Open Archive Toulouse Archive Ouverte

OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible

This is an author's version published in:
http://oatao.univ-toulouse.fr/22664

Official URL

DOI : https://doi.org/10.1007/s10707-018-00339-6

To cite this version: Taillandier, Patrick and Gaudou, Benoit and Grignard, Arnaud and Huynh, Quang-Nghi and Marilleau, Nicolas and Caillou, Philippe and Philippon, Damien and Drogoul, Alexis *Building, composing and experimenting complex spatial models with the GAMA platform.* (2018) GeoInformatica. 1-24. ISSN 1384-6175

Any correspondence concerning this service should be sent to the repository administrator: tech-oatao@listes-diff.inp-toulouse.fr

https://doi.org/10.1007/s10707-018-00339-6

Building, composing and experimenting complex spatial models with the GAMA platform

Patrick Taillandier, Benoit Gaudou, Arnaud Grignard, Quang-Nghi Huynh Nicolas Marilleau, Philippe Caillou, Damien Philippon, Alexis Drogoul

Abstract
The agent-based modeling approach is now used in many domains such as geography, ecology, or economy, and more generally to study (spatially explicit) socio-environmental systems where the heterogeneity of the actors and the numerous feedback loops between them requires a modular and incremental approach to modeling. One major reason of this success, besides this conceptual facility, can be found in the support provided by the development of increasingly powerful software platforms, which now allow modelers without a strong background in computer science to easily and quickly develop their own models. Another trend observed in the latest years is the development of much more descriptive and detailed models able not only to better represent complex systems, but also answer more intricate questions. In that respect, if all agent-based modeling platforms support the design of small to mid-size models, i.e. models with little heterogeneity between agents, simple representation of the environment, simple agent decision-making processes, etc., very few are adapted to the design of large-scale models. GAMA is one of the latter. It has been designed with the aim of supporting the writing (and composing) of fairly complex models, with a strong support of the spatial dimension, while guaranteeing non-computer scientists an easy access to high-level, otherwise complex, operations. This paper presents GAMA 1.8, the latest revision to date of the platform, with a focus on its modeling language and its capabilities to manage the spatial dimension of models. The capabilities of GAMA are illustrated by the presentation of applications that take advantage of its new features.

Keywords Agent-based modeling · Spatial simulation · Platform · Modeling language

1 Introduction

Agent-based modeling is now a popular approach for representing, studying or exploring complex geographical systems [40]. This modeling approach proposes to represent

✉ Patrick Taillandier
patrick.taillandier@inra.fr

individually the entities composing a system and their interactions, and to let macroscopic behaviors emerge by simulation.

The adoption of such modeling approach was greatly fueled by the emergence of increasingly powerful software platforms, which now allow modelers without a strong background in computer science to easily and quickly develop their own models.

The trend observed for a few years is the development of much more descriptive models [28, 31, 45] integrating a high quantity of data [22], with a detailed representation of the environment. Supporting the development of such models requires addressing a set of scientific locks related to data integration, calculation efficiency, and visualization of simulations, which are particularly challenging if the platform has to be usable even by non-computer scientists. Trying to overcome these challenges is one of the main objectives of the GAMA (GIS Agent-based Modeling Architecture) platform.

GAMA has been developed since 2007 as an open-source project, undergoing several improvements over the years in order to answer the needs of its growing user base: dealing with more and more data formats, improving its ergonomics, efficiency and stability. However, GAMA also aims at going further and providing new ways to create or simulate models.

Its latest version, GAMA 1.8, has brought in many improvements: its modeling language has been enriched with new data types, new supported data format and new agent architectures. Powerful in-house simulation capabilities have been integrated (side-by-side execution of simulations with different parameter sets, multi-threaded execution of agents within simulations, etc.). In addition, its general user-interface has gained in flexibility and efficiency (editor, navigator, data viewers, etc.).

This article aims at presenting these enhancements individually but also in their general logic, as GAMA is evolving from a pure "agent-based" modeling platform to an "agent-oriented" generic modeling and simulation platform, where the coupling of heterogeneous data and modeling paradigms is now the rule rather than the exception. It is structured as follows. Section 2 presents the state of the art of modeling platforms and highlights the specificities of GAMA. Section 3 presents GAMA 1.8, and more particularly its modeling language, its capacity to manage the spatial dimension, and its new advanced features. In Section 4, the way real modeling projects make use of GAMA is detailed. Finally, Section 5 concludes by describing what its future road-map is.

2 Existing agent-based platforms

These last years have seen the development of many platforms, toolkits and frameworks dedicated to the implementation and simulation of agent-based models. In a recent survey, Kravari and Bassiliades [44] reviewed 24 of them. One of the possible criteria to classify these platforms is the type of language used to define models. Three categories (non-exclusive) can be defined:

- Platforms allowing to define models with a generic high-level programming language (Java, C++, Python, Smalltalk...). These platforms are, for most of them, dedicated to computer scientists and allow the development of large-scale models that integrate numerous data and rich agent behaviors. They take advantages of the numerous external libraries that have been developed independently of the platforms themselves. Platforms such as Repast [51], SWARM [49], JADE [11] and CORMAS [47] belong to this category.

- Platforms allowing to define models using a specific modeling language. These platforms are often simpler to use than the ones of the first category and are dedicated to a wider range of users. They require nonetheless some basic skills in algorithmic. The NetLogo [66] and GAMA platforms belong to this category.
- Platforms allowing to define a model with a graphical modeling language. These platforms often require no or few knowledge in algorithmic, but are very limited in terms of richness of models that can be built. Startlogo TNG [57] or AgentSheets [56] belong to this category.

Among these platforms, one of the most popular in the academic and industrial communities is JADE. JADE is an open-source Java FIPA-compliant framework that aims at simplifying the implementation of multi-agent systems. It is a generic platform - its scope exceeds agent-based simulation and covers all fields of multi-agent systems. JADE proposes many powerful features for building multi-agent systems (BDI architecture, migration of agents over computers…). However, using it to design models requires a lot of development in Java, and good skills in computer sciences and programming, in particular when dealing with geographical systems as JADE does not provide any feature to manage the spatial components of models.

Another open-source family of platforms allowing to define models with Java is Repast. In contrast to JADE, these frameworks are specialized in agent-based simulation, and thus provide many useful features for modelers, in particular to manage the spatial component of models (GIS data loading, grid topology, etc.). They offer as well different ways to write models (Java, graphical modeling, Netlogo modeling language…) in order to satisfy different kinds of users. For simple models, the modelers have a real choice of modeling languages. However, when it comes to more descriptive large-scale models, the modelers have to use a generic programming language (e.g. Java). Therefore, developing such descriptive models still requires high level skills in programming.

Another open-source platform based on a generic programming language is CORMAS. Compared to JADE and Repast, this platform, which allows to define models in Smalltalk, is more accessible to non-computer scientists. It provides modelers with many tools dedicated to participatory modeling and simulation (graphical modeling, replay of simulations, rich user-interaction definition…), which explains its success for projects where stakeholders are implicated. However, CORMAS is limited when it comes to define large-scale descriptive models, in particular concerning the spatial component. Indeed, CORMAS can only manage grid environment (rectangle or hexagon) and does not allow to directly manage vector geographical data.

If all these platforms offer powerful features, they remain, even CORMAS, complex to use by modelers with little skills in computer programming. NetLogo aims at overcoming this difficulty by allowing modelers to develop their models through a simple-to-use dedicated modeling language. NetLogo is nowadays one of the most popular platforms and was a powerful driver of the spread of agent-based modeling in social sciences. However, even with its numerous extensions, NetLogo suffers from important limitations such as the impossibility to define several displays of the environment and to use classic agent-oriented features (e.g. inheritance), basic integration of data (e.g. no agentification of GIS data), and computational performance. Therefore, most of the models developed with NetLogo are simple toy models, and NetLogo is rarely used to develop models that are more descriptive.

The objective behind the development of GAMA was to combine the advantages of all these platforms: allowing modelers to build models as quickly and easily as in NetLogo, while

going beyond what Repast or CORMAS offer in terms of simulated experiments. In order to reach this objective, GAMA provides modelers with a simple-to-use agent-based programming language, a powerful management of geographical data, flexible visualization tools and the capacity to carry out simulations with hundreds of thousands of agents. GAMA is currently used in many projects dealing with many domains such as environmental decision-support systems, urban systems, mobility, epidemiology, climate change adaptation and disaster mitigation.

3 The GAMA platform

3.1 An advanced IDE for agent-based modeling and simulation

One of the reasons of the success of modern modeling platforms, and more especially Netlogo, comes from the fact that they are integrated platforms allowing to switch very easily between a modeling perspective (writing of the model code) and a simulation perspective (running of the simulation). This possibility has a deep impact on the modeling process and on the modeler habits: rather than waiting to have a complete model before running simulations, Netlogo allows modelers to experiment very quickly the impact of the modification of the model code, and consequently favors a test and try modeling approach. The GAMA platform follows the same concept, but proposes more advanced modeling and simulation tools.

Indeed, GAMA is a full Integrated Development Environment (IDE) allowing like Netlogo to quickly switch between the modeling and simulation perspective, and that proposes a rich user interface. More specifically, GAMA, which is based on the Eclipse IDE, takes advantage of this IDE to offer to modelers numerous tools to manage models and simulations: flexible user interface including the possibility to arrange the different panels of GAMA using drag and drop or predefined layouts, powerful agent inspectors allowing to get information about one or several agents (tabular view), powerful search engine to get information and examples of uses of operators.

As GAMA models are often centered around data, GAMA provides modelers with viewers for common files (CSV, shapefile, OSM file, ASC, Geotiff...). The idea is to allow modelers to view the data they manipulate and the meta-data that are the most useful for their modeling task. As an example, the shapefile viewer allows the modeler to view the various geometries, but also each attribute name, the size of the bounding box or the projection.

In addition, GAMA provides a powerful modeling editor aiming at easing the modeler's work (Fig. 1). This editor integrates all the classical tools of modern IDEs such as auto-coloring, auto-compilation, auto-completion, and the possibility to automatically format or comment some lines of code. It allows to define code templates, which facilitate the development of models by sharing snippets of codes between models. Finally, it is fully linked to an extensive online documentation, allowing modelers to get information about the different keywords, operators and statements available.

In addition to the compiler of models, GAMA also integrates an interpreter of the language that can be invoked using an interactive console, allowing users to directly interact with agents (experiments, simulations and any agent belonging to a model) and to get a direct feedback of the execution of the code. This console is available throughout the platform, allowing, for example, to execute lines of code, change the preferences of GAMA or consult the documentation even when no simulation is running.

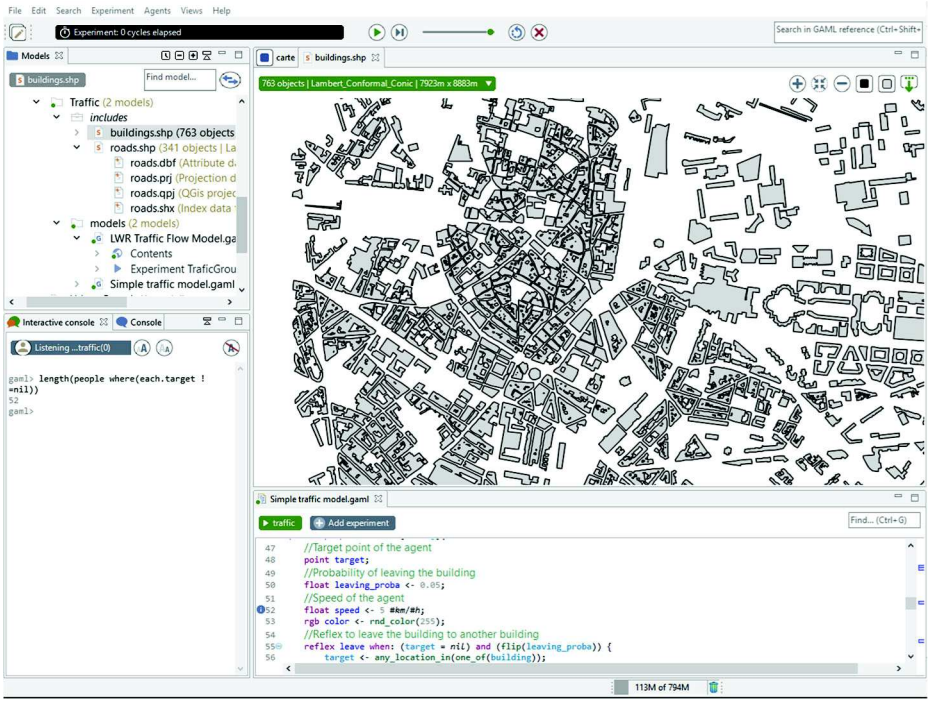


Fig. 1 Modeling interface of GAMA

3.2 The GAML modeling language

One of the strengths of GAMA is its ease of use, thanks to its modeling language called GAML (**GA**ma **M**odeling **L**anguage). GAML is an *agent-oriented language*, which means that everything "active" (entities of a model, simulations, experiments…) can be represented in GAML as an agent, and is thus a computational component owning its own data and executing its own behavior, alone or in interaction with other agents.

The language takes its roots in object-oriented languages like Java or Smalltalk, but extends the object-oriented programming approach with powerful agent concepts (like skills, declarative definitions or agent migration) to allow a better expressiveness in models. It is close to the Netlogo's agent-based modeling language, but it enriches the traditional representation of agents with modern computing notions like inheritance, type safety or multi-level agency. In addition, through the possibility to use different behavioral architectures for programming agents, GAML extends the agent-based paradigm to eliminate the boundaries between the domain of a model (which, in ABM, is represented with agents) and the experimental processes surrounding its simulations (which are usually not represented with agents), including, for example, visualization processes.

The main concepts behind GAML and GAMA come from [27]. This orientation has several conceptual consequences among which at least two are of immediate practical interest for modelers:

- Since simulations, or experiments, are represented by agents, GAMA can support high-level model compositionality, i.e. the definition of models that can use other models as

inner agents, leveraging multi-modeling or multi-paradigm modeling as particular cases of composition [41].
- The visualization of models can be expressed by models of visualization, composed of agents entirely dedicated to visually represent other agents, allowing for a clear separation of concerns between a simulation and its representation and, hence, the possibility to play with multiple representations of the same model at once [35].

The main concepts behind GAML can be described as follows (Fig. 2):

- Like in the object-oriented paradigm, where the notion of *class* is used to supply a specification for *objects*, agents in GAML are specified by their *species*, which provides them with a set of *attributes* (what they know), *actions* (what they can do), *reflexes* (what they actually do), and also specifies properties of their *population*, for instance their spatial organization (e.g. grid and graph) - called *topology* in GAMA, or their *schedule* (in which order and when they should be executed). More details about the notion of *species* in GAMA can be found in [38].
- Any *species* can be nested in another *species* (called its *macro-species*). A *species* can also inherit its properties from another *species* (called its *parent species*), creating a relationship

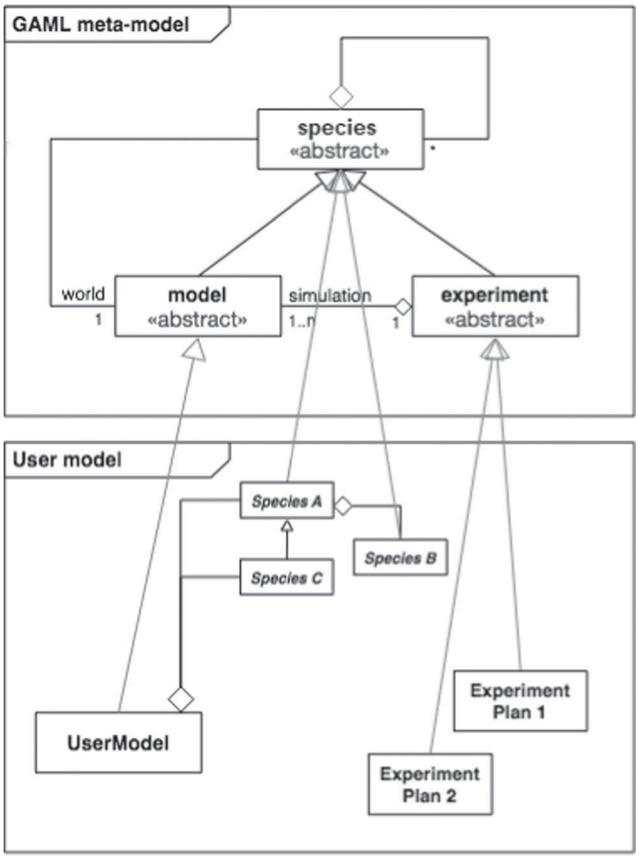


Fig. 2 Main concepts behind GAML

similar to *specialization* in object-oriented design. For example in Fig. 2, *Species B* is nested in *Species A,* and *Species C* inherits (attributes and behaviors) from *Species A*. Similarly, any *model* is nested in an *experiment*. In addition, *species* can be constructed in a compositional way with the notion of *skills*, which are bundles of *attributes* and *actions* that can be shared between different species and inherited by their children.

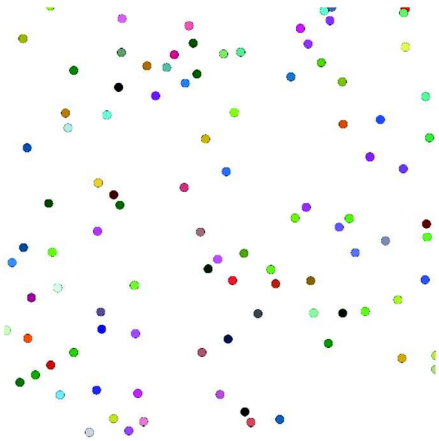
- *Simulations* and *experiments* are then instances of two species, which are, respectively, called *model* and *experiment plan*. We can think of them as "specialized" categories of species. An *experiment* will manage how simulations will be executed, and in particular what are the values of parameters and how are displayed the outputs of the simulations.

Writing a model in GAML consists then in defining a species which inherits from *model*, in which other species, inheriting (directly or not) from *species,* and representing the entities that populate this model, will be nested, and which is itself nested in one or several experiment plans among which a user will be able to choose which one he/she wants to execute. When a simulation is launched (i.e. when the user asks GAMA to execute an *Experiment Plan*), GAMA creates a single agent of the chosen *experiment plan* that will be responsible to create and execute *simulation* agents.

As an example, Fig. 3 shows a simple model: in this model, a species of agents called *simple_agent* is defined. The *moving skill* is attached to this species providing to the agents of this species some built-in attributes and built-in actions related to agent movement. We define for this species an attribute called *color* that is initialized with a random initial value. In addition, we define for this species a reflex called *behavior* that will be activated at each simulation step and that calls the *wander* action of the *moving skill* allowing the agent to move randomly (random walk). We define also for this species an aspect called *circle* that allows to represent the agents of this species as circles with their *color* attribute. We define as well the global section of the model that represents the first level of agency, and which describes the global component of the model. In this model, we just specify in the global section that we want at the initialization of the simulation to

```
model SimpleModel

global {
    init {
        create simple_agent number: 100;
    }
}
species simple_agent skills: [moving] {
    rgb color <- rnd_color(255);

    reflex behavior {
        do wander;
    }

    aspect circle {
        draw circle(1) color: color border: #black;
    }

}

experiment main_experiment type: gui {
    output {
        display map {
            species simple_agent aspect: circle;
        }
    }
}
```

a)



b)

Fig. 3 **a** GAML code of the *simpleModel:* this model creates 100 simple agents displayed by a circle with a random color; at each simulation step, these agents move randomly; **b** snapshot of the resulting simulation

create 100 agents of the *simple_agent* species that will be randomly located in the environment. At last, we define an *experiment plan* called *main_experiment* that just defines one output of type *display* called *map* that will display the agents of the species *simple_agent* with their *circle* aspect.

3.3 Spatial component of models

The interests of coupling GIS and ABM have been identified by many works (see for example [23]). However, most of existing agent-based platforms still propose a weak integration of GIS data in simulations, in particular the ones that are usable even by non-computer scientists. GAMA was developed from the beginning to overcome this limitation. Indeed, contrary to other agent-based modeling platforms such as CORMAS or Netlogo, GAMA allows to naturally integrate GIS vector and raster data. In addition, it allows to manage several representations of the same environment with different spatial organizations (for example grid and graph).

In GAMA, all models are provided with a continuous 3D environment. This environment will be nested with two main types of species: classic species that will have no specific spatial organization, and grid species, in which agents will represent cells organized as a grid. Regarding the grid species, the modeler can define the dimension of the grid by specifying the number of rows and columns, the size of the cells, or by using a raster geographical file (asc or geotiff). The modeler can also define the type of neighborhood that will be used: Moore, Von Neumann or hexagonal neighbors. In addition to these two type of species, agents can be organized as graph after their creation: the agents will represent either the nodes (for representing for example a social network) or the edges of the graph (for representing for example a road network). The graph can be directly defined by the modelers (e.g. *agent A* is linked to *agent B* through *edge agent C*), automatically built with classic network generators [6] (random network, preferential attachment), or built through spatial operators (e.g. building of a graph from a set of polylines).

An important feature of GAMA is that, like in a GIS, all agents are spatialized, i.e. are provided with a georeferenced geometry. This geometry, which is based on a vector representation and is by default a point, can be simple (point, polyline or polygon) or complex (composed of several sub-geometries). The geometry of the agents can be directly defined by the modeler (by using a list of points or predefined graphical primitives) or loaded from geographical data (databases, shapefile, OSM, GML...). Indeed, GAMA allows to use geographical data to create agents, each object of the geographical data will be automatically used to instantiate an agent, GAMA taking care of managing the coordinate reference system of the data (projecting the data to the coordinate reference system chosen by the modeler) and, if necessary, of reading the values of the attributes.

To facilitate the manipulation of agents, GAMA provides modelers with more than 100 spatial operators directly inspired by the ones proposed by most of GIS:

- *Creations*: creation of basic geometries such as circle, triangle, rectangle, hexagon, cube, or sphere.
- *Transformation*: buffer, translation, rotation, scaling, simplification, convex hull, decomposition into squares or triangles (Delaunay)...
- *Queries*: spatial queries concerning other geometries or agents, e.g. geometries at a given distance, overlapping a geometry, inside a geometry, closest to a geometry or farthest to a geometry....
- *Projections*: application of a spatial projection to a geometry according to a target coordinate reference systems.

- *Relations*: angle between two geometries, path between two geometries, distance between two *geometries*
- *Properties*: compute if two geometries overlaps, if one covers, crosses or touches another...
- *Operations*: union, difference, intersection...
- *Statistics*: spatial clustering, spatial interpolation, Moran Index….
- *Point-related*: random point in a geometry, closest point of a geometry to another, points along a geometry….

A specificity of GAMA is that it adapts the results of the operators according to the spatial organization (*topology* in GAMA) used for the computation. Figure 4 shows an example where the same operator to compute the distance between two geometries is used with the continuous environment, a grid and a graph.

Also based on these different topologies, GAMA integrates some high-level and optimized built-in actions dedicated to the movement of agents (defined for example in the *moving* skill presented in Fig. 2): it is for example possible to ask an agent to move in the direction of a given target (a coordinate in the environment) according to a given *topology*.

In order to optimize the spatial computation, and more particularly the spatial query, GAMA uses a dynamic Quadtree structure that is updated according to the movement of the agents. Regarding the computation of shortest paths on graphs, GAMA proposes different algorithms such as Dijkstra, Floyd Warshall, or NBA* [53]. At last, several algorithms are also available to compute shortest paths and distances on a grid (with the possibility to consider obstacles) such as Dijkstra, A*, or JPS [39].

3.4 Towards more complex models

3.4.1 High performance computing

These last years, more and more works propose to carry out spatial simulation at large scale with a one-to-one representation of people. For example, Pires and Crooks [54] simulates the

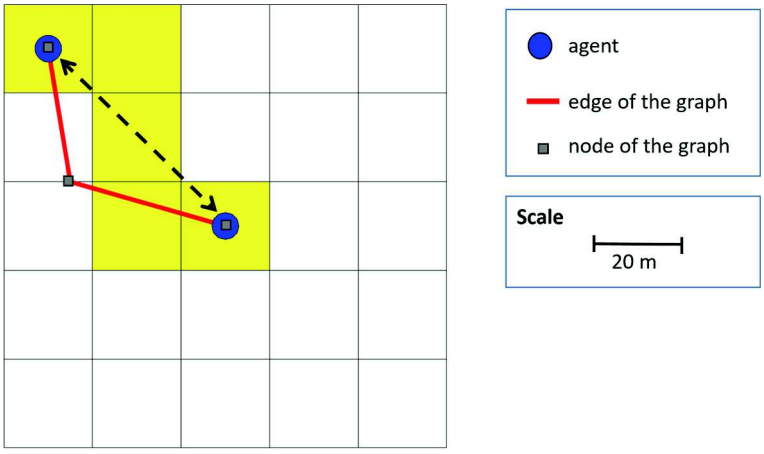


Fig. 4 Computation of distances between two agents (blue circle) according to different GAMA *topologies*: distance with the continuous *topology* (Euclidean): 56.6 m; distance with the grid *topology* (yellow cells): 4 cells, distance with the graph *topology* (red lines representing roads): 66.8 m

emergence of riots in an informal settlement in Nairobi (Kenya) at full scale with 235,000 agents. Another example is the model proposed by Cotineau et al. [21] that simulates the dietary dynamics in the Paris Region with 8 million of agents.

Carrying simulations, with a large number of agents and/or very complex dynamics, requires many computations. A common way to deal with this issue is to use parallel computation.

Like some other platforms, GAMA enables to run several simulations in a multi-thread mode (one thread per simulation). This is particularly useful when running a model exploration plan either using an *experiment plan* agent of type *batch* or using the *headless* mode. Indeed, like Netlogo with its behavior space and behavior search, GAMA provides modelers with a specific type of *experiment* called *batch* that allows to run several simulations with potentially different parameter sets without displaying the step-by-step evolution of the simulation. The method used to explore the parameter set can be chosen among several available (exhaustive search, hill climbing, tabu search, simulated annealing, or genetic algorithm). Concerning the headless mode, it enables to run GAMA simulations without any Graphical User Interface.

In addition to these capabilities, two main external modules concerning the exploration of models were developed:

- *GAMAR* is a *R* package that enables from the *R* software environment to define experiment plans, to run GAMA simulations and to analyze simulation results.
- The *OpenMole plugin* allows to define and run on supercomputers adaptive exploration plans by using the OpenMole framework [58]. OpenMole is an open-source framework that allows users to define experiment plans with advanced parameter space exploration algorithms and to automatically deploy simulations on a computer grid or a cluster.

GAMA allows as well to go further by enabling to distribute the execution of agents (or some specific computation in the model) on several threads. The concurrency between agents (from multi-threaded concurrency to complete parallelism within a species of agents or between the targets of an ask statement) can be controlled through simple GAML keywords, hiding to modelers all the complexity behind the parallel computation.

3.4.2 Co-modeling and multi-simulation

As models tend to be more and more complex and descriptive, it becomes more and more difficult to reuse them in different contexts. A solution to tackle this issue is to use a modular approach [10], in which the goal is develop complex models as a composition of numerous sub-models, which can be developed independently and used in a stand-alone way. Several works such as [55] have shown the interest of using such approach for developing complex spatial models.

GAMA is particularly adapted to manage this coupling of models by its native support of co-modeling and co-simulation [26]. A *co-model* [41] is a model composed of several sub-models, that could be themselves composed of several sub-models. In GAMA, simulations are agents, and their behavior corresponds to the execution of a step of the simulation. The co-modeling is thus simply the idea of creating an agent simulation inside another simulation and to schedule it as any other agent of the simulation. This extends the idea introduced in *experiment plan* agents that are able to create and run *simulation* agents (i.e. models). One of the main advantages of the co-modeling is to facilitate the development of complex models by adopting an incremental approach (implementing sub-model by sub-model), in which each sub-model can be independently tested and validated. In addition, it could promote the re-usability of a sub-model among several projects.

Also using the agent-oriented architecture of GAMA, a feature, which stems from the multi-thread capacity of GAMA, is the possibility to run several simulations, with potentially different parameter values or random generator seeds, from the Graphical User Interface (GUI). To this purpose, the description of the execution of the simulation (in an *experiment plan*) should simply include the creation of several simulation agents (following the GAML classical syntax) with different global parameters. The creation of several simulations and their observation in the GUI allows to explore a model in a simple and intuitive way: the modelers can directly, step by step, see the impact of the randomness, or of the parameter values on the simulation. Figure 5 shows an example of use of a multi-simulation: 4 simulations of a classic foraging ant model with different parameter sets (concerning the number of ants, the evaporation rate and the diffusion rate of the pheromone) are run in parallel.

3.4.3 Advanced visualization

For many modelers, visualization is not only their first point of entry to build a model, but also an increasingly prevalent way of designing, verifying, and validating models [25]. Visualization gives the possibility to intuitively check agent states as well as collective or emerging structures [36]. The round-trip between writing and the visualization of a model is an important part of the daily life of many modelers [35]. In addition, in order to speak to a large and heterogeneous audience, it is necessary to represent different level of structures from the same model. GAMA enables to easily differentiate a model from its visualization by offering the possibility to define different displays per simulation composed of several layers.

As pointed out by [4], the use of 3D is still uncommon in the world of agent-based simulation, whereas using the third dimension can deeply increase the immersive

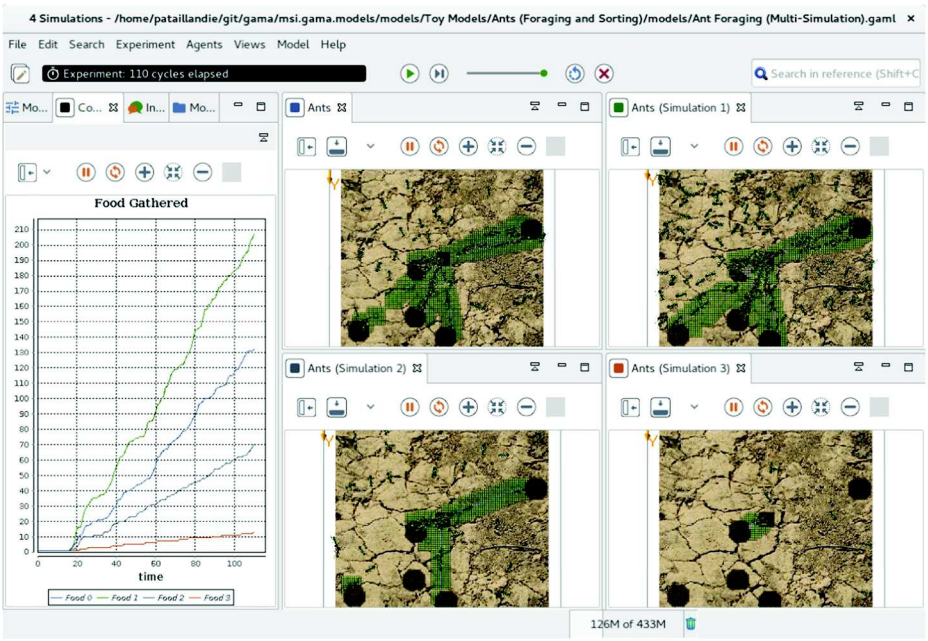


Fig. 5 Example of use of multi-simulation: 4 simulations concerning ant foraging are run is parallel with 4 different sets. The left chart shows in real time the results of the 4 simulations in terms of food gathered

property of a simulation, but also allows to visualize additional information like in Time Geography [7]. GAMA offers many advanced visualization features regarding the definition of 3D visualization with textures, complex lighting and customized dynamic cameras (see Fig. 6). GAMA also handles the rendering of large-scale models, being able to display in real time several thousands of agents.

3.4.4 Advanced agent architecture: BDI

In parallel to improvements related to the definition of more complex simulation environments through the use of many data sources, one of the major works of these last years concerned the possibility to define more complex agents with cognitive, social, and emotional capabilities.

The interest of using cognitive and emotional agents in social simulations has been shown by many works [1, 15, 67] and more and more models tend to integrate such complex agents. In order to facilitate the definition of such agents, GAMA integrates an agent architecture based on the BDI (Belief, Desire, Intention) paradigm [16]. This paradigm proposes a straightforward formalization of the human reasoning through intuitive concepts (beliefs, desires and intentions). The developed architecture has been created to be as complete as possible and available to a non-expert public. It gives agents a belief base, a desire base, an intention base, a reasoning engine, a social engine, and an emotional engine [13, 14, 17].

The agent architecture also provides a new way to describe the agent behavior that is closer to the classical Multi-Agent System Paradigm [68] and its Perception-Decision-Action loop. Modelers can thus define what agents will perceive at the beginning of each step and how these percepts affects its mental state (creation of new beliefs, belief revision…). To gives agents some inference capabilities, a set of belief and desire update rules can be defined. Finally, the way agents act in the environment will be defined through a set of plans. In addition, to give more flexibility to the modeler, classical reflex blocks can be added. Some works have already shown that the use of this architecture eases the modeling of human beings [3, 64].

Fig. 6 Example of displays that can be defined with GAMA: used of 3D obj files with textures for cars, a georeferenced satellite image as background, and a building shapefile with 3D extrusion and textures for buildings

4 Examples of applications

The GAMA platform is now used in many application cases. We mention below only a few representative recent works that were published after 2014.

4.1 Traffic and transportation simulation

Many works have shown the interest of using agent-based modeling for traffic and transportation issues [20]. To ease the work of modelers, several agent-based frameworks dedicated to traffic or transportation such as MATSim [8], SUMO [43] or Agentpolis [42] have been developed. If these frameworks are well-fitted for the study of normal traffic conditions, their adaptation to more specific applications (e.g. non-normative behaviors, car accidents, mobility practice evolution) can be very fastidious, in particular for non-computer scientists. Indeed, for modelers without high level programming skills, adapting these platforms to specific application contexts is out of reach as it requires to write code in JAVA or C++. GAMA integrates many features simplifying the design of traffic and transportation models. It is a true alternative to these frameworks, which explains why it has been used for several projects dealing with such systems.

A good example is the GaMiroD model [29] that deals with daily mobility in a city. This model proposes a realistic representation of the transportation infrastructure (roads, public transportation...) and to model every inhabitants of a city and their daily activities. At each time step (1 simulation step = 30 s), each inhabitant will then be able to move and carry out his/her own activities. For the movement, each inhabitant will be able to choose his/her mean of transport (foot, bus, or car), which will impact the global traffic. The model was used for different applications such as the evaluation of air pollution in the city of Grenoble in France with 400,000 agents representing the inhabitants, 68,000 the buildings, 28,000 the roads, and 750 agents the bus stations and bus services. This model illustrates the capacity of GAMA to manage a large number of agents and to integrate a high quantity of data (shapefiles of roads, public transportations and buildings, information about inhabitants…).

Another example is the MOSAIIC Model [24]. This model proposes a detailed representation of the driver operational behaviors at a low time scale (1 simulation step = 1 s). In particular, it allows to take into account the road infrastructures and traffic signals (through classic GIS data such as Open Street Map data), the change of lanes of the drivers and their respects of norms (traffic signals, priorities...). It also takes into account the generation of traffic jams and their impact on the driver behaviors. The model was used to simulate the traffic of the city of Rouen in France (more than 70,000 drivers per hours) during peak hours and to test some evacuation scenarios.

Recently, GAMA has been used to study the interaction of social, economic and physical characteristics of urban areas to understand how people use and experience cities [5]. In [33], the model simulates the daily life of the inhabitants of a city, in particular their mobility. It uses a tangible interface to let stakeholders interact with the simulation. With this interface, stakeholders can modify the city plan and get a direct feedback of the impact of these modifications on the simulation. Another example is [34] that proposes a novel information visualization approach developed and deployed in the state of Andorra. This framework analyzes and represents the flow of people through a multi-level interactive and tangible agent-based visualization.

Many other traffic and transportation models have been produced these last years with GAMA concerning a wide range of applications such as the management of the road infrastructure [12] or the pedestrian traffic [59].

4.2 Risk mitigation

Risk mitigation is an important topic, in particular for urban areas and many projects chose to tackle this issue with agent-based modeling and simulation using GAMA. Among these projects, some use the capabilities of GAMA concerning the development of serious games in which participants interact with a rich and credible model.

For example, Sprite [61] allows a participant to play the role of the mayor of the Oleron island in France. The goal of the participant is to find an appropriate balance between popularity, economy, attractiveness, ecology and safety, in particular concerning the submersion risk. Sprite proposes a precise representation of the island through GIS data and integrates an advanced submersion model. In addition, it benefits from the capacity of GAMA regarding user interaction with the simulation in runtime.

Another example that also deals with the management of the submersion risk is Littosim [9]. In Littosim, participants play the role of the land planning manager of a coastal municipalities prone to marine submersion risk. It integrates several sub-models such as water rise and socio-economical evolution models. One of the particularities of Littosim enabled by the features of GAMA is to allow the use of distributed controls: each land planner can interact with its own view of the simulation through a personal computer tablet.

In addition to these models that take the form of a serious game, another example concerns the management of construction projects under risky conditions. The Stochastic Multi-agent simulAtion for Construction projeCt (SMACC) model [62, 63] proposes to assess risk consequences for each project stakeholder. This model, that uses the capacities of GAMA concerning the complex modeling of the different stakeholder relations and behaviors, has been applied on different construction projects: the building of the crossroad of Bab El Karmadine at Tlemcen (Algeria) and of a nursing training institute in the South-West of France.

4.3 Crisis management

An interesting example concerns the reproducing and exploring of past disasters. The ARCHIVES geo-historical model [30] aims at simulating the management of flood in Hà Nội (Việt Nam) in 1926. This model uses reconstructed data (GIS and data linked to the event) from the French and Vietnamese archives. The goal was to provide historians with a tool to not only replay historical events, but also to explore what could have happened in alternative "what-if" scenarios. This model is another example of use of heterogeneous data (grid file, shapefiles, csv files, etc.) and of several representations of the same space (grid representation for the flood model, continuous representation for the people agent movement, etc.).

Other examples could have been cited such as the management of evacuation sign in case of Tsunami in Vietnam [46] or the spreading of dengue fever in Delhi [48] and in Vietnam [52].

As far as evacuation models are concerned, GAMA is also used in several projects aiming at studying human behaviors facing a crisis (e.g. forest or building fires). In particular, [60] aims at modeling the effects of emotions and social bounds on the human behavior in crisis situation. This work is mainly theoretical and aims at building a model of emotion and reasoning. Conversely, models built in the SWIFT project [2] start from actual interviews of victims after Victoria bushfires (Australia) and attend to build the evacuation/house defense

model from these quantitative data, with a particular focus on the distinction between objective and subjective data. After a first simple model based on human behavior built using a Finite State Machine agent architecture (behavior is characterized by state automata, with specific actions in each state and transition condition between states), the model has been extended to use the BDI architecture of GAMA [3].

4.4 Resource management

The GAMA platform is particularly well-suited to represent socio-environmental systems as these systems require by nature to take into account many different kinds of agents and dynamics and to include a huge amount of (geographical) data.

The most interesting example concerning the use of GAMA to represent socio-environmental systems is the MAELIA model [65]. This model aims at assessing at watershed level a wide range of scenarios regarding water and land use management strategies in combination with global changes. In particular, it proposes to represent the dynamic interactions between human activities (farming practices), ecological processes (hydrology and crop growth), and governance systems (water regulations and releases from dams) at fine spatio-temporal resolutions. This model is a good illustration of the capacity of GAMA for dealing with complex models (composed of more than 20 sub-models, several gigabytes of data...). It also takes advantage of GAMA built-in capabilities in terms of multi-criteria decision making process.

Another example is the PASHAMAMA model [18] that aims at evaluating the combined and retroactive effects of colonization of the northern part of "El Oriente", the Amazonian region of Ecuador, from the 70's following the oil exploitation of the area and the oil pollution hazards on the population. The model also proposes scenario based exploration in order to simulate the actual and possible future situations to assist stakeholder decision making process. Once again, this model serves as an example for GIS data importation in GAMA, but also uses Digital Elevation Model data to provide a better simulation of the water flow dynamics.

Finally, the ARCHEM project [37] proposes a methodology to visualize the fine-scale sediment transport of a river in order to improve the management decisions regarding sediment recharge operations. It provides an immersive 3D representation of the watershed coupled with a morphodynamic model and focuses on the spatial and temporal impact (travel speed, particle size and channel geometry) of these operations based on several scenarios for sediment re-injection in the Rhone river.

5 Conclusion

GAMA is a collaborative open-source effort.[1] These last years have seen particularly important and sometimes spectacular improvements in its development, in response to new needs in the modeling community (parallel exploration of models, side-by-side display of alternative scenarios, more realistic modeling of human decision processes, and so on). It has also proposed new possibilities to design models and simulations (e.g. co-modeling).

GAMA has now reached a level of maturity and stability that makes it, nowadays, the platform of choice for a large community of users and in a wide range of domains.

[1] https://github.com/gama-platform/gama

GAMA is however in continuous evolution and new features are constantly added to the platform. One of the new features that is currently under development concerns the possibility to save the different steps of a simulation to give the possibility to backtrack to previous steps or to replay simulations. If this feature is already implemented in some platforms such as CORMAS, its usability for more descriptive models composed of thousands of agents with complex variables (3D geometries, graph....) is a real challenge.

As models tend to be more and more descriptive and detailed, an important issue concerns the initialization of the simulation and more particularly how the population of agents (with their attributes) is created. Indeed, the attributes of social agents (e.g., individuals, households or institutions) has often a deep impact on their behaviors. Consequently, being able to generate synthetic population conform to the data available on the real population becomes a necessity and a concern for many modelers. In order to support modelers in their population generation task, we are currently working on a plug-in for GAMA to be able to use the Gen* toolkit dedicated to population generation and spatialization [19]. This plugin will allow to generate a synthetic population from data (census, sample of the population, GIS data…) through simple GAML operators.

A last feature that is already tested internally (i.e. not available to the general public so far) concerns the possibility to run GAMA "in the cloud". As a matter of fact, nowadays, in order to run even a simple model, a user has to download a "complete" version of GAMA adapted to its working environment, to install it (with a correct Java virtual machine) and to maintain it over time (by downloading updates or new versions, especially when Java virtual machines or operating systems are updated). Even for experienced users, this can prove a source of difficulty. Not to mention the fact that, with the independent development of new GAMA plug-ins, some models may require additional installation steps. In order to ease the sharing and use of models, we are currently developing an online version of GAMA, thanks to which modelers will be able to write and simulate models within a web browser (light client) without having to install on their computer a complete version of GAMA. This development, targeted for the 2019 release of GAMA, will probably also change the way models can be collaboratively written [32, 50].

Appendix

GAMA, which is under a GNU General Public License, can be downloaded from the GAMA Website: http://www.gama-platform.org. It is available in 32 and 64 bits for Windows, OS-X and Linux. GAMA requires a Java Virtual Machine (1.8) to run, but the website proposes a version of GAMA with an embedded JVM. A library of more than 300 models is provided with GAMA: this library contains models presenting how to use the different features of GAMA (data importation, agent movement, 3D display, etc.), classic toy models (ant foraging, boids, Schelling, SugarScape…) and tutorials. The Website of GAMA provides a complete documentation on the platform. It provides as well a series of tutorials to learn the bases of GAML. At last, an online user group[2] is available to ask questions about the platform.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

[2] https://groups.google.com/forum/#!forum/gama-platform

References

1. Adam C, Gaudou B (2016) BDI agents in social simulations: a survey. Knowl Eng Rev 31:207–238
2. Adam C, Gaudou B (2017) Modelling human Behaviours in disasters from interviews: application to Melbourne bushfires. Journal of Artificial Societies and Social Simulation 20
3. Adam C, Taillandier P, Dugdale J (2017) Comparing agent architectures in social simulation: Bdi agents versus finite-state machines. In: Hawaii International Conference on System Sciences (HICSS)
4. Allan RJ (2009) Survey of agent based modelling and simulation tools. Tech rep
5. Alonso, L., Zhang, Y. R., Grignard, A., Noyman, A., Sakai, Y., ElKatsha, M., ... & Larson, K.: Cityscope: a data-driven interactive simulation tool for urban design. Use case volpe. In: International Conference on Complex Systems, pp. 253–261, Springer (2018)
6. Amblard, F., Bouadjio-Boulic, A., Guti'errez, C.S., Gaudou, B.: Which models are used in social simulation to generate social networks? a review of 17 years of publications in jasss. In: Winter Simulation Conference (WSC), 2015, pp. 4021–4032. IEEE (2015)
7. Bach, B., Dragicevic, P., Archambault, D., Hurter, C., Carpendale, S.: A descriptive framework for temporal data visualizations based on generalized space-time cubes. In: Computer Graphics Forum, vol. 36, pp. 36–61. Wiley Online Library (2017)
8. Balmer, M., Rieser, M., Meister, K., Charypar, D., Lefebvre, N., Nagel, K., Axhausen, K.: Matsim-t: Architecture and simulation times. Multi-Agent Systems for Traffic and Transportation Engineering pp. 57–78 (2009)
9. Becu, N., Amalric, M., Anselme, B., Beck, E., Bertin, X., Delay, E., Long, N., Manson, C., Nicolas, M., Pignon-Mussaud, C., Rousseaux, F.: Participatory simulation of coastal flooding: building social learning on prevention measures with decision-makers. In: 8th International Congress on Environmental Modelling and Software, pp. 1–14. Toulouse, France (2016)
10. Bell AR, Robinson DT, Malik A, Dewal S (2015) Modular abm development for improved dissemination and training. Environ Model Softw 73:189–200
11. Bellifemine, F., Poggi, A., Rimassa, G.: Jade: a fipa compliant agent development environment. In: Proceedings of the fifth international conference on Autonomous agents, pp. 216–217. ACM (2001)
12. Bhamidipati, S., Van der Lei, T., Herder, P.: A layered approach to model interconnected infrastructure and its significance for asset management. European Journal of Transport and Infrastructure Research (EJTIR), 16 (1), 2016 (2016)
13. Bourgais M, Taillandier P, Vercouter L (2016) An agent architecture coupling cognition and emotions for simulation of complex systems. In, Social Simulation Conference
14. Bourgais, M., Taillandier, P., Vercouter, L.: Enhancing the behavior of agents in social simulations with emotions and social relations. In: The 18th Workshop on Multi-agent based Simulation-MABS 2017 (2017)
15. Bourgais M, Taillandier P, Vercouter L, Adam C (2018) Emotion modeling in social simulation: a survey. Journal of Artificial Societies and Social Simulation 21
16. Bratman, M.: Intentions, plans, and practical reason. (1987)
17. Caillou, P., Gaudou, B., Grignard, A., Truong, C.Q., Taillandier, P.: A simple-to-use bdi architecture for agent-based modeling and simulation. In: Advances in Social Simulation 2015, pp. 15–28. Springer (2017)
18. Chapotat, W., Houssou, L., Bouadjio Boulic, A., Maestripieri, N., Lerigoleur, E., Gaudou, B., Saqalli, M.: An agent-based model of the amazonian forest colonisation and oil exploitation: the oriente study case (poster). In: *S. sauvage*, J.M. Sanchez Perez, A.E. Rizzoli (eds.) International Environmental Modelling and Software Society (iEMSs), Toulouse, France, vol. 5, pp. 1335–1335. International Environmental Modelling & Software Society, http://www.iemss.org (2016)
19. Chapuis, K., Taillandier, P., Renaud, M., Drogoul, A.: Gen*: a generic toolkit to generate spatially explicit synthetic populations. International Journal of Geographical Information Science pp. 1–17 (2018)
20. Chen B, Cheng HH (2010) A review of the applications of agent technology in traffic and transportation systems. IEEE Trans Intell Transp Syst 11(2):485–497
21. Cottineau, C., Perret, J., Reuillon, R., Rey-Coyrehourcq, S., Vallée, J.: An agent-based model to investigate the effects of social segregation around the clock on social disparities in dietary behaviour. In: CIST2018-Representing territories (2018)
22. Crooks, A., Malleson, N., Wise, S., Heppenstall, A.: big data, agents and the city. Big Data for Regional Science (2017)
23. Crooks, A.T., Castle, C.J.: The integration of agent-based modelling and geographical information for geospatial simulation. In: Agent-based models of geographical systems, pp. 219–251. Springer (2012)
24. Czura, G., Taillandier, P., Tranouez, P., Daudé, E.: Mosaiic: City-level agent-based ´ traffic simulation adapted to emergency situations. In: Proceedings of the International Conference on Social Modeling and Simulation, plus Econophysics Colloquium 2014, pp. 265–274. Springer (2015)
25. Dorin A, Geard N (2014) The practice of agent-based model visualization. Artificial life 20(2):271–289

26. Drogoul A, Huynh NQ, Truong QC (2016) Coupling environmental, social and economic models to understand land-use change dynamics in the mekong delta. Frontiers in Environmental Science 4:19
27. Drogoul, A., Vanbergue, D., Meurisse, T.: Multi-agent based simulation: Where are the agents? In: International Workshop on Multi-Agent Systems and Agent-Based Simulation, pp. 1–15. Springer (2002)
28. Edmonds, B., Moss, S.: From kiss to kids–an anti-simplisticmodelling approach. In: International workshop on multi-agent systems and agent-based simulation, pp. 130–144. Springer (2004)
29. Fosset P, Banos A, Beck E, Chardonnel S, Lang C, Marilleau N, Piombini A, Leysens T, Conesa A, Andre-Poyaud I, Thevenin T (2016) Exploring intra-urban accessibility and impacts of pollution policies with an agent-based simulation platform: Gamirod. Systems 4(1):5
30. Gasmi, N., Grignard, A., Drogoul, A., Gaudou, B., Taillandier, P., Tessier, O., An, V.D.: Reproducing and exploring past events using agent-based geo-historical models. In: International Workshop on Multi-Agent Systems and Agent-Based Simulation, pp. 151–163. Springer (2014)
31. Gaudou, B.: Toward complex models of complex systems - one step further in the art of agent-based modelling. Habilitation `a diriger des recherches, Universit Toulouse 1 Capitole (2016)
32. Gaudou, B., Marilleau, N., Ho, T.V.: Toward a methodology of collaborative modeling and simulation of complex systems. In: Intelligent Networking, Collaborative Systems and Applications, pp. 27–53. Springer (2010)
33. Grignard, A., Alonso, L., Taillandier, P., Gaudou, B., Nguyen-Huu, T., Gruel, W., Larson, K. : The Impact of New Mobility Modes on a City: A Generic Approach Using ABM. In: Unifying Themes in Complex Systems IX. ICCS 2018. Springer proceedings in complexity, pp; 272-280, Springer (2018)
34. Grignard, A., Macià, N., Alonso Pastor, L., Noyman, A., Zhang, Y., & Larson, K.: Cityscope andorra: a multilevel interactive and tangible agent-based visualization. In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (2018)
35. Grignard, A., Drogoul, A.: Agent-based visualization: a real-time visualization tool applied both to data and simulation outputs (2017)
36. Grignard, A., Drogoul, A., Zucker, J.D.: Online analysis and visualization of agent based models. In: International Conference on Computational Science and Its Applications, pp. 662–672. Springer (2013)
37. Grignard, A., Fantino, G., Lauer, J.W., Verpeaux, A., Drogoul, A.: Agent-based visualization: A simulation tool for the analysis of river morphosedimentary adjustments. In: International Workshop on Multi-Agent Systems and Agent-Based Simulation, pp. 109–120. Springer (2015)
38. Grignard, A., Taillandier, P., Gaudou, B., Vo, D. A., Huynh, N. Q., & Drogoul, A.: GAMA 1.6: Advancing the art of complex agent-based modeling and simulation. In: International Conference on Principles and Practice of Multi-Agent Systems, pp. 117–131. Springer (2013)
39. Harabor DD, Grastien A et al (2014) Improving jump point search. In, ICAPS
40. Heppenstall AJ (2011) Crooks. A.T., See, L.M., Batty, M.: Agent-based models of geographical systems. Springer Science & Business Media
41. Huynh, Q.N.: Comodels, engineering dynamic compositions of coupled models to support the simulation of complex systems. Ph.D. thesis, Paris 6 (2016)
42. Jakob, M., Moler, Z.: Modular framework for simulation modelling of interaction-rich transport systems. In: Proceedings of the 16th IEEE Intelligent Transportation Systems Conference (ITSC 2013) (2013)
43. Krajzewicz D, Erdmann J, Behrisch M, Bieker L (2012) Recent development and applications of SUMO - simulation of urban MObility. International Journal On Advances in Systems and Measurements 5(3&4):128–138
44. Kravari K, Bassiliades N (2015) A survey of agent platforms. Journal of Artificial Societies and Social Simulation 18(1):11
45. Laatabi A, Marilleau N, Nguyen-Huu T, Hbid H, Babram MA et al (2018) Odd+ 2d: An odd based protocol for mapping data to empirical abms. Journal of Artificial Societies and Social Simulation 21(2):1–9
46. Le, V.M., Chevaleyre, Y., Vinh, H.T., Zucker, J.D.: Hybrid of linear programming and genetic algorithm for optimizing agent-based simulation. application to optimization of sign placement for tsunami evacuation. In: Computing & Communication TechnologiesResearch, Innovation, and Vision for the Future (RIVF), 2015 IEEE RIVF International Conference on, pp. 138–143. IEEE (2015)
47. Le Page, C., Bousquet, F., Bakam, I., Bah, A., Baron, C.: Cormas: A multiagent simulation toolkit to model natural and social dynamics at multiple scales. In: Proceedings of Workshop" The ecology of scales", Wageningen (The Netherlands) (2000)
48. Maneerat S, Daudé E (2016) A spatial agent-based simulation model of the dengue vector aedes aegypti to explore its population dynamics in urban areas. Ecol Model 333:66–78
49. Minar, N., Burkhart, R., Langton, C., Askenazi, M., et al.: The swarm simulation system: a toolkit for building multi-agent simulations (1996)
50. Nguyen, T.K., Gaudou, B., Ho, T.V., Marilleau, N.: Application of pams collaboration platform to simulation-based researches in soil science: The case of the micro-organism project. In: Computing and Communication Technologies, 2009. RIVF'09. International Conference on, pp. 1–8. IEEE (2009)

51. North MJ, Collier NT, Ozik J, Tatara ER, Macal CM, Bragen M, Sydelko P (2013) Complex adaptive systems modeling with repast simphony. Complex adaptive systems modeling 1(1):3
52. Philippon D, Choisy M, Drogoul A, Gaudou B, Marilleau N, Taillandier P, Truong CQ (2016) Exploring trade and health policies influence on dengue spread with an agent-based model. In: international workshop on multi-agent-based simulation (MABS). Singapore
53. Pijls W, Post H (2009) Yet another bidirectional algorithm for shortest paths. Tech. rep, Econometric Institute Research Papers
54. Pires B, Crooks AT (2017) Modeling the emergence of riots: a geosimulation approach. Comput Environ Urban Syst 61:66–80
55. Pumain, D., Reuillon, R.: An incremental multi-modelling method to simulate systems of cities evolution. In: Urban Dynamics and Simulation Models, pp. 57–80. Springer (2017)
56. Repenning, A.: Making programming more conversational. In: Visual Languages and Human-Centric Computing (VL/HCC), 2011 IEEE Symposium on, pp. 191–194. IEEE (2011)
57. Resnick, M.: Starlogo: An environment for decentralized modeling and decentralized thinking. In: Conference companion on Human factors in computing systems, pp. 11–12. ACM (1996)
58. Reuillon R, Leclaire L, Rey-Coyrehourcq S (2013) Openmole, a workflow engine specifically tailored for the distributed exploration of simulation models. Futur Gener Comput Syst 29(8):1981–1990
59. Rose J, Ligtenberg A, Van der Spek S (2014) Simulating pedestrians through the innercity: An agent-based approach. UAB Press
60. Ta, X.H., Longin, D., Gaudou, B., Ho, T.V.: Impact of group on the evacuation process - theory and simulation. In: L. De Raedt, Y. Deville, M. Bui, T.T.D. Lin (eds.) Symposium on Information and Communication Technology (SoICT), Hue, Vietnam, pp. 350–357. ACM DL (2015)
61. Taillandier, F., Adam, C.: Games ready to use: A serious game for teaching natural risk management. Simulation & Gaming p. 1046878118770217 (2018)
62. Taillandier F, Taillandier P, Hamzaoui F, Breysse D (2016) A new agent-based model to manage construction project risks–application to the crossroad of bab el karmadine at Tlemcen. Eur J Environ Civ Eng 20(10):1197–1213
63. Taillandier F, Taillandier P, Tepeli E, Breysse D, Mehdizadeh R, Khartabil F (2015) A multi-agent model to manage risks in construction project (smacc). Autom Constr 58:1–18
64. Taillandier, P., Bourgais, M., Caillou, P., Adam, C., Gaudou, B.: A bdi agent architecture for the gama modeling and simulation platform. In: MABS 2016 Multi-Agent-Based Simulation (2016)
65. Therond, O., Sibertin-Blanc, C., Lardy, R., Gaudou, B., Balestrat, M., Hong, Y., Louail, T., Panzoli, D., Sanchez-Perez, J.M., Sauvage, S., et al.: Integrated modelling of socialecological systems: The maelia high-resolution multi-agent platform to deal with water scarcity problems. In: 7th International Environmental Modelling and Software Society (2014)
66. Tisue, S., Wilensky, U.: Netlogo: A simple environment for modeling complexity. In: International conference on complex systems, vol. 21, pp. 16–21. Boston, MA (2004)
67. Truong, Q.C., Taillandier, P., Gaudou, B., Vo, M.Q., Nguyen, T.H., Drogoul, A.: Exploring agent architectures for farmer behavior in land-use change. a case study in coastal area of the vietnamese mekong delta. In: International Workshop on Multi-Agent Systems and Agent-Based Simulation, pp. 146–158. Springer (2015)
68. Woolridge M, Wooldridge MJ (2001) Introduction to multiagent systems. John Wiley & Sons, Inc., New York, NY, USA