Simulation Modelling Practice and Theory 000 (2017) 1–26

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Model-driven agent-based simulation development: A modeling language and empirical evaluation in the adaptive traffic signal control domain

Fernando Santos[a,b,*], Ingrid Nunes[a,c], Ana L.C. Bazzan[a]

[a] Universidade Federal do Rio Grande do Sul (UFRGS), Instituto de Informática, Avenida Bento Gonçalves, 9500, Porto Alegre, Rio Grande do Sul, 91501-970, Brazil
[b] Universidade do Estado de Santa Catarina (UDESC), Ibirama, Brazil
[c] TU Dortmund, Dortmund, Germany

ARTICLE INFO

Article history:
Available online xxx

Keywords:
Agent-based modeling and simulation
Model-driven development
Development effort
Traffic signal control

ABSTRACT

Model-driven development (MDD) is an approach for supporting the development of software systems, in which high-level modeling artifacts drive the production of time and effort-consuming low-level artifacts, such as the source code. Previous studies of the MDD effectiveness showed that it significantly increases development productivity, because the development effort is focused on the business domain rather than technical issues. However, MDD was exploited in the context of agent-based development in a limited way, and most of the existing proposals demonstrated the effectiveness of using MDD in this context by argumentation or examples, lacking disciplined empirical analyses. In this paper, we explore the use of MDD for agent-based modeling and simulation in the adaptive traffic signal control (ATSC) domain, in which autonomous agents are in charge of managing traffic light indicators to optimize traffic flow. We propose an MDD approach, composed of a modeling language and model-to-code transformations for producing runnable simulations automatically. In order to analyze the productivity gains of our MDD approach, we compared the amount of design and implementation artifacts produced using our approach and traditional simulation platforms. Results indicate that our approach reduces the workload to develop agent-based simulations in the ATSC domain.

1. Introduction

Agent-based simulations have been widely used to understand the emergent behavior of complex systems. These systems are composed of multiple entities, or agents, which can interact with each other and are situated in an environment that they can perceive and modify through their actions. It is not trivial to formulate analytical models that can simulate the behavior of such complex systems. Building them is a challenging task that has been widely investigated in the context of agent-based modeling and simulation (ABMS), a simulation paradigm that uses autonomous agents and multiagent systems to reproduce and explore a phenomenon under investigation.

* Corresponding author.
E-mail addresses: fernando.santos@udesc.br (F. Santos), ingridnunes@inf.ufrgs.br (I. Nunes), bazzan@inf.ufrgs.br (A.L.C. Bazzan).

https://doi.org/10.1016/j.simpat.2017.11.006

ABMS has been used to model simulations in many application areas, such as traffic, ecology, economics, and epidemiology [58]. According to Macal and North [58], in these cases, ABMS was selected as the simulation paradigm because it can explicitly incorporate the complexity arising from individual behavior and interactions that exist in real-world scenarios. Additionally, in agent-based simulations, agents can be endowed with learning or evolutionary capabilities to adapt to changes in themselves or the environment. Artificial intelligence techniques that provide such capabilities are well established and can be incorporated in agents leading to more realistic simulations [54].

The development of agent-based simulations involves different roles that must interact and communicate, each role having distinct expertise [30]. This is often a barrier to a successful development. Usually, the domain expertise is concentrated on the *thematician* and *modeler* roles, while the technical expertise (in ABMS and its simulation platforms) is concentrated on the *computer scientist* and *programmer* roles. Researchers have already argued about the importance of tools and building blocks that increase the abstraction level and therefore reduce the required technical expertise [44,54,65,89]. Such approaches would potentially ease the development of agent-based simulations.

Many alternatives have been proposed for supporting the development of agent-based simulations. Agent-based simulation platforms are the most promising, because they consider multiagent system (MAS) aspects such as agents, interactions, and the environment, in addition to simulation aspects such as the creation and initialization of entities and agents. These platforms, however, demand previous expertise in ABMS or programming. This demand for technical expertise would be significantly reduced by the provision of a solution that enables creating simulations by means of ABMS-related building blocks.

An approach towards this direction is *model-driven development* (MDD) [6,78], a software development approach whose goal is to express domain concepts effectively. MDD makes domain concepts (e.g., adaptation) available for modeling by means of a domain-specific language (DSL) [60]. Traditional software development approaches, in contrast, often only provide concepts from the solution space (e.g., programming statements and abstract types). Transformation engines and code generators reduce or suppress the development effort when using MDD [78].

MDD has already been considered by the modeling and simulation (M&S) community as a promising approach for producing executable simulations from models [20], and MDD approaches focused particularly on ABMS have been proposed. On the one hand, there are MDD approaches that rely on—or are inspired by—Unified Modeling Language (UML) diagrams. In these approaches, effectiveness is compromised because UML is not expressive enough to specify intricate aspects of agent-based systems [9]. On the other hand, there are MDD approaches that propose new metamodels or modeling languages. However, they consider modeling and code generation of just a few aspects of agent-based simulations, leaving much left to be developed in specific applications. Sophisticated agent features which are recurrent in agent-based simulations, such as adaptation or learning, are not considered. Moreover, these MDD approaches have been mostly evaluated from the point of view of *feasibility*. Evaluation, with quantification, of *effectiveness* is almost never considered. Therefore, there is a lack of empirical evidence regarding the effectiveness that MDD approaches promote to the development of agent-based simulations.

In this paper, we address these issues by further exploring the use of MDD in the context of ABMS and empirically assessing the benefits it provides. Previous work on MDD showed that the more specific the application domain, the higher the chance of success [46]. Therefore, we focus on the adaptive traffic signal control (ATSC) domain, in which autonomous agents are in charge of managing traffic light indicators and should be able to adapt their control policy in order to optimize traffic flow. We thus present an MDD approach for developing agent-based simulations in the ATSC domain. To develop our MDD approach, we (i) performed a domain analysis; (ii) designed a metamodel and modeling language; and (iii) developed model-to-code transformations. For the domain analysis, we considered existing simulations, which adopt either adaptation or reinforcement learning techniques in our target domain, to keep the scope limited due to the aforementioned reasons. Because these techniques are often complex to develop, the designed modeling language provides building blocks for modeling together with automated transformations for code generation. Although we focused on a specific domain, concepts included in our modeling language can potentially be adopted in other domains. In order to analyze the productivity gains of our MDD approach, we compared the amount of design and implementation artifacts produced using our approach and traditional simulation platforms. Results indicate that our approach reduces 60–86% of the workload to develop agent-based simulations in the ATSC domain.

Specifically, this work provides the following contributions: (i) a domain-specific modeling language (DSML) and code generator, which together support the development of applications in the investigated domain and potentially in similar domains; and (ii) an empirical evaluation that concretely demonstrates the benefits of MDD for developing agent-based simulations in the ATSC domain. The DSML metamodel captures aspects that have not been considered in existing metamodels for agent-based simulations and thus can be considered as a contribution to the ABMS field as well. Additionally, from the steps followed to identify recurrent concepts that are present in existing agent-based simulations, we describe a derived bottom-up domain analysis method. This method led us to successfully develop an MDD approach in the ATSC domain and can potentially be used to identify and to abstract concepts in other domains.

2. Background on model-driven development

Software engineering provides support for professional software development by means of techniques for specification, design, and evolution of systems. Software models are widely used in software engineering. In model-based development,