ABStractme: Modularized Environment Modeling in Agent-based Simulations

(Demonstration)

Deividi Moreira[1], Fernando Santos[1,2], Matheus Barbieri[1], Ingrid Nunes[1,3], Ana L. C. Bazzan[1]
[1]UFRGS, Brazil; [2]UDESC, Brazil; [3]TU Dortmund, Germany
{dfsmoreira, fsantos, mcbarbieri, ingridnunes, bazzan}@inf.ufrgs.br

ABSTRACT

This paper presents ABSTRACTME, a tool for modeling the simulated environment in agent-based simulations. Differently from existing alternatives, ABSTRACTME allows specification of the environment in terms of *concerns*, which improve modularization. Moreover, it supports the modeling of setup aspects of the simulation, in addition to entities and the spatial abstraction. The tool generates ready-to-use code for the NetLogo simulation platform. A user study provided evidence that ABSTRACTME is useful, enjoyable, and easy to use and learn.
Demonstration video: `https://youtu.be/Z4DeVDwdjVw`

Keywords

Agent-based Modeling and Simulation, Modeling Tool, Environment, Code Generation, Concern

1. INTRODUCTION

Agent-based simulations (ABSs) have been widely used to understand complex systems. Building an agent-based simulation is a challenging task that has been investigated in the area of Agent-based Modeling and Simulation. The simulated environment, on which entities and agents are located, is a key element that must be specified in ABSs. In addition to the spatial abstraction, setup aspects must be considered in order to run a simulation: model parameters and strategies for creating and initializing entities. Despite the particular characteristics of each application area, these aspects share recurrent elements. However, they are often specified from scratch for every new simulation, through setup routines. Researchers have already argued about the demand for alternatives that increase the abstraction level and potentially ease the construction of ABSs [10, 11, 21].

Domain-specific languages (DSLs) [16] is a promising approach to support this demand. DSLs are modeling or programming languages specifically tailored to a particular domain, being able to express domain-specific high-level abstractions. Previous work has been done to increase the abstraction level of ABMS or provide support in other ways. However, they are limited to DSLs focused on specifying general purpose multiagent systems [2, 9, 20, 22], abstract approaches that do not provide elements for modeling either the environment or its setup [5, 6, 7, 8, 12], or simulation platforms that do not provide a rich set of high-level, recurrent abstractions for ABMS [13, 14, 17, 18, 25].

In this paper, we present ABSTRACTME, an Eclipse plugin that supports ABS development. Given that the there are many aspects associated with ABS, e.g. structure and dynamic behavior, our tool currently provides support to model the simulated *environment*, which is the basis of any ABS. The tool provides an implementation to the DSL4ABMS language [1, 23], which addresses the aforementioned aspects often left out of the scope of existing tools. Moreover, it provides the concept of *concern*, which is a grouping of model elements according to a particular aspect of the simulated environment. This supports the specification of the environment in a modularized way. The DSL4ABMS elements were identified based on a bottom-up domain analysis, which considered a set of 18 existing simulations (17 from the CoMSES Net Computational Model Library [19] and one related to the *Haiti Earthquake* [3]). The key features of ABSTRACTME are: (i) specification of environment concerns; (ii) modeling of individual concerns; and (iii) code generation to the NetLogo platform. A user study assessed the impact of DSL4ABMS in the comprehension of agent-based simulations. Results indicate that our language can indeed facilitate understanding of agent-based simulations, mainly when they include complex entities.

2. THE ABSTRACTME TOOL

ABSTRACTME was built upon Graphiti[1], and thus is integrated with the Eclipse platform. Figure 1 shows the graphical interface of ABSTRACTME, which is composed of three main views. The *Project Explorer* view (A) shows the projects created by the designer, as usual in the Eclipse platform. In this view, the designer can manage projects and diagrams. The *Diagram Editor* (B) is the main view of the ABSTRACTME, in which the designer models the ABS environment. The modeling elements of the DSL4ABMS are provided in the *Palette* view (C), from which the designer can drag and drop elements into the diagram.

Following the specification of the DSL4ABMS, ABSTRACTME organizes the environment model into an *overview* diagram and *concern* diagrams. The goal of the *overview* diagram (shown in Figure 1(B)) is to specify the many concerns associated with the ABS environment. This provides scalability to the environment modeling, as it is split into modularized parts, and supports the parallel modeling of each

[1] `https://eclipse.org/graphiti/`

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

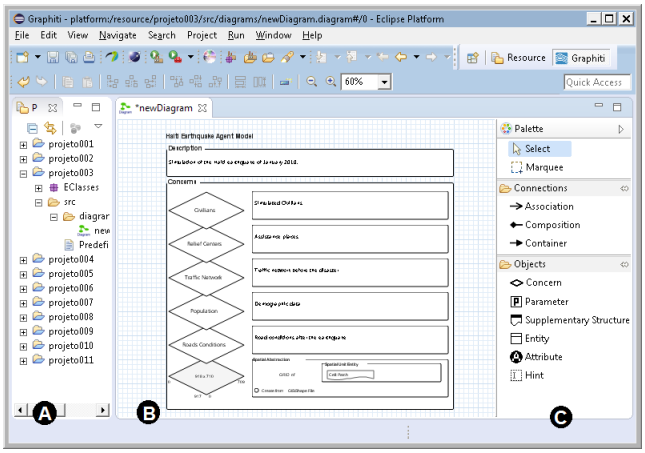

Figure 1: ABStractme Interface

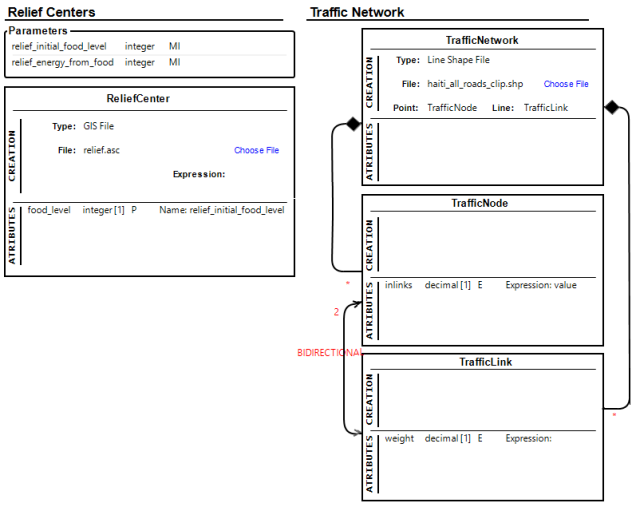

Figure 2: Concern Diagrams

concern by different domain experts. This is useful mainly when modeling simulations composed of a number subsystems [24]. Moreover, the overview diagram captures the title and purpose of the simulation, with a description. A diamond represents each concern, with its description beside it. If there is more than one concern, they are shown as a stack to emphasize that they represent the decomposition of the simulation into conceptual layers. Except for the bottom layer, there is no semantics associated with the ordering, and designers can choose an order using any criteria, such as abstraction level. The concern in the bottom layer, in turn, always represents the spatial abstraction of the environment in which there are situated entities. The designer chooses the appropriate abstraction among those available: grid, cartesian space, or graph. Then, the name of the spatial unit can be set to a meaningful term according to the vocabulary of the application area of the simulation.

When the designer double-clicks a concern element, ABStractme opens the corresponding concern diagram. Examples of concern diagrams are shown in Figure 2. In this type of diagram, it is possible to model elements of the concern, which can include parameters and entities with attributes and relationships. The following relationships between entities are supported, each with a particular semantics: *composition*, to represent that the entity is composed of others; *container*, to represent that the entity may contain others; and *association*, to represent relationships with other semantics between two entities (e.g., friendship between people). Setup aspects must be specified in order to run a simulation. For specifying how an entity is created, a creation strategy must be selected. It can be *designer defined*, in which the designer specifies the number of entities that are created and their locations. Alternatively, *CSV*, *GIS*, or *shape* files can be chosen, in which entities are created according to the content of the file provided. How model parameters or entity attributes are initialized must also be specified. Provided options range from parameter- or entity-specific initialization (static, manual, expression, parameter, being the last limited to entity attributes) to the initialization in batch through files (*CSV*, *GIS*, and *shape*).

ABStractme also provides code generation for the NetLogo simulation platform [25]. Statements that define the spatial abstraction and entities, visual components for manual inputs, and routines that create and initialize entities from files are automatically generated. The produced code is ready-to-use, as opposed to existing similar agent-based tools that usually generate only code skeletons. The designer just needs to open the generated file using NetLogo and press the setup button to have all the environment model created.

3. EVALUATION

The usability of ABStractme was evaluated by means of a user study. Ten volunteers—undergraduate and graduate students of the Institute of Informatics at UFRGS—were submitted to a modeling session. Study participants had to model the environment of the Haiti Earthquake [3] using ABStractme, based on a given DSL4ABMS model as the goal was to evaluate the tool usability only, together with the compliance of ABStractme with the DSL4ABMS specification. Before the modeling session, participants were asked to fill a background form. 60% reported little or no knowledge in ABMS, and 40% reported intermediary knowledge. After the modeling session, a post-evaluation based on the Usefulness, Satisfaction, and Ease of use questionnaire [15] was applied (complete results are available elsewhere [4]).

Overall, 98% of the participants agreed (above the intermediate level) that the tool is useful, enjoyable, and easy to use and learn. Participants emphasized the intuitive interface of ABStractme and the ability to specify concerns using distinct diagrams allowing a clear separation of concern in models. The impossibility of resizing elements and the difficulty to edit text fields, despite being reported as negative aspects, can be easily fixed in future releases.

4. CONCLUSIONS

This paper presented ABStractme, a novel tool for modeling the simulated environment in ABS. Besides features for modeling using visual diagrams in terms of concerns, the tool provides code generation for the NetLogo simulation platform that seamlessly produces ready-to-execute code. Ongoing work already addresses other ABS aspects to complement environment specification, such as dynamic aspects. Our long-term goal is to provide a comprehensive model-driven approach to ABS.

REFERENCES

[1] DSL4ABMS. http://www.inf.ufrgs.br/prosoft/resources/dsl4abms/. Accessed: 2017-02-22.

[2] R. Červenka, I. Trenčanský, M. Calisti, and D. Greenwood. AML: Agent modeling language toward industry-grade agent-based modeling. In *International Workshop on Agent-Oriented Software Engineering*, Lecture Notes in Computer Science, pages 31–46, New York, USA, July 2005. Springer.

[3] A. T. Crooks and S. Wise. GIS and agent-based models for humanitarian assistance. *Computers, Environment and Urban Systems*, 41:100–111, 2013.

[4] D. F. da Silva Moreira. Ferramenta para a modelagem de simulações baseadas em agentes usando linguagem específica de domínio. Trabalho de Conclusão de Curso, UFRGS, 2016. http://hdl.handle.net/10183/150944.

[5] A. Garro, F. Parisi, and W. Russo. A process based on the model-driven architecture to enable the definition of platform-independent simulation models. In N. Pina, J. Kacprzyk, and J. Filipe, editors, *Simulation and Modeling Methodologies, Technologies and Applications*, volume 197 of *Advances in Intelligent Systems and Computing*, pages 113–129. Springer Berlin Heidelberg, 2013.

[6] A. Garro and W. Russo. easyABMS: A domain-expert oriented methodology for agent-based modeling and simulation. *Simulation Modelling Practice and Theory*, 18(10):1453–1467, 2010.

[7] A. Ghorbani, P. Bots, V. Dignum, and G. Dijkema. MAIA: a framework for developing agent-based social simulations. *Journal of Artificial Societies and Social Simulation*, 16(2):9, 2013.

[8] J. J. Gómez-Sanz, C. R. Fernández, and J. Arroyo. Model driven development and simulations with the INGENIAS agent framework. *Simulation Modelling Practice and Theory*, 18(10):1468 – 1482, 2010. Simulation-based Design and Evaluation of Multi-Agent Systems.

[9] C. Hahn. A domain specific modeling language for multiagent systems. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, pages 233–240. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[10] L. Hamill. Agent-based modelling: The next 15 years. *Journal of Artificial Societies and Social Simulation*, 13(4):7, 2010.

[11] F. Klügl and A. L. C. Bazzan. Agent-based modeling and simulation. *AI Magazine*, 33(3):29–40, 2012.

[12] F. Klügl and P. Davidsson. AMASON: Abstract meta-model for agent-based simulation. In M. Klusch, M. Thimm, and M. Paprzycki, editors, *Multiagent System Technologies*, volume 8076 of *Lecture Notes in Computer Science*, pages 101–114. Springer Berlin Heidelberg, 2013.

[13] F. Klügl, R. Herrler, and M. Fehler. SeSAm: Implementation of agent-based simulation using visual programming. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '06, pages 1439–1440, New York, NY, USA, 2006. ACM.

[14] S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan, and G. Balan. Mason: A multiagent simulation environment. *Simulation*, 81(7):517–527, 2005.

[15] A. M. Lund. Measuring usability with the USE questionnaire. *Usability Interface*, 8(2):3–6, 2001.

[16] M. Mernik, J. Heering, and A. M. Sloane. When and how to develop domain-specific languages. *ACM computing surveys (CSUR)*, 37(4):316–344, 2005.

[17] N. Minar, R. Burkhart, C. Langton, and M. Askenazi. The swarm simulation system: A toolkit for building multi-agent simulations. Technical report, Santa Fe Institute, Santa Fe, 1996.

[18] M. J. North, N. T. Collier, and J. R. Vos. Experiences creating three implementations of the repast agent modeling toolkit. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 16(1):1–25, Jan. 2006.

[19] OpenABM Consortium. OpenABM, 2016.

[20] L. Padgham and M. Winikoff. Prometheus: A methodology for developing intelligent agents. In F. Giunchiglia, J. Odell, and G. Weiß, editors, *International Workshop on Agent-Oriented Software Engineering*, Lecture Notes in Computer Science, pages 174–185, Bologna, Italy, July 2002. Springer.

[21] H. V. D. Parunak, R. Savit, and R. L. Riolo. Agent-based modeling vs. equation-based modeling: A case study and users' guide. In J. S. Sichman, R. Conte, and N. Gilbert, editors, *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, Lecture Notes in Computer Science, pages 10–25, Paris, France, July 1998. Springer.

[22] J. Pavón and J. Gómez-Sanz. Agent oriented software engineering with INGENIAS. In V. Mařík, M. Pěchouček, and J. Müller, editors, *International Central and Eastern European Conference on Multi-Agent Systems*, Lecture Notes in Computer Science, pages 394–403, Prague, Czech Republic, June 2003. Springer.

[23] F. Santos, I. Nunes, and A. Bazzan. Model-driven engineering in agent-based modeling and simulation: a case study in the traffic signal control domain. In S. Das, E. Durfee, K. Larson, and M. Winikoff, editors, *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017)*, São Paulo, May 2017. IFAAMAS. Forthcoming.

[24] D. Weyns, F. Michel, H. V. D. Parunak, O. Boissier, M. Schumacher, A. Ricci, A. Brandao, C. Carrascosa, O. Dikenelli, S. Galland, A. Pijoan, P. S. Kanmeugne, J. A. Rodriguez-Aguilar, J. Saunier, V. Urovi, and F. Zambonelli. Agent environments for multi-agent systems–a research roadmap. In D. Weyns and F. Michel, editors, *Agent Environments for Multi-Agent Systems IV*, volume 9068 of *Lecture Notes in Computer Science*, pages 3–21. Springer, 2015.

[25] U. Wilensky. NetLogo, 1999. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL.