Environmental Modelling & Software 46 (2013) 104–117

Contents lists available at SciVerse ScienceDirect

Environmental Modelling & Software

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

An extensible toolbox for modeling nature–society interactions

Tiago Garcia de Senna Carneiro[a,*], Pedro Ribeiro de Andrade[b], Gilberto Câmara[c], Antônio Miguel Vieira Monteiro[c], Rodrigo Reis Pereira[a]

[a] Earth System Simulation Laboratory (TerraLAB), Federal University of Ouro Preto (UFOP), Campus Universitário, Morro do Cruzeiro, Ouro Preto, MG 35900-000, Brazil
[b] Earth System Science Center (CCST), National Institute for Space Research (INPE), Av. dos Astronautas 1758, São José dos Campos, SP 12227-001, Brazil
[c] Image Processing Division (DPI), National Institute for Space Research (INPE), Av. dos Astronautas 1758, São José dos Campos, SP 12227-001, Brazil

ARTICLE INFO

Article history:
Received 21 March 2012
Received in revised form
8 February 2013
Accepted 7 March 2013
Available online 12 April 2013

Keywords:
Nature–society models
Multi-scale modeling
Environmental modeling
Discrete event simulation
Cellular automata
Multi-agent systems

ABSTRACT

Modeling interactions between social and natural systems is a hard task. It involves collecting data, building up a conceptual approach, implementing, calibrating, simulating, validating, and possibly repeating these steps again and again. There are different conceptual approaches proposed in the literature to tackle this problem. However, for complex problems it is better to combine different approaches, giving rise to a need for flexible and extensible frameworks for modeling nature–society interactions. In this paper we present TerraME, an open source toolbox that supports multi-paradigm and multi-scale modeling of coupled human-environmental systems. It enables models that combine agent-based, cellular automata, system dynamics, and discrete event simulation paradigms. TerraME has a GIS interface for managing real-world geospatial data and uses Lua, an expressive scripting language.

Software availability

Name: TerraME
Developer: Federal University of Ouro Preto (UFOP), Brazil; National Institute for Space Research (INPE), Brazil
Contact: tiago@iceb.ufop.br, pedro.andrade@inpe.br
Programming language: Lua
Optional additional software: MySQL and TerraView
License: GNU LGPL (open source)
Website: http://www.terrame.org

1. Introduction

Planners and policy makers need models that capture how human actions act on natural systems (Turner et al., 1995). These models represent coupled nature–society systems in different ways. Their capacity to capture the impact of human actions in nature depends on the spatial and temporal scales used. It also hinges on the chosen hypotheses about human behavior and environmental response. Despite the challenges involved in building them, these models have an important role. They bring forth unstated assumptions hidden in policy proposals, helping us to understand the possible results of different choices (Moran, 2010).

In this paper, we use the term *paradigm* to mean a worldview intrinsic to a scientific theory. Models of nature–society interactions use different paradigms, including cellular automata, agent-based models, map algebra, and system dynamics (White and Engelen, 1997; Parker et al., 2003; Karssenberg and De Jong, 2005; Batty, 2012). In many cases using a single paradigm is not enough. For complex problems, it is better to combine different methods to learn more about how human societies interact with nature (Rindfuss et al., 2004).

Most designers of nature–society modeling tools choose a paradigm and build a toolbox that supports it. Supporting a single paradigm has many advantages. Most paradigms have a lot of documentation and user communities, which helps potential adopters. However, designer choices may also limit a software's

* Corresponding author. Tel.: +55 (31) 3559 1692; fax: +55 (31) 3559 1660.
E-mail addresses: tiago@iceb.ufop.br (T.G.S. Carneiro), pedro.andrade@inpe.br (P.R. Andrade), gilberto.camara@inpe.br (G. Câmara), miguel@dpi.inpe.br (A.M.V. Monteiro), rreisp@gmail.com (R.R. Pereira).

1364-8152/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.envsoft.2013.03.002

ability to grow. Tool designers have to choose a programming environment, user interfaces, data types and their relations, algorithms, data handling, and storage. A design suited for one paradigm may not be adequate to support others. Although multi-paradigm modeling tools can in theory combine different ways of modeling, building such tools is a hard task. This begs the question: "*What kinds of software architecture are better suited for multi-paradigm modeling of nature–society interactions*?" In what follows, we refer to this challenge as the *multi-paradigm model design problem*.

This paper presents a possible response to this question. We were inspired by how Bjarne Stroustrup built C++ (Stroustrup, 1994). He designed C++ in a bottom-up, modular fashion, allowing object-oriented, generic programming, and procedural programming styles. The flexibility of C++ has no doubt contributed to its widespread use. Following these ideas, our proposed solution for the multi-paradigm model design problem stems from three conjectures. First, the tool should provide a *collection of data types and functions* needed by different paradigms. This leads to a bottom-up design based on building blocks that are combined by the modeller. The second conjecture is that *nature–society interactions happen in geographical space*. Unlike human and capital resources, that are mobile, natural resources are fixed. When dealing with environmental problems, we have to capture geographical features such as soil, climate, vegetation, and biodiversity in a spatially explicit way. Thus, models for nature–society interactions need a spatial component that represents natural landscapes and the results of human interactions with them. Third, *nature–society interactions occur at different scales*. Many problems need to be expressed as multi-scale models where matter, energy, and information flow between different scales. The toolkit should allow the user to break a complex model into simpler sub-models. Each sub-model is a micro-world with its own temporal and spatial resolution and behavior. Sub-models can then be nested and combined in different ways. Thus, our proposed architecture puts together a *set of data types* with *methods to build and connect geospatial micro-worlds*.

Based on these conjectures, we have designed and implemented the TerraME toolbox. It has building blocks for model development, allowing the user to specify the spatial, temporal, and behavioral parts of a model independently. Its components are expressive, enabling different approaches to be combined. TerraME's main aim is flexibility. It does not enforce a unique modeling paradigm, but provides the tools needed by the modeller. TerraME is an open source software distributed under the GNU LGPL license and is available at www.terrame.org.

In the next section, we consider the challenges for designing software to model nature–society interactions, pointing out the choices we made. We describe the general architecture of TerraME in Section 3. Section 4 has examples that show the main features of TerraME. We finish the paper by reflecting on the contributions and the limits of our proposed solution to the multi-paradigm model design problem.

2. Design choices for nature–society interaction modeling toolboxes

In this section, we discuss four decisions faced by designers of modeling tools that support nature–society interactions. In each case, we point out the choices we made in TerraME.

- Choosing which modeling paradigms to support.
- Selecting the model interface.
- Defining how the model interfaces with databases and GIS.
- Providing tools for verification, calibration, and validation.

2.1. Choice of modeling paradigms

Nature–society modeling paradigms include *Cellular Automata* (von Neumann, 1966), *System Dynamics* (Forrester, 1961), *Agent Based-Systems* (Wooldridge and Jennings, 1995), *Map Algebra* (Tomlin, 1990), and *Discrete Event System Specification* (Zeigler et al., 2005). Cellular automata (CA) are finite machines organized in a lattice connected by neighborhood relations. CAs can produce complex patterns from simple rules. In the system dynamics view, the world consists of stocks of energy, information, or matter. Model rules are differential equations defining flows that transport energy, information or matter between stocks. Agent-based models represent autonomous individuals that interact with themselves, the environment, and other agents. Map algebra uses raster maps to allocate properties in space and provides functions over maps to convey change. In the discrete event formalism, an event is an individual temporal episode. Instead of having functions that compute the next step of the simulation, an event-based model has a set of events and conditions when they occur.

Most existing modeling tools are centered on a paradigm, although they may support others. Examples of agent-based modeling tools are NetLogo (Tisue and Wilensky, 2004) and RePast (North et al., 2006). System modeling tools include STELLA (Roberts et al., 1983), Vensim (Eberlein and Peterson, 1992), and Simile (Muetzelfeldt and Massheder, 2003). PCRaster is a map algebra toolbox with extensions for dynamic modeling (Karssenberg et al., 2001, 2009; Wesseling et al., 1996). JDEVS is an event-based modeling software (Filippi and Bisgambiglia, 2004). Focusing in a paradigm favors knowledge reuse. Users familiar with one modeling paradigm will be comfortable when facing a new toolbox based on similar ideas. If one knows STELLA, learning Vensim and Simile is straightforward. Models developed in NetLogo can be ported to RePast without excessive work (Crooks and Castle, 2012). Designers can also extend an existing tool to support other paradigms than their original choice.

The alternative is to build a multi-paradigm modeling tool in a bottom-up way. This is what we did in TerraME since we hold that nature–society relations are inherently complex. As expressed by Mike Batty: "*in modeling*, the quest for parsimony, simplicity, and homogeneity is increasingly being confronted by the need for plausibility, richness, and heterogeneity" (Batty, 2012). A multi-paradigm toolbox allows modellers to combine different paradigms when solving a problem. However, such tools are harder to learn since there are many concepts to be grasped. Flexibility comes at a price. We recognize that not all users will be willing to make it, although we believe the effort is worthwhile.

2.2. Selecting the model interface

Modeling toolboxes need to provide analytical power to express complex problems. Nearly all tools use a programming language with additional high-level statements. Some tools also provide icon-based graphical programming, like the *system dynamics* tools STELLA and Simile. Visual interfaces are appealing and enable decision-makers to quickly grasp model behavior. However, it is not easy to express spatial variation using icons. Thus, most spatially-based tools use a programming language as their main interface.

In TerraME, we chose a programming language interface. To support rapid model implementation we chose Lua, an open-source interpreted language with extensible semantics (Ierusalimschy et al., 1996). The modeller uses a clear and expressive language that calls demanding operations in C++, hidden from him. This provides a good trade-off between source code directness and computational efficiency.