Environmental Modelling & Software 89 (2017) 172–189

Contents lists available at ScienceDirect

Environmental Modelling & Software

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

Design and demonstration of a data model to integrate agent-based and field-based modelling

Merijn P. de Bakker*, Kor de Jong, Oliver Schmitz, Derek Karssenberg

Department of Physical Geography, Faculty of Geosciences, Utrecht University, Heidelberglaan 2, PO Box 80115, 3508 TC Utrecht, The Netherlands

ARTICLE INFO

Article history:
Received 2 September 2016
Received in revised form
27 September 2016
Accepted 12 November 2016

Keywords:
Agent-based modelling
Field-based modelling
Integrated modelling
Data model
Map algebra
Modelling language

ABSTRACT

Dynamic environmental modelling of spatio-temporal systems often requires the representation of both fields and agents. Fields are continuous with values in the whole spatio-temporal domain of a model, while agents are bounded in space and often mobile. It is currently difficult for environmental modellers with limited software engineering background to construct such field-agent models, as modelling frameworks mostly do not support the integration of fields and agents. To overcome this issue, we describe a data model combining fields and agents in a single concept. This data model represents fields, agents and relations by grouping items sharing properties into a phenomenon. The concepts domain, property set and value handle spatio-temporal attribute representations. The data model is implemented in a software prototype that shows how data on fields and agents is stored and manipulated.

1. Introduction

Dynamic environmental models represent the temporal evolution of the environment by applying a state transition function on the system's state variables, mostly at each time step or for a series of events occurring over time. Their state variables often vary in the spatial domain as well as in the temporal domain, which requires that an appropriate representation of modelled phenomena has to be used that is capable of storing spatio-temporal information. Two approaches currently exist (Filatova et al., 2013; Goodchild, 2013; Parker, 2005), as shown in Fig. 1. In the field-based approach, attributes are assumed to have a value defined everywhere in space-time, often discretized in cells or voxels and time steps. Examples of field-based approaches are aboveground biomass represented by a raster map, where each grid cell value represents the biomass in the grid cell, for each time step, or atmospheric pressure, where each voxel is assigned a pressure value. In the agent-based approach, attributes are assigned to, sometimes mobile, entities bounded in space-time. Agents are sometimes referred to as individuals in ecology and objects in geographical information science (e.g. Grimm and Railsback, 2013; Goodchild et al., 2007). An example of the agent-based approach is where the modeller chooses to represent biomass as a biomass value assigned to each individual tree, in order to simulate, for instance, tree growth of individual trees (e.g. Valentine and Mäkelä, 2005).

In a large number of environmental problems, in particular those that require an integrated approach, there is a strong need to use environmental models that combine the agent-based and field-based approaches. This is because many systems contain both phenomena that are better represented by fields and those that are better represented by agents. In addition, there is often interaction between fields and agents (Fig. 1). In ecosystem models, for instance, animals are often best represented as agents, while their habitat is better represented as a field (e.g. Schippers et al., 2014; Bennett and Tang, 2006). Other examples include interactions between water users (agents) and groundwater dynamics (field) (Castilla-Rho et al., 2015), the influence of weather (fields) on trees (agents) (Schelhaas et al., 2007), and modelling of air pollution (field) and personal exposure to this air pollution of mobile individuals (Sbihi et al., 2015).

The integration of fields and agents in models is however still complicated from the perspective of model implementation, as standardized software is lacking. Generally speaking, field-agent integration can be approached in two ways. One is to build models from scratch with a general-purpose language (Fig. 2A), for instance FORTRAN or C++. With such a language, the modeller has

* Corresponding author.
E-mail address: m.p.debakker@uu.nl (M.P. de Bakker).

http://dx.doi.org/10.1016/j.envsoft.2016.11.016
1364-8152/

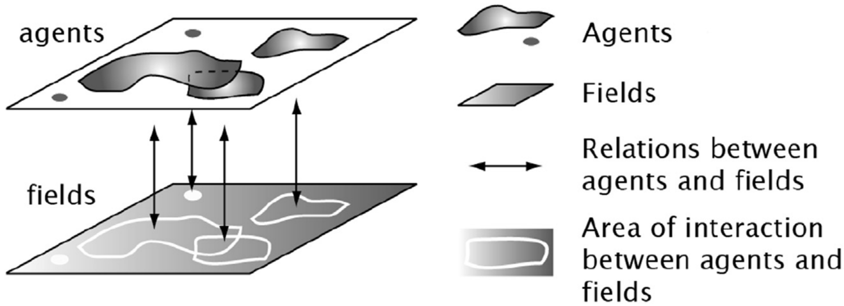

Fig. 1. An integrated agent-based and field-based model consists of *agents*, which are discrete entities in space (points, lines, areas) and are possibly mobile, and *fields*. Fields represent continuous attributes. Many agents have continuous variation within their spatio-temporal extent. Agents and fields interact depending on location and extent of agents.

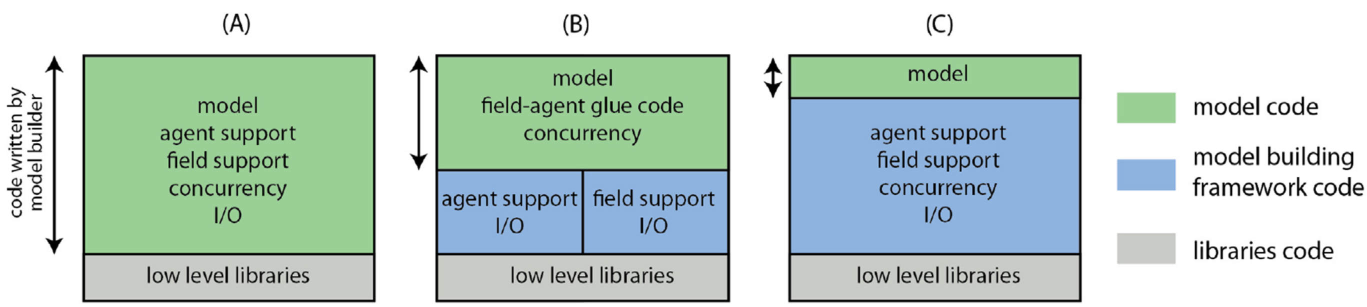

Fig. 2. Share of model code, framework code and general-purpose programming libraries code in three different approaches to programming models. (A), Using generic, general-purpose programming libraries to build a model from scratch; (B), Using separate model building frameworks for agents and fields; (C), proposed approach using a model building framework that supports fields and agents. Of the approaches presented in the figure, approach (C) requires the least programming effort from the modeller.

to implement software using elementary constructs which requires extensive software engineering skills, especially if the model implementation is challenging, e.g. when the model needs to be executed on a supercomputer. With a few exceptions in certain research areas this exceeds the programming capabilities of most domain specialists. Using these languages comes with a number of disadvantages (Karssenberg, 2002; van Engelen, 2001) and shifts the modeller's focus from describing domain processes to implementing numerical algorithms. A solution to this problem is offering model builders specialized programming interfaces tailored to their domain of knowledge (Fig. 2B). Domain experts then program models using a model building software framework (e.g. Karssenberg et al., 2010), which provides a programming interface matching the conceptual level of thinking of domain specialists. The building blocks of models can be offered in a generic scripting language or, in its most sophisticated form, as a Domain-Specific Language (DSL) (Mernik et al., 2005; van Deursen et al., 2000), which is a language that matches the semantics of a specific domain of use, improving productivity, verification and optimization (Kosar et al., 2010; Fisher, 1999). This approach of using model building software frameworks is promising because it becomes easier for domain specialists to program models (Holst and Belete, 2015). However, the current key limitation of model building software frameworks is that they focus on either field-based (e.g. Karssenberg et al., 2010) or agent-based modelling (e.g. NetLogo, 2012), but not both (Carneiro et al., 2013). This is inconvenient because coupling involves additional technical overhead and thus makes it more difficult for domain experts to build models.

The disadvantages of the current situation would be resolved by a model building framework supporting both fields and agents (Fig. 2C), which preferably includes a domain-specific language capable of manipulating fields and agents. The design of such a language is mostly based on a semantic model of the application domain (e.g. Fowler, 2010), e.g a (conceptual) data model. In this paper, we aim at the development and software implementation of a conceptual and physical data model integrating fields and agents that is useful in a domain-specific language by supporting multi-paradigm operations. The development of a new data model is required as existing data models, with respect to field-agent integration in dynamic environmental models, are still lacking regarding a number of aspects. Recent important contributions have approached integration of fields and agents from a theoretical point of view (e.g. Goodchild et al., 2007; Kjenstad, 2006; Galton, 2001), but the use of these concepts in a framework for dynamic environmental modelling is thus far underdeveloped. Also, existing physical, implemented, data models used in environmental model building frameworks (e.g. North et al., 2013; Karssenberg et al., 2010) do not integrate fields and agents. Further, to our knowledge, concepts and generic software frameworks that explicitly deal with spatio-temporal attribute variation within the extent of agents do currently not exist. This is relevant in models that model the evolution of individual features in a landscape, including their internal variation, for instance ecosystem models simulating temporal changes in the heterogeneity of foliage in individual tree crowns (Valentine and Mäkelä, 2005).

A key requirement of the conceptual data model is that it needs to function as an input argument of functions in a domain-specific language for dynamic environmental modelling. We build upon the concept of map algebra, introduced by Tomlin (1983) and since then used in many model building software packages for manipulation of fields (e.g., ESRI, 2015; van Deursen et al., 2000). Map algebra offers syntactically simple statements in the style of algebraic expressions on rasters instead of numbers. In our envisioned modelling language, the input arguments, which are raster maps in