Cormas, an Agent-Based simulation platform for coupling human decisions with computerized dynamics

Pierre Bommel[1], Nicolas Becu[2], Christophe Le Page[3], François Bousquet[3]

Abstract **This paper aims at presenting the new functionalities of Cormas, a generic Agent-Based Modeling (ABM) platform dedicated to common-pool resources management. As free software, Cormas is used by an international community of researchers willing to understand the relationships between societies and their environment. It is intended to facilitate the design of ABM as well as the monitoring and analysis of simulation scenarios. To meet the increasing demand of our community of practice, the Cormas development has taken an innovative direction oriented towards the collective design of models and interactive simulation. In accordance with the principles of participatory methods and serious games, many experiments developed with Cormas combine two layers of complexity: the natural dynamic of the system, simulated by the computer, and the stakeholders' interactions and decisions played by the actors. Between these two extremes, a range of intermediate situations exists where some decisions are human and others are computer-specified. The term hybrid agent simulation covers these intermediary situations. The main idea is to enable the stakeholders to interact with the execution of a simulation by modifying the behavior of the agents and the way they use the resources. Thus, it is possible to collectively explore scenarios to better understand how a desired situation may be reached. This may feed back into the collective design of the model. As our intention is to involve more deeply the stakeholders into the modeling process, it is necessary to have an easily changeable tool to act on the simulation and to modify the conceptual model on the fly. After having explained the purpose and the philosophy of the Companion Modeling, this paper presents how the Cormas functionalities (asymmetry of information, agent manipulation, modification of behavior, stepping back and distributed**

[1] CIRAD - UPR GREEN, France & Universidad de Costa Rica, CIEDA, Costa Rica
[2] CNRS - UMR 7266 LIENSs, La Rochelle, France
[3] CIRAD - UPR GREEN, Montpellier, France

simulation) are put into practice through three experiments with stakeholders facing actual environmental challenges.

Keywords: Participatory modeling, Interactive simulation, Natural resources management, Stakeholders involvement, Agent-Based Model.

1 Introduction

Cormas (for Common-Pool Resources and Multi-Agent Systems) is an Agent-Based Modeling (ABM) platform dedicated to natural and common-pool resources management [1]. As an open-source and free software, it is used by an international community of researchers [2] willing to understand the relationships between societies and their environment. Cormas is intended to facilitate the design of ABM as well as the monitoring and analysis of agent-based simulation scenarios. Indeed, the purpose of ABM is to understand how independent entities can interact, be coordinated and may co-evolve, while producing effects on the system as a whole. An agent can be described as an autonomous entity that has the capacity to adapt when its environment changes, and conversely his actions may modify the environment. A multi-agent system is made up of a set of entities that act at the same time, i.e. several agents that perform activities, share common resources and communicate with each other. As they are centered on the individual, ABMs enable the user of a simulation to assume the role of an agent and, for example, to "think like a wolf, a sheep or a fly" [3]. In the purpose to deal with natural renewable resources management, Cormas is mainly oriented towards the representation of interactions between stakeholders and their environment with a specific focus on the interactions between natural and social dynamics.

From recent years, the development of Cormas has taken an innovative direction more oriented towards participatory modeling, i.e. collective design of models and interactive simulation. This new orientation has been taken to meet the increasing demand of our community of practice. Indeed, in parallel with the development of Cormas, a modeling methodology called the ComMod approach - for Companion-Modeling - [4], [5] has been setup and formalized. If the classic use of simulation is for prediction, this is not the option we have chosen because the long-term economic and social future cannot be predicted, although it can be partially decidable. We assume that stakeholders can "decide" long-term objectives on the basis of a shared conception of how the present situation should evolve. It is thus possible to explore scenarios collectively to better understand if the desired situation may be reached. The underlying model for the simulations depends on the way the actors are represented. Two major types of representation can be distinguished: (i) virtual agents performing predefined activities in a computerized ABM, or (ii) human agents playing their role in a role-playing game

(RPG). Even if it is not in the classic sense, a RPG can be seen as a representation of the world, i.e. a model. But between these two extremes, a range of intermediate situations exists where some decisions are human and others are computer-specified. The term hybrid agent simulation model covers all these intermediary situations [6]. The mediation approach presupposes that the stakeholders are well informed of the issues dividing them and of the fact that they all have an interest in solving the original problem.

For that purpose, we are developing Cormas towards two directions: 1) to facilitate the collective design and implementation of ABMs and 2) to enable the development of interactive simulations in order to let the users participating actively, alone or with others, in the execution of a scenario. As a generic framework, Cormas allows the user to specialize and refine pre-defined entities for his own model. But this new version is particularly suitable for:

· Changing the parameters of one or a set of agents,

· Manipulating an agent directly with the mouse on the computer: moving it on a precise location, sending him specific messages (predetermined behavior) or even designing new behaviors thanks to an activity diagram editor that is directly interpreted by the agent.

· Stepping back in time of a simulation and restarting the interactive simulation to a previous state (bifurcations), or replaying forward a previously stored simulation,

· Distributing a simulation on several machines, monitoring the evolution of a remote simulation and remotely manipulating the entities,

· Displaying particular points of view of the simulated landscape, opening several zooms and enabling specific "Habitus" for the available points of view.

Our intention is to involve more deeply the stakeholders into the modeling process. Because, if adaptive management has become a buzzword, in practice people's participation is often just a catchy expression used by scientists to justify the process of extracting information [7]. On the contrary, participatory modeling should encourage producing models that are able to promote mutual recognition of perceptions, knowledge appropriation and finally collective decision-making. For this it is necessary to have an easily changeable tool to act on the simulation and to modify the conceptual model on the fly. Our various field experiments have shown the need to create continuity between the conceptual model and its implementation. We hope this new version will contribute to achieve this goal.

2 Cormas' overview

As a framework that proposes predefined classes and a set of visualization tools, the Cormas environment is intended to facilitate the implementation of ABM as well as the monitoring and analysis of simulations. It uses VisualWorks

[8], a programming environment based on Smalltalk, one of the first purely object oriented language. A framework means that the modeler implements its model by specializing predefined classes in Cormas. These general classes contain attributes and generic methods that can be reused by the specialized classes. Mainly, three generic types of entities are available: the “social agent” group, the “spatial” group and the “passive entity” group that contains the other kinds of entities such as messages, land covers, strategies, etc. Thus, when creating a specific agent (a Forager for example), the modeler can define it from the “AgentLocation” generic class. This new agent will be automatically registered by the scheduler and will be also able to move on the space or to perceive his neighbors by calling the generic methods (#moveTo: and #perceive) defined at the level of the AgentLocation super-class. The following class diagram presents how a simple model with two kinds of entities (VegetationUnit and Forager) is designed into Cormas.

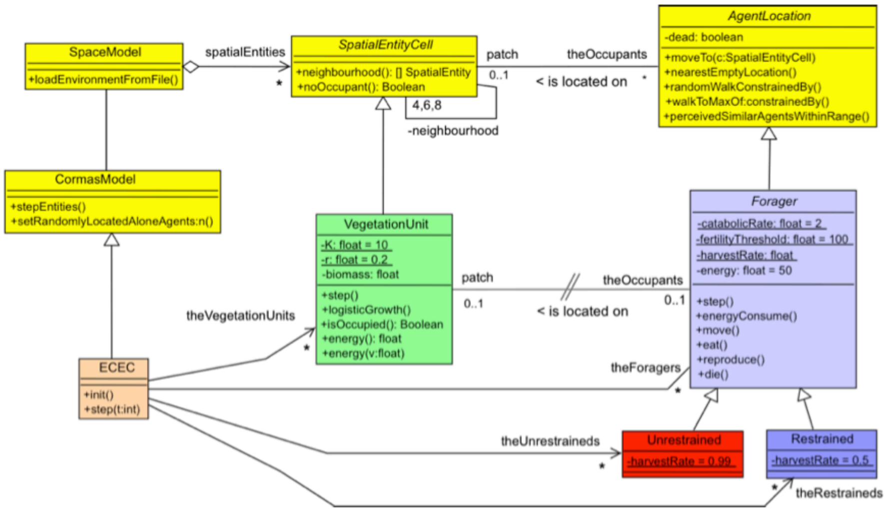

Fig. 1: UML class diagram of the ECEC model, adapted to Cormas (implementation stage). The yellow classes belong to the framework and the other (colored) classes are for the ECEC model.

One tutorial is based on this simple model called “ECEC”. It is a model by [9], that was replicated into Cormas. As it specializes AgentLocation, the Forager class inherits from useful methods predefined into its super-class such as #nearestEmptyLocation or #moveTo: they allow any instance of Forager to perceive the free places around him and to move to a target place. These generic methods can then be reused in the #step method to specify the global behavior of the agent. CormasModel is the abstract scheduler of the model and is in charge of the overall control of the dynamics. Here, it is specialized by the ECEC class that can reuse many predefined methods for instantiating the initial state of a simulation and for activating the entities. The scheduler contains three attributes, each one pointing on a list of the instances of the three concrete classes of the model (the green, red and purple classes). In Cormas, these attributes are set automatically when creating the classes of the model and the lists of instances are updated during a simulation (by removing the dead agents and adding the new ones).

The advantage of using a platform is also that it frees the modeler from many coding constraints. As Cormas complies with the MVC architecture[4], it allows the modeler to focus solely on his subject without worrying about the accessories that come with a simulator. After having coding the agents and the other entities of his model, the modeler must simply specify the way the entities are activated by the scheduler. Finally, he can specify the way he wants to visualize the entities and the probes of his model. For this phase, several interfaces are available that prevent the modeler to code the model display and the curves of the probes (see fig. 2).

When a model implementation is done, some simulations can be run. A spatial grid can be open showing the virtual landscape and the agents. Various ways of displaying them can be selected by the user in order to see the virtual word from different "Points of View".

Finally, the modeler can run analyses by setting three types of sensitivity analyses: simple stochastic analysis that repeats several simulations, OAT analysis (One factor At a Time) to study of the signature of the parameters (the value of a parameter is gradually or randomly changed for each simulation), and crossed analysis for which several parameters' values are changed simultaneously. The data of these analyses (recovered as time series, average over a simulation, min or max on a period) are saved in CSV or Excel format.

[4] MVC for Model-View-Controller is a model for software architecture that specifies a clear separation between the code of a model and how to visualize and manipulate it. This architectural model was designed in 1979 by Trygve Reenskaug [10], who was working on the design of Smalltalk with Alan Kay, Dan Ingals, Ted Kaehler, Adele Goldberg at the Palo Alto Research Center of Xerox.

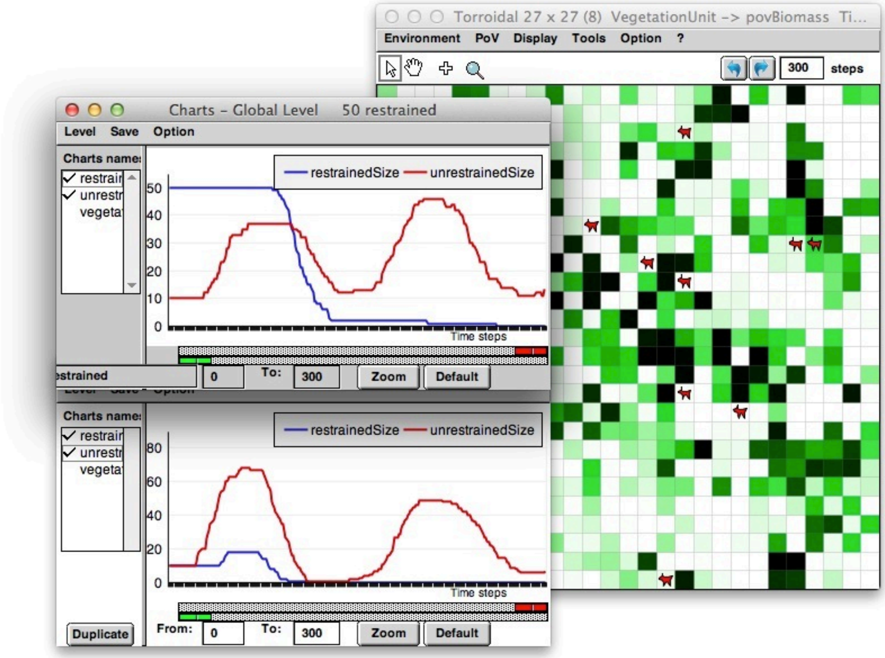

Fig. 2: The spatial grid displaying the biomass viewpoint of the vegetation (right part) and two chart interfaces showing the evolution of the populations of foragers: the upper chart presents a simulation with 50 initial restrained foragers when the lower one starts with 10 agents.

3 Participatory modeling to support the ComMod approach

From recent years, the development of *Cormas* is more focused on participation. This means that *Cormas* has taken an innovative direction towards the collective design and the interactive simulation of ABM. This new orientation has been taken to meet the increasing demand of our community of practice.

In parallel with the development of *Cormas*, the *ComMod* approach [5], [11]–[13] has been set up and applied in many countries. “Companion” means that the aim is to help the stakeholders in defining their own long-term objectives, to “accompany” them, instead of proposing a “turnkey” formula for renewable resource management [14]. In this complex field, it seems necessary to take some distance with the positivist posture that designates the scientific knowledge as the only accurate one. In contrast to this positivist posture and following in the footsteps of the Constructivist epistemology (in the sense of [15] and [16], the ComMod approach seeks to collectively “construct” knowledge on the basis of the stakeholders' perceptions and their social experience. Beyond traditional decision support systems, this participatory approach is based on the construction of a shared conception of how the participants perceive the present situation and how it could or should evolve. As a mediation approach, ComMod presupposes that the stakeholders must be well informed of the issues and that they all have an interest in solving the original problem.

To better understand the present situation of a given system, the collective design of an ABM is carried out in order to seek a mutual recognition of everyone's representation. In such a context, the modeling process is more a communication platform to facilitate collective learning than a predefined itinerary for piloting renewable resources [17]. To facilitate this difficult phase, the use of role-playing game (RPG) is often proposed in which the participants play their own role in a virtual and simplified situation. Even if it isn't in the conventional sense of the word, a RPG can be already seen as a representation of the world, i.e. a *model* [18]. The discussion on the game during the debriefing helps to confirm or revise some parts of this conceptual model.

RPGs involving local stakeholders enable representing context-specific situations of given social-ecological systems. A participant to a gaming session playing the role representing its main activity in real life provides information to specify the behavior of the corresponding computer agent [19], [20]. This approach is now well recognized in empirically-based ABM [21]: behaviors exhibited during the gaming session and the decisions made by the players are used to specify rules-based methods for the computerized agents [22]. Yet, the formulation of generalizable decision-making algorithms may be difficult for participants who tend to focus on their peculiar situation: as an individual, it may be difficult to think in terms of behaviors representative of a group of individuals.

In a second phase and from the conceptual model, the implementation of the ABM offers the possibility to explore scenarios collectively. Because and contrary to what one might expect, the design of an ABM does not immediately give access to understanding of its behavior. Indeed, time in ABM plays an active and decisive role by activating the entities progressively. The sequence of activities and interactions can often produce surprising outcomes hardly predictable. Even if the elementary mechanisms are simple, we are not able to take into account many elements that influence each other at the same time [23]. Thus, the simulation helps to understand the functioning of the system and to assess if the evolution of the virtual system is coherent with the real one. By confronting the common perception of the system with its virtual evolution, the simulation helps to correct the conceptual model in order to be more compliant with the actual situation.

The underlying model for the simulations depends on the way the actors are represented. Two major types of representation can be distinguished: (i) virtual agents performing predefined activities in a computerized ABM, or (ii) human agents playing their role in a RPG (as previously said, ABM and RPG both are types of models that characterize a situation). Between these two extremes, a range of intermediate situations exists where some decisions are human and others are computer-specified. The term hybrid agent simulation model covers all these intermediary cases [6].

4 The new functionalities of Cormas, oriented towards participative design and simulation

Instead of watching a simulation without interfering with the process, an interactive simulation aims at evaluating different decisions taken by the agents. For that purpose, the participants can change the parameters or can also send specific orders or even modify the main strategy of an agent. By interacting with the virtual system through "avatar" agents, the participants can test alternative strategies or new practices to assess their consequences.

The works recently undertaken on Cormas are consistent with this participatory approach oriented towards collaborative prospective. Some new tools are now available that allow the participants to actively contribute in the design of an ABM and to interact with the simulator. The users can define indicators that meet their requirements and choose to observe the simulation through specific filters (called "points of view" in Cormas). Because they are often spatially distributed, it is also possible to see just a part of the space (usually the one that concerns the participant). The users can also interact directly with their avatar (supposed to represent them) by moving them or sending instructions to use for example the water resources or to change the land cover. It is also possible to provide a set of core activities from which the user can shape a new strategy. Thus, with these new tools, the modeler no longer describes the overall behavior of the agents but provides basic activities that the users organize to interact with the environment and the other agents. Because these interactive simulations can also be distributed on networked computers, multiple users can interact on the same virtual environment. The objective is not to have distributed simulations on the Internet, but to interact within the same room: this proximity between the users facilitates direct interaction and non-verbal communication. These tools for interactive modeling are based on concrete experiences with ComMod and some new works are still in progress to cater for the growing needs of recent study cases.

4.1 Designing a model

As Cormas is a framework, the modeler has to specialize some predefined classes, mainly "social", "spatial" or "passive" entities. When designing a specific class, its attributes must be specified. Cormas assists the modeler in setting the initial value of these attributes. The following capture presents the initial value of the 'energy' attribute of the Forager class (see Fig. 1). It is set to 50 energy points by default. It means that at the initialization of a simulation, all the foragers start with 50 energy points. And during the simulation run, each new instance of Forager will have also 50 points.

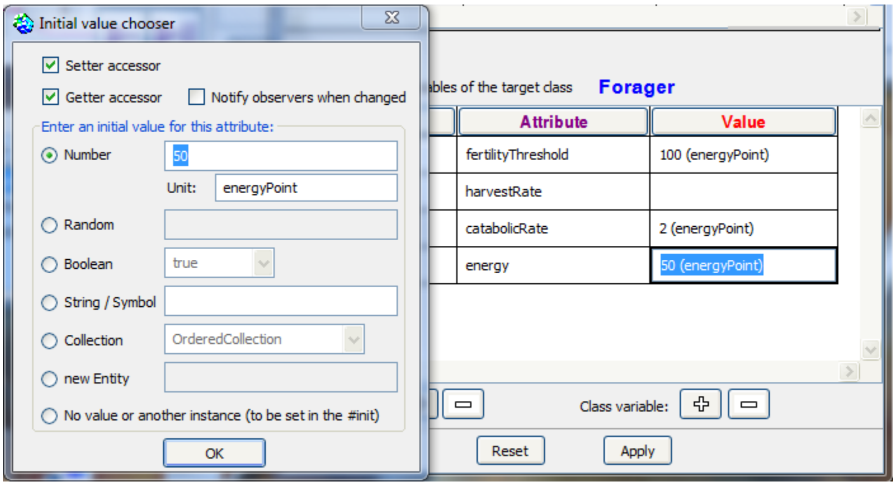

Fig. 3. Interfaces to set attributes and their default values

The default value can be easily changed. To test the effect of a new value, a table presenting all the numerical parameters allows the user to modify them temporary. For example, it is easy to change the initial number of restrained foragers as shown in the following capture.

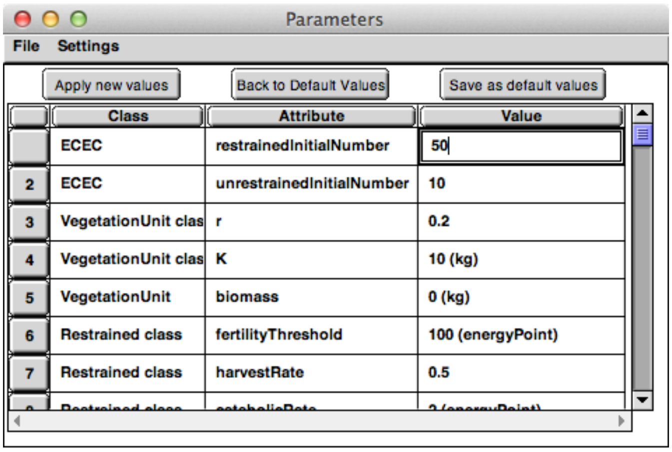

Fig. 4. Table of the numeric attributes

Thus to compare the two distinct simulations, a copy of the previous curves can be displayed (see Fig. 2). At the initialization and also during the progress of a simulation, the scheduler automatically registers all the new instances and removes the destroyed ones.

4.2 Multi windowing for displaying the agents

As Cormas complies with the MVC architecture, it is possible to visualize the spatial environment through several windows. Because the model is independent from the way to see it, one can select various points of view (PoV) to display (or not) the entities. In the PoV menu of a window that displays the spatial grid, some specific PoVs are available for each class of the model. By default, three PoVs are proposed: “nil” that doesn’t display the instances of the class, “defaultPoV” for which a standard figure (or color) is available and “povId” that displays each entity with a different color. But it is easy to draw specific PoVs thanks to the PoV setter interface. In the following interface, *povState* has been drawn to display the animals in good or weak condition (energy < 20). Thus by selecting *povState*, the figure of each forager is displayed according to its current condition.

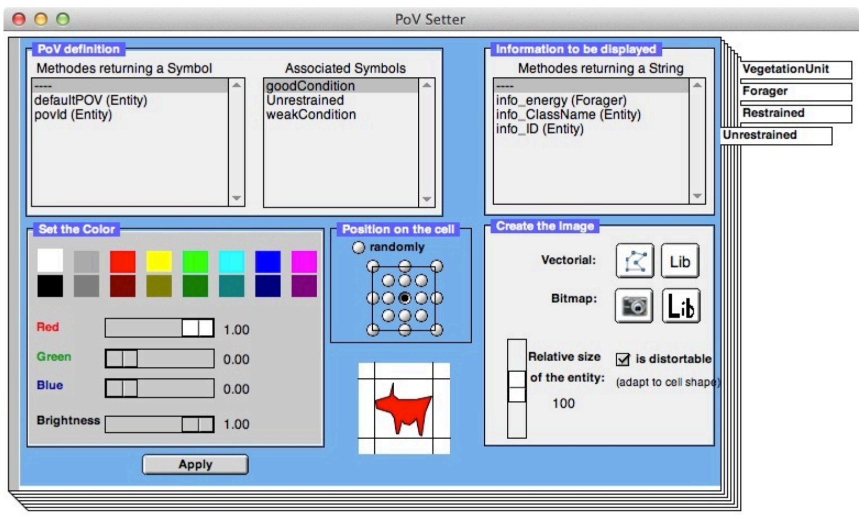

Fig. 5. The PoV setter presents the *povState* of an Unrestrained Forager in “goodCondition”

It is also possible to open another spatial grid and to select different PoVs in order to see two different viewpoints simultaneously. The zoom tool will open a third window displaying just a part of the grid (selected by the user). Some information can be displayed (example: the biomass value of each cell) and by using a contextual menu on an agent, this one can be tracked. The following screenshot shows a view on the full spatial grid (left part) and two zooms on the same region visualized with different viewpoints (right part).

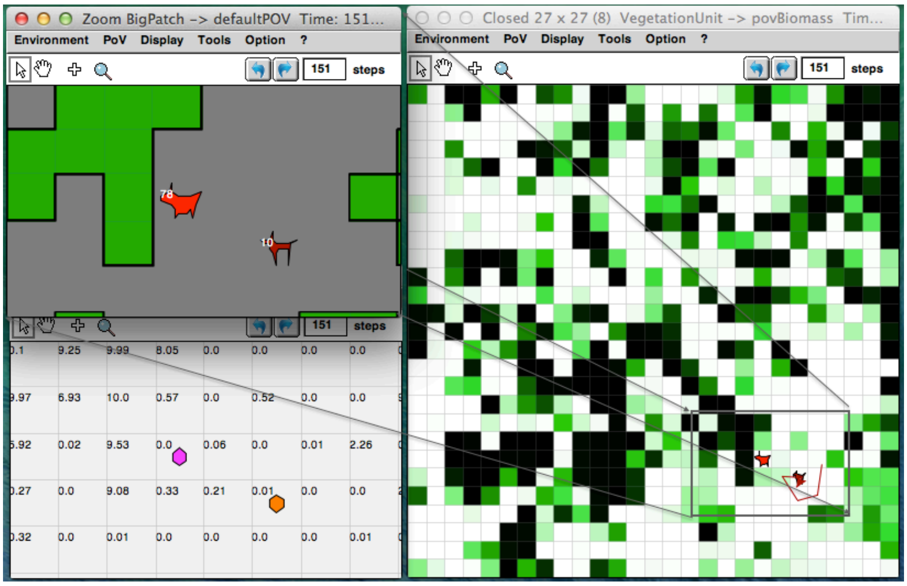

Fig. 6: A screenshot of the full spatial grid (left part) and two zooms on the same region visualized with different viewpoints (right part). The forager agents are displayed with *povState* or *podId*; the weakest is tracked on left part and their energy level is revealed on the upper right grid.

4.3 Manipulating the agents

To interact with a simulation, it is possible to modify the parameters' value of the entities (as shown in Fig. 4). But it is possible to act directly on the space and on the agents during the simulation. There are mainly two ways of acting: either on all entities simultaneously, or on some specific ones. In the first case, we can change the state of a group of agents or create new ones.

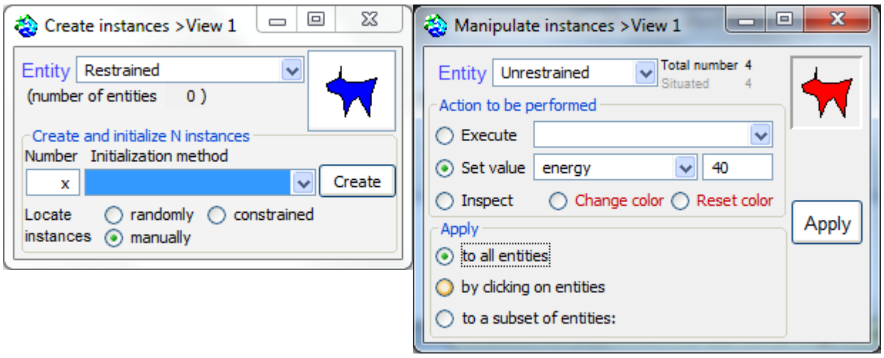

Fig. 7. Manipulation interfaces, one to create new instances (right) and the other to execute action or to change attributes' value of the agents

In the second case, the "Manipulation" tool allows the user to control an agent individually by moving him on another place or by sending to him some messages. Indeed, right-clicking on an agent opens a contextual menu that offers the possibility to select a message in an automatic scrolling list containing all the available methods of the class and subclasses of this agent. The following screen capture shows two ways to send messages to an agent: (left) preselecting the message that will be executed by each agent "Unrestrained" clicked by the user, or (right) clicking on an agent then selecting a method from the list to be performed.

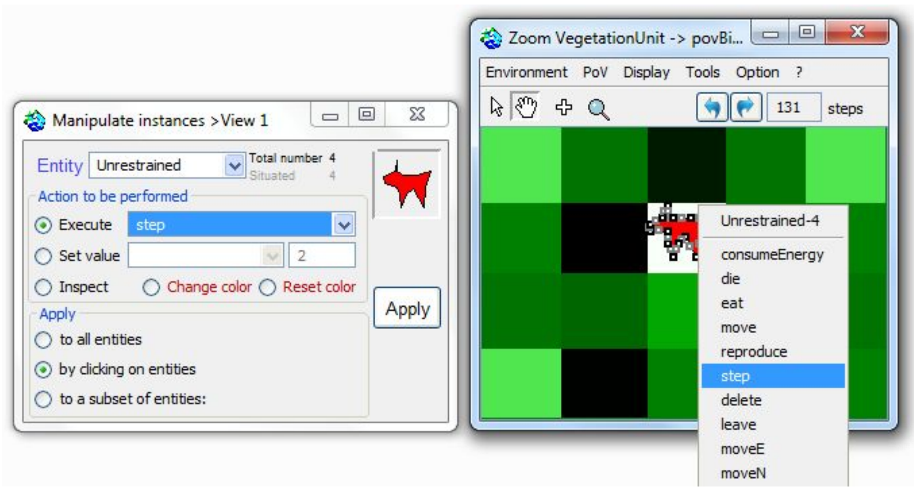

Fig. 8. Two ways to send the "step" message to an agent in Cormas: (left) preselecting the "step" message that will be executed by each clicked agent, or (right) clicking on an agent then selecting the "step" method from the list to be performed.

4.4 "Habitus": customizing the interfaces

Lately Cormas has been enhanced with a feature that enables customizing the spatial interface that fits specific ways to perceive and interact with the entities represented in the model. This feature is based on the concept of Habitus defined by Bourdieu as the set of ways of being, feeling, acting and thinking that are proper to an individual [24]. This concept is especially interesting because following Bourdieu, a Habitus structures the behavior and the actions of the individual while also structuring their position in a multidimensional social space. Hence, when developing a RPG, the Habitus feature of Cormas will guide the modeler in structuring the position of the different roles in the space of the socio-ecological interactions of the system. Defining a Habitus in Cormas, consists in defining: (1) how users can see the space interface: what entities are displayed and in which way, what information is available (textual information or tracking agents), and (2) how they can interact with this interface (what kind of entities can be created and what type of actions the users can ask an entity to be executed: to move, to consume, to slash-burn, etc.). The Habitus configuration interface is divided in two main parts: the observation of the entities and the manipulation of the entities. With the first part (top of the interface), the modeler can specify for each entity what PoVs can be accessed by the user, as well as the default PoV that must be displayed when opening the space interface for the first time. The same applies for the textual information to be displayed for each type of entity, as well as for the attributes and the probes that can be inspected. Playing with these configuration settings allow the modeler to develop very different ways to visualize and access the information about the simulated socio-ecological system.

The second part of the Habitus configuration interface (bottom part) allows the modeler to customize the list of operations to control an agent. If by default, all the operations are available, the list of control will most often be different from one Habitus to another, restricting the possibilities for each user in an asymmetric manner (see section 5.3). Hence for instance, we can develop two configuration settings: one for a user to control the grazing of a forager and a second for another user to control its reproduction. In the same idea, the modeler may customize the type of entities that can be moved or created, directly through the space interface during a simulation run.

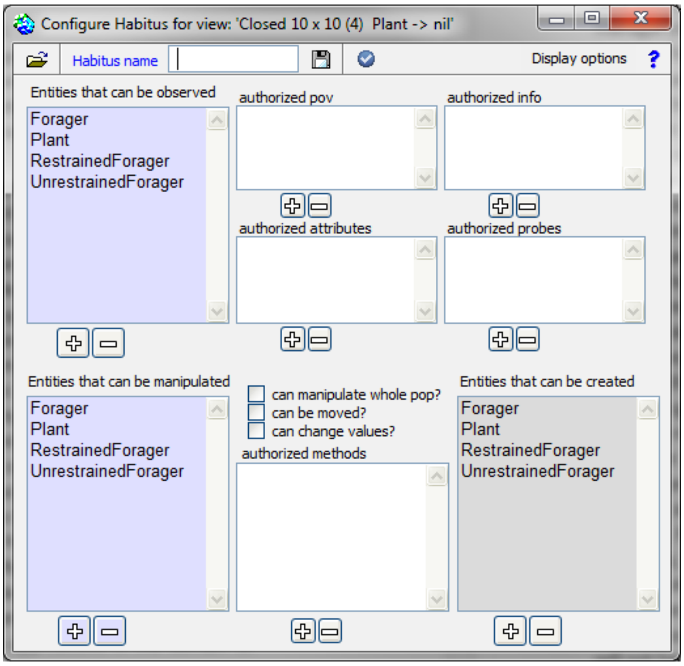

Fig. 9. The Habitus interface to restrict the option to visualize the entities (top part) and the control on them (bottom part)

4.5 Distributing the views for interactive simulation

In Cormas, a simulation can be distributed on several machines. This means that several users can remotely monitor the same simulation (with different viewpoints) and can remotely manipulate entities.

The distribution architecture in Cormas does not comply with the IEEE standard for common wargames across worldwide host computers. As this standard is much dedicated to real-time distributed applications by the commercial and military gaming industry, our goal is not to resolve complex dead-reckoning processes to enable massive on line simulation[5].

[5] In order to save bandwidth and to avoid the lag effects, dead-reckoning (for deduced reckoning) estimates the current position of an entity by using the previously detected position and by calculating the new position based upon known or estimated speeds over elapsed time and course.

As Cormas is based on the MVC architecture, the distribution is not completely duplicated on each computer, but only the views and the controllers. In that case, only one computer runs the simulation (the server) and the other connected computers (the clients) display some specific points of view on the virtual environment and provide limited control on the simulation. The remote visualization enables multiple users to manipulate their agents and to act collectively on the same virtual environment.

Even if a simulation can be distributed on the Internet, we prefer to use this ability on networked computers within the same room. For us, physical proximity is important as it allows the users to interact directly by talking to each other or by non-verbal communication.

4.6 Executable Activity Diagrams

Cormas offers an editor that enables the drawing of simple activity diagrams. During a simulation, these diagrams are executed directly by the agents, without any need for translation into code. These diagrams are interpreted "on the fly" by Cormas. Thus, it is possible to modify the behavior schema of an agent without coding it. It is also possible to modify the simulator while it is running, without stopping or restarting the simulation.

For simplicity sake, the elements available on the editor are restricted to initial and final nodes, decision points, simple activity nodes (without parameters nor ability to handle an activity output) and transitions. A decision point authorizes only two transitions to come out of it, indicating the fulfillment (true) or the negative answer (false) of a decision test.

By selecting an activity node or a decision point on the tool bar, the user can add a new element on the diagram. Then, he must choose the operation to be performed by this element by selecting it from an activity chooser. This one displays a list of operations belonging to the target class and its super-classes. Therefore, the user can draw a transition from the given node to another. Two transitions start from a decision point: one for which the answer of the decision test is true (green) and one for false (red). Thus, from basic operations already defined by the modeler, anyone may generate new upper level behavior without any programming skills.

This editor does not avoid the modeler to program his ABM. Its objective is rather to collectively design the behavior of an agent by organizing plug-and-play activity nodes. These activities contain pieces of code (software bricks or components) that were previously coded by the developers.

Because it is intended for non-specialists, the editor has been designed to be as simple as possible. For that reason it does not contain sophisticated features such as swimlane, iteration and concurrency notations that are nowadays specified by UML 2.0 [25]. In return, this simplicity enables anyone to participate more actively in the modeling design with greater efficiency thanks to the immediate assessment of any changes.

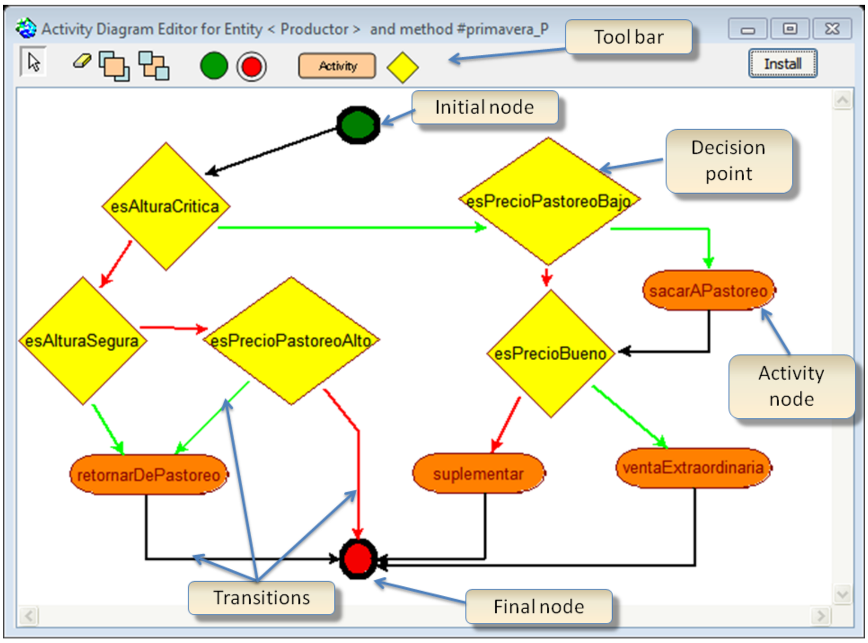

Fig. 10. The executable activity diagram editor

The executable editor operates differently than the standard "Executable UML" (xUML) [26], [27], which specifications require the translation of a diagram into code by executable UML compilers. Conversely in Cormas, an activity diagram is not compiled into code but is directly interpreted by the agents. In other words, a new activity diagram is saved as part of the source code of an ABM. It can be reopened at any time, modified and performed without compilation. By taking advantage of the Smalltalk facilities (a reflective programming language, dynamically typed), it is possible to modify the diagram of an agent while the simulation is running. As soon as the modified diagram is saved, the agent begins to perform his new behavior. This specificity can be useful when a user who is observing a trend of a simulation, wants to test how a change of the agent behavior could modify the direction towards where the simulation is going.

4.7 Time Travel Simulation: Stepping forwards and backwards in time

If the simulation dynamic is commonly done by time-step forward in time, Cormas enables also to navigate backward in time. As reverse-time calculation is mathematically unfeasible, Cormas does not simulate in reverse when stepping back. Therefore, to enable the step back capability, a previous forward simulation must be run to save snapshots. Thus, a click on the back button is merely a means of returning to the previously stored state. Thus one can go forward or backward to a particular instant in simulation time by restoring a recorded state.

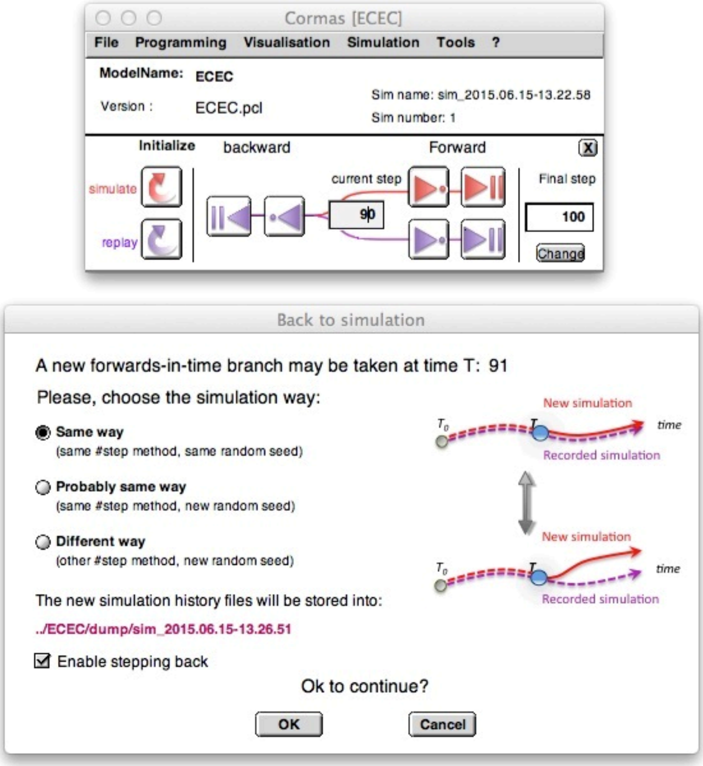

Fig. 11. Top: the main interface of Cormas with the 'Simulate' (red) and 'Replay' (purple) buttons. Bottom: the time bifurcation interface to start again simulating from a recorded state.

The step forward and back facility helps to analyze model and to verify if its mechanisms behave as expected. When trying to understand a strange behavior of a model, the user can return back to a specific moment just before the unclear period and, as for a movie, restart from that state to follow slowly how and why the entities act in such a way. But from that particular state, it is also possible to run a new simulation step by step to check if the entities behave similarly or if the system evolves in another way (called time bifurcation, see fig. 11).

Because a standard instantiation of a simulation may create artifacts (all agents with same age for example), it is practical to run the system for a moment until a balanced state. Then this current state can be saved to provide the starting point for future simulations.

Finally, the snapshot and restore ability is used when manipulating the agents: undo and redo buttons are available on the spatial grid to cancel a user action or to reactivate it.

5 Putting into practice

5.1 Interactive simulations to codesign with villagers an ABM on bushmeat hunting in the periphery of Korup National Park (Cameroon)

An ongoing project in the periphery of the Korup National Park (Cameroon) aims at helping the local population in managing the wild fauna, especially the overexploited animals hunted for their meat for personal consumption and for money. For that purpose, an early and interactive use of a stylized scale model was achieved with hunters in villages at the periphery of the park [28].

One of the challenges to design the model was the formulation of decision-making algorithms for the hunting activity. Two sets of workshops were organized in villages of the study area to introduce the spatially explicit individual-based module of the main hunted specie (the blue duiker, *Cephalophus monticola*, a small forest antelope) and then to elicit and specify the hunting practices of participants through collective discussions during the presentation of the computer simulation model. A 3-step exercise was carried out in order to facilitate the comprehension of the computer model among participants. The first step was meant to introduce the abstract representation of a village in the forest and the blue duiker individual-based population module. The different types of land cover and the notion of cell as a 1-ha portion of space were presented, as well as the 7 colors used to represent the various stages of individual blue duikers. In a second step, hunting with traps was introduced in a wider portion of forest (two villages linked by a road, a stylized map still without any realism). The last step was built on the elements previously introduced but was based on an explicit representation of the 7 villages and the Northern periphery of the Korup National Park. The whole portion of space represented in the model was gradually expanded: 1.5 km

* 1.5 km in the first step; 5 km * 5 km in the second step; 16 km * 18 km in the last step. This process of zooming out allowed starting focusing on the biology and the behavior of the blue duiker. The objective was to communicate and to discuss the related parameters and the underlying assumptions for the participants to not consider the model as a black box and to become familiar with it. In the final step of the workshops, the more realistic representation of the region in the model allowed making the final discussions more concrete. During the first stages of the workshops, the functionalities of the Cormas platform to interactively modify the attributes of the spatial entities (forest portions) and to directly create and manipulate located entities (animals and traps set by hunters) were used to display easy-to-follow configurations open to collective discussions (see fig. 12).

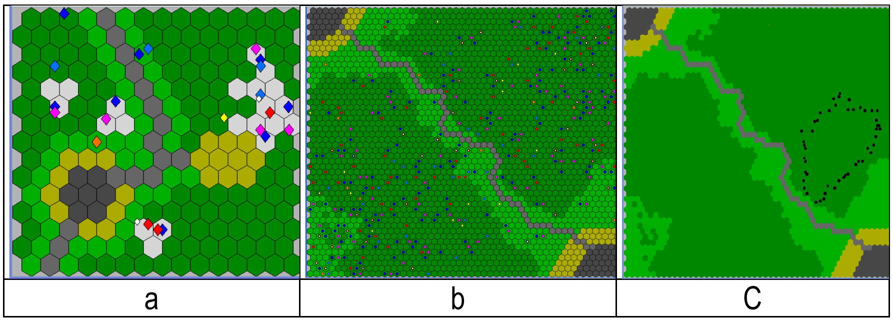

Fig. 12. (a): in a schematic representation of a village (dark grey) crossed by a road (light grey) and surrounded by agricultural fields (brown) in a forest (secondary in light green, primary in dark green), the various stages of antelope agents are displayed : adult (gravid females in pink, females in red, males in dark blue), subadult (males in light blue, females in orange), juvenile (in yellow) and newborn (in white). When a couple of adults have mated, they establish a 3-ha territory (3 very light grey cells) and exclude other adult antelopes to settle and reproduce there. (b): the spatial representation is zoomed out to display two villages connected by a road. In the forest; a population of antelope agents is created with a local density proportional to the distance to the nearest village. (c): results of a trap-path set interactively by a participant.

5.2 The collective design of an ABM by using Executable UML with livestock producers in Uruguay

The livestock sector plays a central role in the economy of Uruguay, which has the world's highest number of cattle per capita (3.8). Because of severe droughts that affected the north Uruguayan region in the last decades, we initiated a project to improve the adaptation capacities of livestock farmers. Indeed, in the late 1990s, livestock breeders experienced severe droughts provoked by climate change: millions of animals died or had to be slaughtered prematurely causing numerous bankruptcies. If certain farmers were less affected by these extreme situations, it was unclear how they worked exactly and which strategy was better in the long run.

To test different breeder strategies and to facilitate the communication among farmers and support services, we have built an ABM of livestock producers. The first step is more standard since it consists in implementing an ABM with pasture growth, herd dynamics and simple agents roughly imitating farmers' strategies [29]. This first version was presented and discussed with livestock breeders during several workshops. The main criticism concerned the over simplistic behaviors of the agent that makes his decision by looking solely at the pasture height or at the cattle health. It was therefore requested to revise the strategies of the agents.

The second step was more participative since it consists in modifying and assessing the model with cattle farmers. In order to make this assessment more lively and efficient, we conceived the xUML tool presented in the section 4.6. From a set of basic operations already available, anyone is able to generate new upper level behavior without any programming skills.

The use of the xUML editor revealed two interesting features. Firstly, by being able to modify the agents' behavior, anybody could play with the model and therefore better understand its logic. The immediate response obtained after any modification often acts as a stimulus for participants and increases their awareness of its underlying mechanisms. This leads to new questions about how the model operates, but also this has triggered discussions and debates about on how best to address climate crises. In conclusion, although the agent's strategies proposed by the first version of the model had often seemed too simplistic initially, many farmers afterwards categorized themselves as traditional producer like the one represented by the model.

The second feature concerns the collective debugging of technical aspects of the model. By testing alternative strategies with the xUML editor, the participants identified some biases: they realized that in drought conditions, the agents always reacted too late. For instance, in case of lack of grass, the decision to feed the herd with supplement did not apparently prevent it from collapsing. The participants understood that during crises, the agents had to act more frequently than only once per season as stated by the first model version. The consequence was to correct the model by repeating the agent’s activities every week rather than just once per season.

The results of these collective exercises exceeded our expectations. Beyond discussions and debates they triggered, it has contributed to identify better adaptive strategies so that the resilience of livestock producers can be improved. Furthermore, many of the farmers and technicians who participated in the workshops are continuing the experience with the model. They use it to seek for more effective management strategies under normal and drought periods. Now, we hope it will facilitate the emergence of new and more efficient practices for farm management that can account for climate changes [30].

5.3 Distributed asymmetric simulation to raise awareness about the multiple viewpoints on interactions between biodiversity and peri-urban development projects

In France, urban sprawling is a serious issue in many municipalities as it has major consequences in terms of loss of agricultural and forest area with destruction and fragmentation of habitats for biodiversity. In 2010, the bill dealing with the "Grenelle 2" national commitment for environment, has reinforce environmental conservation measures in urban planning at the municipality and inter-municipalities scale. Yet at the local level, these measures face difficulties due to the division of tasks and responsibilities related to environmental management in urban planning. Local actors lack an overall view and do not always understand why and how the different stakeholders engage with biodiversity.

To respond to this issue, an ABM was developed that simulates the interactions between land use change and two emblematic species of French countryside: the Montagu's harrier, a red listed migratory bird of prey that plays a role in the regulation of pests and seed dispersal, and the domestic bees, pollinators that provide ecosystem services for sweet chestnut forest. Groundwater quality is also modeled as a direct consequence of the type of land use composing the landscape at a given moment. On the social side, the ABM models the actions of five main stakeholders: a mayor, a property developer, a forester, a farmer and an ecologist.

Each of them has a specific objective related to the evolution of land use and depends on the others and on ecological dynamics to fulfill it (for more details see [31]. Those stakeholders are actually represented by virtual avatars in the computer simulation and by roles played by human agents who will make the decisions. Thus, the model is actually an hybrid agent simulation model that mixes computerized processes (ecological dynamics, land market demand, popularity of elected representatives...) and human decisions (buy lands, choose agricultural practices and timber harvesting methods, build new constructions, deliver construction permits, promote eco-friendly practices). When running a RPG workshop with the model, the simulation is distributed on several machines, one for each role. On each machine, a specific point of view on the virtual environment is displayed and a limited number of actions on the virtual entities are available. This customization is done using the Habitus functionality of Cormas (section 4.4) and used in distributed mode (section 4.5).

Playing the RPG with such an asymmetric setting helps participants to immerse themselves in the role of those stakeholders and better understand their constraints, their perspective and why sometime they misunderstand the choices of others about biodiversity conservation.

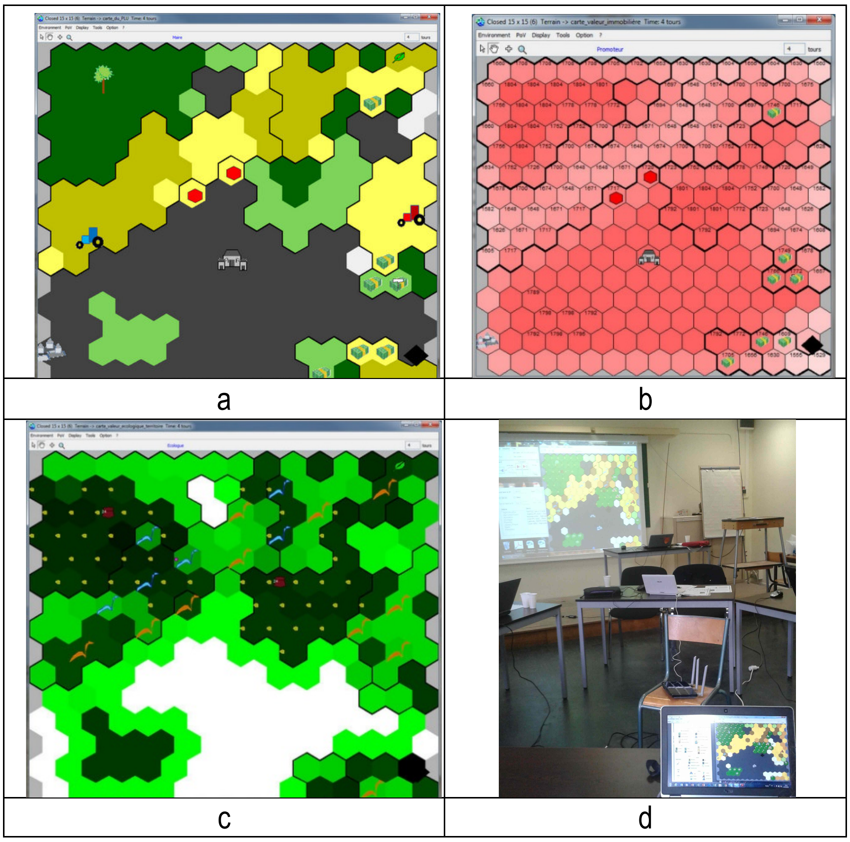

Fig. 12. (a): The Mayor's habitus displays the map of the local urban development plan that allows the player to act on land transactions that are symbolized by banknotes icons on the map. (b) The property developer habitus displays the map of land market prices with which the player interacts with land transaction. (c) The ecologist's habitus provides different information and means of action. It displays the location and health of Montagu's harriers and bees, which allows him to survey those populations and measure their ecosystem services. (d) The organization of the room

6 Perspectives and conclusions

Cormas is mainly dedicated for non-computer scientists and our objective is to help them in designing, implementing and assessing reliable and efficient simulation models. For that purpose, we try to keep the platform as simple as possible. This is the reason why Cormas does not propose continuous systems (neither temporally nor spatially). The DEVS formalism - Discrete Event System Specification [32] - for instance is much more sophisticated and enables to simulate realist movements and collisions. But in the domain of natural resources management, these refinements are not the first priority and seem not essential.

More rough concepts often seem sufficient to address the problems of this domain for which the question relies more on an overall understanding of the system rather than on the temporal precision of the interactions.

To help thematicians to design and implement their models by reducing their dependency on computer scientists for coding, the future developments of Cormas are focused on the design of user interfaces to automatically generate a part of the computer code. For that, we work on the integration of a class diagram editor that will generate the class structure and their attributes and will translate the associations into code. This tool is expected for the end of 2015.

We also develop a more complete activity diagram editor for manipulating variables. This will allow the modeler to define decision points by graphically describing the test with the model parameters and relational operators. Activities with input and output parameters will be available to define more complex operations than the current editor offers. However this tool will be useful only for modelers and may not be used with stakeholders.

We are also working on map integration that will enable for instance to load maps from Google Earth. A connection with R (a software for statistical computing) is almost ready that enables to run intricate sensitivity analyses from R and display professional statistical graphics.

But the main orientation of Cormas' future remains the interactivity with stakeholders and local actors. That is the reason why the major effort is dedicated on man-machine interfaces and ergonomic design. For instance, we are currently developing an extension for controlling the movement of agents on the spatial grid through tangible objects that are physically moved on a table. This work uses digital recognition of QR code printed on the top of tangible objects. By projecting the spatial grid on a table, this extension will be used in hybrid simulations mixing virtual environments and tangible objects. All these developments are taking place in parallel with concrete field experiences with a variety of local actors in order to deliver meaningful inputs on social and environmental issues.

Acknowledgments. **This paper was prepared with support from the "Ecosystem-based strategies and innovations in water governance networks for adaptation to climate change in Latin American Landscapes" (EcoAdapt) research program funded by the European Commission under FP7 contract ENV.2011.4.2.3-1/283163. The contents of this document are the sole responsibility of the authors and can under no circumstances be regarded as reflecting the position of the European Union. The authors assume the collective responsibility for the quality of the submitted and published work.**

References

[1] F. Bousquet, I. Bakam, H. Proton, et C. L. Page, « Cormas: Common-pool resources and multi-agent systems », in *Tasks and Methods in Applied Artificial Intelligence*, A. P. del Pobil, J. Mira, et M. Ali, Éd. Springer Berlin Heidelberg, 1998, p. 826‑837.

[2] C. Le Page, N. Becu, P. Bommel, et F. Bousquet, « Participatory Agent-Based Simulation for Renewable Resource Management: The Role of the Cormas Simulation Platform to Nurture a Community of Practice. », *Journal of Artificial Societies & Social Simulation*, v*vol. 15, no. 1. <http://jasss.soc.surrey.ac.uk/15/1/1.html>*, 2012.

[3] U. Wilensky et K. Reisman, « Thinking like a Wolf, a Sheep, or a Firefly: Learning Biology through Constructing and Testing Computational Theories-An Embodied Modeling Approach », *Cognition and Instruction*, vol. 24, nº 2, p. 171-209, janv. 2006.

[4] O. Barreteau et others, « Our Companion Modelling Approach », *Journal of Artificial Societies and Social Simulation vol. 6, no. 1. <http://jasss.soc.surrey.ac.uk/6/2/1.html>*, 2003.

[5] M. Étienne, Éd., *Companion Modelling. A Participatory Approach to Support Sustainable Development*. Dordrecht: Springer Netherlands, 2011.

[6] C. Le Page, G. Abrami, O. Barreteau, N. Becu, P. Bommel, A. Botta, A. Dray, C. Monteil, et V. Souchère, « Models for sharing representations », in *Companion Modelling*, Springer Netherlands, 2014, p. 69–101.

[7] A. Voinov et F. Bousquet, « Modelling with stakeholders », *Environmental Modelling & Software*, vol. 25, nº 11, p. 1268-1281, nov. 2010.

[8] J. Brauer, « The VisualWorks Development Environment », in *Programming Smalltalk – Object-Orientation from the Beginning*, Springer Fachmedien Wiesbaden, 2015, p. 77-96.

[9] Pepper J.W. et Smuts B.B., « The evolution of cooperation in an ecological context: an agent-based model », *Dynamics of human and primate societies: agent-based modeling of social and spatial processes*, Oxford, p. 45-76, 2000.

[10] T. Reenskaug, « The Model-View-Controller (MVC) Its Past and Present », *University of Oslo Draft*, 2003.

[11] F. Bousquet, O. Barreteau, C. Mullon, et J. Weber, « Modélisation d'accompagnement: systèmes multi-agents et gestion des ressources renouvelables », présenté à Quel environnement au XXIème siècle ? Environnement, maîtrise du long terme et démocratie, Abbaye de Fontevraud, France, 1996.

[12] [Collectif ComMod] F. Bousquet, M. Antona, S. Aubert, C. Barnaud, O. Barreteau, N. Becu, S. Boisseau, P. Bommel, A. Botta, J. Castella, et others, « La posture d'accompagnement des processus de prise de décision: les références et les questions transdisciplinaires », in *Modélisation de l'environnement : entre natures et sociétés.*, Quae. Hervé Dominique (ed.), Laloë Francis (ed.)., 2009, p. 71–89.

[13] T. R. Gurung, F. Bousquet, et G. Trébuil, « Companion modeling, conflict resolution, and institution building: sharing irrigation water in the Lingmuteychu Watershed, Bhutan », *Ecology and Society 11(2): 36*, p. 36, 2006.

[14] F. Bousquet, *Companion Modeling and Multi-agent Systems for Integrated Natural Resource Management in Asia*. Int. Rice Res. Inst., 2005.

[15] E. von Glasersfeld, « Le Moigne's defense of constructivism », *GRASCE, Entre systémique et complexité, chemin faisant [Between systemics and complexity – making the way]. Available at http://www.vonglasersfeld.com/225*, Paris, p. 85–90, 1999.

[16] J.-L. Le Moigne, *Les épistémologies constructivistes*, Presses Universitaires de France. 1995.

[17] C. Le Page, D. Bazile, N. Becu, P. Bommel, F. Bousquet, M. Etienne, R. Mathevet, V. Souchère, G. Trébuil, et J. Weber, « Agent-Based Modelling and Simulation Applied to Environmental Management », in *Simulating Social Complexity*, Springer Berlin Heidelberg, 2013, p. 499–540.

[18] W. Daré et O. Barreteau, « A role-playing game in irrigated system negotiation: between play and reality. », *Journal of Artificial Societies and Social Simulation*, vol. 6, nº 3, <http://jasss.soc.surrey.ac.uk/6/3/6.html>, 2003.

[19] O. Barreteau, « The joint use of role-playing games and models regarding negotiation processes: characterization of associations », *Journal of Artificial Societies and Social Simulation vol. 6, no. 2. <http://jasss.soc.surrey.ac.uk/6/2/3.html>*, 2003.

[20] F. Bousquet, O. Barreteau, P. d' Aquino, M. Etienne, S. Boissau, S. Aubert, C. Le Page, D. Babin, et J.-C. Castella, « Multi-agent systems and role games: collective learning processes for ecosystem management », *Complexity and Ecosystem Management. The Theory and Practice of Multi-Agent Systems, Edward Elgar, Londres*, p. 248–286, 2002.

[21] M. A. Janssen et E. Ostrom, « Empirically based, agent-based models », *Ecology and Society*, vol. 11, nº 2, p. 37, 2006.

[22] P. L. P. D'Aquino, « Using Self-Designed Role-Playing Games and a Multi-Agent System to Empower a Local Decision-Making Process for Land Use Management: The SelfCormas Experiment in Senegal », *Journal of Artificial Societies and Social Simulation* <http://jasss.soc.surrey.ac.uk/6/3/5.html>, 2003.

[23] G. Deffuant, G. Weisbuch, F. Amblard, et T. Faure, « Simple is beautiful? and necessary », *Journal of Artificial Societies and Social Simulation*, vol. 6, nº 1, <http://jasss.soc.surrey.ac.uk/6/1/6.html>, 2003.

[24] P. Bourdieu, *Le Sens pratique*, Éditions de Minuit. Paris, 1980.

[25] OMG, « The Unified Modeling Language Specification v2.0 ». 2005.

[26] S. J. Mellor et M. Balcer, *Executable UML: A Foundation for Model-Driven Architectures*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2002.

[27] OMG, « Model Driven Architecture, MDA Guide Version 1.0.1 », 2008.

[28] C. Le Page, K. S. Bobo, T. O. W. Kamgaing, B. F. Ngahane, et M. Waltert, « Interactive Simulations with a Stylized Scale Model to Codesign with Villagers an Agent-Based Model of Bushmeat Hunting in the Periphery of Korup National Park (Cameroon) », *Journal of Artificial Societies and Social Simulation*, vol. 18, nº 1, p. 8, 2015.

[29] F. J. Dieguez Cameroni, R. Terra, S. Tabarez, P. Bommel, J. Corral, D. Bartaburu, M. Pereira, E. Montes, E. Duarte, et H. Morales Grosskopf, « Virtual experiments using a participatory model to explore interactions between climatic variability and management decisions in extensive grazing systems in the basaltic region of Uruguay », *Agricultural Systems*, vol. 130, p. 89‑104, sept. 2014.

[30] P. Bommel, F. Dieguez, D. Bartaburu, E. Duarte, E. Montes, M. Pereira Machin, J. Corral, C. J. P. de Lucena, et H. Morales Grosskopf, « A Further Step Towards Participatory Modelling. Fostering Stakeholder Involvement in Designing Models by Using Executable UML », *Journal of Artificial Societies and Social Simulation*, vol. 17, nº 1, <http://jasss.soc.surrey.ac.uk/17/1/6.html> , 2014.

[31] N. Becu, N. Frascaria-Lacoste, et J. Latune, « Distributed Asymmetric Simulation - Enhancing Participatory Simulation Using the Concept of Habitus. », présenté à ISAGA 2014: The shift from teaching to learning: Individual, Collective and Organizational Learning through Gaming Simulation, Dornbirn, Austria, Bertelsmann, 2014.

[32] B. P. Zeigler, H. Praehofer, et T. G. Kim, *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press, 2000.