was proposed as a simulation platform for prototyping societies of agents and developing Java-based *animation* and *simulations* of collective adaptive systems [28]. However, its usefulness proved to extend towards more complex visions, as common pool of resources management [29], along with new frameworks for resource allocation such as computational justice [30] following principles of sustainable institutions [29]. In Presage2, formal models of distributed information processing systems expressed as agent societies can be operationalized through simulation, while software components' interaction is supported by means of a powerful rule-based agent choreography mechanism.

3 Designing Case Study Robotic Applications with Presage2

With Presage2, case studies approaching education in robotics can be further designed to highlight specific educational principles to sustain the way future generations will approach real-world problems. Fig. 1 presents a "**virtual build and test**" application development scenario, through which students may learn to include into the agent-based model a whole range of representations of surrounding reality. The development can be further guided following well defined mobile robotic use cases expressing *self-organizing*, *coordination*, *cooperation* and *negotiation* aspects where *behavioural assemblages* emulate a certain *group dynamics*, like *robotic foraging*, *robot soccer* and *formation maintenance* [31].

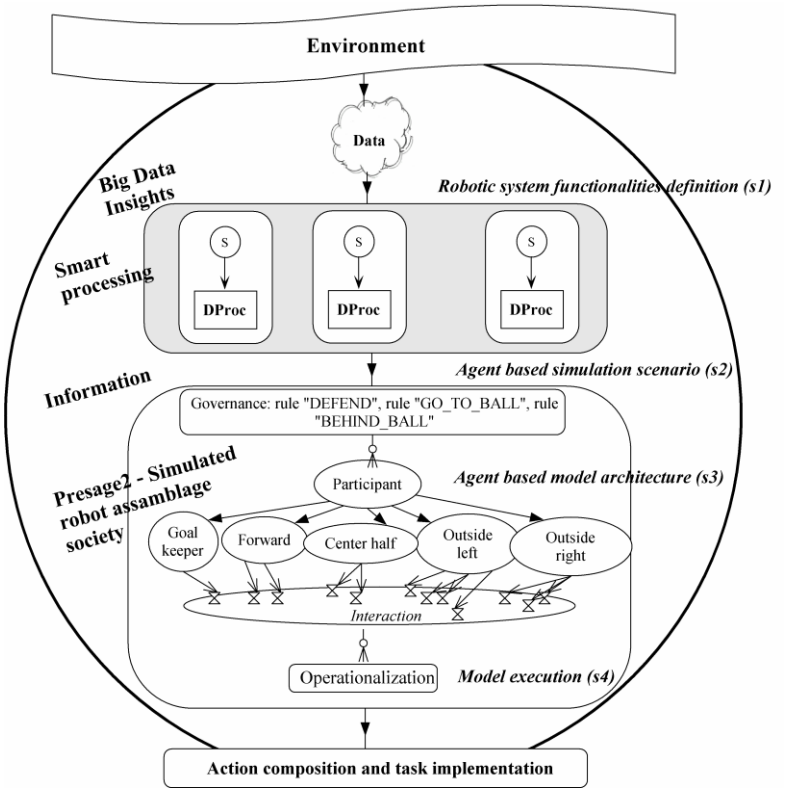

Fig. 1. Virtual build and test scenario in robot education tasks

Fig. 1 illustrates four main steps to follow when educational case studies in robotics are defined with Presage2 multi-agent programming platform:

- (s1) *Robotic task identification*. A requirements document is created to define overall robotic system functionalities, as well as performance measure for system validation through simulation, based on independent, local decisions and actions of the system's components;

- (s2) *Scenario definition*. A use case model is created to comply with the robotic task objectives. The agent based simulation scenarios will be further defined and the agent based simulation will be created in an environment that allows validating different aspects of interaction between task participants;
- (s3) *Formal model design of the robotic task*. The architecture of the agent based model is defined, consisting of a set of agents, the environment in which they operate, agents' communication protocol, and the set of general rules according to which they execute their actions;
- (s4) *Agent based operational modelling of the simulated robot assemblage society*. The agent based implementation model is executed in Presage2.

Fulfilling these steps leads to the creation of specific outputs in terms of models and artefacts to be used later for action composition and task implementation on real mobile robots. As a consequence, different design and development aspects can be expressed, such as: a) coordination of various robotic systems; b) experimentation within a simulated environment for coordinating the interactions of the different components of the multi robotics systems; c) modelling of the multi robotic system as a system of systems through the formal model design of the robotic task expressing coordination architectural styles; and d) developing a set of model architectures expressing distributed system design aspects, allowing students to observe global outcomes that are the consequence of agents' *interactions* and adaptation to specific *governance rules.*

3.1 Formal model design of the robotic task using Presage2

This section formalizes a robot soccer task as a Presage2 multi-agent model, depicting step (s3) in the previous section. Robot soccer is an example of a robot task described as an artificial society in which societal rules can be analysed. Robots in a soccer game exhibit active cooperation as well as non-active cooperation in pursuing their tasks.

The set of mobile robots defined in the Presage2 multi-agent model consists on the following players: goal-keeper, forward, outside-left, outside-right, and centre-half. This section only depicts model creation for goal-keeper and forward players, while all the other field players display the same behaviours, being different only on their position on the playground. Table 1 defines the elementary behaviours and general rules triggering these behaviours, which are sequenced to form a complete strategy; each agent selects from a set of behavioural assemblages to complete the task [31].

Table 1. A Presage2 formal model of a robot soccer task

Agent	Behavioural Assemblage	Governance rules (Drools)	Action
Goal-keeper	DEFEND	rule "Defend" when Goal-keeper(detect_ball()) then moveHalfDist();	Go to the half distance between ball and the middle of its team gate

	GO_TO_BALL	rule “Detect Ball” when Goal-keeper (ball_close()) then follow_ball();	Activated when the ball is close to the gate
Forward	BEHIND_BALL	Rule “Attack” when Forward(find_ball()) then goToGate();	Activated when agent is close to the ball, positioned between the ball and the opponent gate
	GO_TO_BALL	rule “Follow Ball” when Forward (ball_close()) then follow_ball();	Activated when the ball is close to the gate

Fig. 2 presents the steps required for the execution of the agent-based simulation model in Presage2.

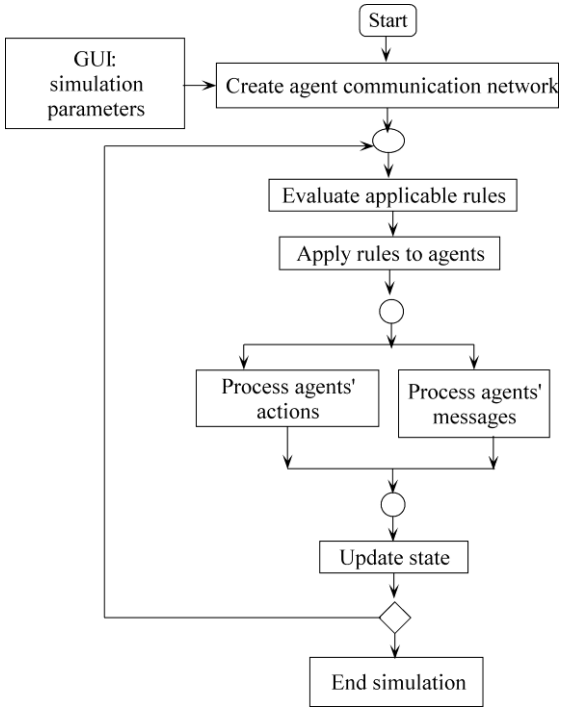

Fig. 2. Execution of the agent-based simulation model in Presage2

Each agent executes different actions (behaviours) based on the specific governance rule that applies to him. Each one of the behaviours is triggered by an environmental condition being tested as a Drools rule. Once the condition is true, the agent action is selected to be executed in a reactive manner. For example, in case of the forward player, once the ball position has been determined, the agent executes an

approach behaviour, which should drive it into a position to kick the ball forward or even into the opponent's goal. To effectively implement actions, different methods were proposed to be used, for example the potential field method [31]. In its associated area, each robot is subjected to attractive forces (generated by the ball or the goal) and repulsive forces (generated by the opponent players).

4 Conclusions

This paper emphasises the role of the development platform in supporting teaching of specific technology-related topics. The concept of "Technology to Teach Technology" is used to highlight specific characteristics of three development platforms that support application building and testing of real world scenarios in virtual environments. A Robot Soccer task was described and its Presage2 model derived. The vision described in the paper emphasises a possible academic perspective for education in robotic engineering field that combines the SOC application development with a high-level visual modelling approach assisted by animation and simulation tools.

5 References

1. European Commission: The European Foundational ICT Body of Knowledge. E-Skills: Promotion of ICT Professionalism in Europe, Capgemini Consulting (2014)
2. IBM: Innov8: IBM Introduces Video Game to Help University Students Develop Business Skills, http://www-03.ibm.com/press/us/en/presskit/22501.wss (2007)
3. IBM: IBM Serious Game Provides Training to Tackle Global Business Challenges. IBM Press, http://www-03.ibm.com/press/us/en/pressrelease/26734.wss (2009)
4. IBM: IBM Launches First Smarter Planet Game to Tackle City Challenges, http://www-03.ibm.com/press/us/en/photo/32607.wss (2010)
5. Volvo: VOLVO CAR UK: Serious Games Replicating A Real-Life Showroom Experience. Serious game at http://www.futurelab.net/ (2007)
6. Knowlton, A.: Darfur is dying: A Narrative Analysis. Master of Arts Thesis, University of Nebraska at Omaha (2009)
7. Microsoft: ESPSDK Overview, http://msdn.microsoft.com/en-us/library/cc526948.aspx, accessed on January 6th (2015)
8. IBM: Robocode is The Number One Download on IBM alphaWorks. http://www-03.ibm.com/press/us/en/pressrelease/992.wss (2001)
9. Nelson, M.: Robocode – the official site. Available at http://robocode.sourceforge.net, on-line accessed January 7, 2015 (2014)
10. IBM DeveloperWorks: Secrets from the Robocode masters: A collection of hints, tips, and advice from Robocode masters. Available http://www.ibm.com/developerworks/library/j-robotips/, on-line accessed January 7, 2015 (2002)
11. Nielson, J. Land and B.F Jensen: Modern AI for Games: Robocode. On-line at http://www.jonnielson.net/RoboReportOfficial.pdf (2010)
12. Hartness, K.: Robocode: using games to teach artificial intelligence. Journal of Computing Sciences in Colleges 19.4, 287-291 (2004)
13. Harper, R.: Evolving Robocode tanks for Evo Robocode. Journal Genetic Programming and Evolvable Machines, vol. 15(4), 403-431 (2014)

14. Karnouskos, S., Guinard, D., Savio, D., Spiess, P., Baecker, O., Trifa, V. and L. Moreira Sa de Souza: Towards the Real-Time Enterprise: Service-Based Integration of Heterogeneous SOA-Ready Industrial Devices with Enterprise Applications. In: 13th IFAC Symposium on INCOM, Russia, vol. 13(1), 2131-2136 (2009)
15. De Deugd, S., Carroll, R., Kelly, K. E., Millett, B. and J. Ricker.: SODA: service-oriented device architecture. IEEE Pervasive Computing, 5(3), 94-96 (2006)
16. Bohn, H., Bobek, A., and F. Golatowski: SIRENA-Service Infrastructure for Real-time Embedded Networked Devices: A service oriented framework for different domains. In: Networking, IEEE International Conference on Systems and International Conference on Mobile Communications and Learning Technologies, 43-43 (2006)
17. Arsanjani, A., Ghosh, S., Allam, A., Abdollah, T. and K. Holley: SOMA: A method for developing service-oriented solutions. IBM Systems Journal, vol. 47(3), 337-396 (2008)
18. Tsai, W.T.: Service-oriented system engineering: a new paradigm. In: IEEE International Workshop on Service-Oriented System Engineering (SOSE'05), 3-6 (2005)
19. Zhang, L.-J., Zhou, N., Chee, Y.-M., Jalaldeen, A., Ponnalagu, K., Sindhgatta, R. R., Arsanjani, A. and F. Bernardini: SOMA-ME: A platform for the model-driven design of SOA solutions. In: IBM Systems Journal, vol. 47(3), 397-413 (2008)
20. Tsai, W. T., Chen, Y., Cheng, C., Sun, X., Bitter, G. and M. White: An Introductory Course on Service-Oriented Computing for High Schools, Journal of Information Technlogy Education, 315-338, vol. 7 (2008a)
21. Tsai, W.T., Sun, X., Huang, Q. and H. Karatza: An ontology-based collaborative service-oriented simulation framework with Microsoft Robotics Studio, Simulation Modelling Practice and Theory 16, 1392–1414 (2008b)
22. Bedrouni, A., Mittu, R., Boukhtouta, A. and J. Berger: Distributed Intelligent Systems. A Coordination Perspective. Springer Science + Business Media (2009)
23. Bernon, C., Gleizes, M.P., Migeon, F. and G. Di Marzo Serugendo: Engineering Self-Organizing Systems. In: Di Marzo Serugendo, G., Gleizes, M. P., Karageorgos, A. (Eds.), Self-organising Software. From Natural to Artificial Adaptation, 283-312, Springer (2009)
24. Luo, J., Li, W., Liu, B., Zheng, X. and F. Dong: Multi-Agent Coordination for Service Composition. In Nathan Griffiths, Kuo-Ming Chao (Eds.), Agent-Based Service-Oriented Computing, 47-80, Springer (2010)
25. Griffiths, N. and K.-M. Chao: Agent-Based Service-Oriented Computing. Springer Book Series: Advanced Information and Knowledge Processing, Springer (2010)
26. Yilmaz, L. and T. Ören: Agent-directed Simulation. In: Agent-Directed Simulation and Systems Engineering, L. Yilmaz, T. Ören (Eds.), Wiley (2009)
27. Helbing, D. and S. Balietti: How to do agent-based simulations in the future: From modeling social mechanisms to emergent phenomena and interactive systems design. Tech. Rep. 11-06-024, Santa Fe Institute, NM, USA, Santa Fe Working Paper (2011)
28. Neville, B. and J. Pitt: PRESAGE: A Programming Environment for the Simulation of Agent Societies. Programming Multi-Agent Systems, Lecture Notes in Computer Science Vol. 5442, 88-103 (2008)
29. Ostrom, E.: Governing the Commons. The Evolution of Institutions for Collective Action. University of Cambridge Press (1990)
30. Pitt, J., Busquets, D., and R. Riveret: Formal Models of Social Processes: The Pursuit of Computational Justice in Self-Organising Multi-Agent Systems. 7th IEEE Int. Conf. on Self-Adaptive and Self-Organizing Systems (SASO), 269 – 270 (2013)
31. Dragoicea, M. and T. Serban: Behavioural Diversity in Cooperative Multi-robot Tasks, Proc. 16th International Workshop on Robotics in Alpe-Adria-Danube Region, May 26-28, Bucharest, Romania, 412-417 (2005)