Exploratory Modeling of Complex Information Processing Systems

Jochen Kerdels
FernUniversität in Hagen
36 PUBLICATIONS **131** CITATIONS
SEE PROFILE

Gabriele Peters
FernUniversität in Hagen
88 PUBLICATIONS **565** CITATIONS
SEE PROFILE

Exploratory Modeling of Complex Information Processing Systems

Jochen Kerdels, Gabriele Peters
Chair of Human-Computer Interaction, University of Hagen, Universitätsstrasse 1 , D-58097 Hagen, Germany
{*Jochen.Kerdels, Gabriele.Peters*}*@Fernuni-Hagen.de*

Keywords: Exploratory Modeling, Modeling of Complex Systems, Emergence, Self Organization

Abstract: A widely adopted approach to study and understand complex systems such as ant colonies or economic systems consists in their modeling and simulation. In contrast to the predominat use of models and simulation in science as a substitute for the real system in order to predict the system's beaviour, the methodology of *exploratory modeling* uses modeling and simulation as a tool to increase knowledge and understanding of the systems themselves, for example to better understand the dynamic properties of a system. In this paper we propose a model which was specifically devised to support this process of exploratory modeling. The model defines four lightweight building blocks such as information processing entities that can be freely combined to model a particular complex system. Furthermore, the model provides an explicit state representation that comprises the entire model *including* an explicit representation of the information that is individually available to every information processing entity of the model. We illustrate our introduction of the proposed model by means of short examples of a concrete system for the simulation of motions in a flock of birds.

1 INTRODUCTION

A widely adopted approach to study and understand complex systems consists in their modeling and simulation. Typical examples of systems that are being investigated in that way include ant colonies, economic systems, nervous systems or biological evolution (Newman, 2011). Yet, much simpler systems based on only a small set of fixed rules like board games or cellular automata are also analyzed in this context (Evans, 2001; Holland, 1998; Gardner, 1970).

In contrast to the predominat use of models and simulation as a means to make valid predictions about the system that is being modeled, the analysis of the systems themselves typically uses the methodology of *exploratory modeling* (Bankes, 1993). In this context the simulation of a complex system serves as a means to test assumptions and to uncover unexpected implications of existing knowledge. Additionally the creation and execution of exploratory models act as a form of *intuition pump* (Dennett, 1998).

One important aspect of exploratory modeling is the ability to model and compare a wide variety of different systems to support the detection of common phenomena and their underlying conditions. Additionally, it should be possible to iterate over a series of variations of a model in order to explore those parameters of the model that are most uncertain.

2 OVERVIEW AND OBJECTIVES OF OUR APPROACH

In order to support the methodical approach addressed in the previous section we designed a general model to describe complex information processing systems. This approach makes the assumption, that every complex system can be represented by a corresponding complex information processing system, i.e., that the complex system can be reduced to a system that only consists of a set of entities that process and exchange information. The model we present here was designed particularly with regard to the following five objectives:

1. *Generality*. It should be possible to model a diverse set of complex systems such as cellular automata, neural networks, agent based systems or evolutionary processes.
2. *Explicit state representation*. The state of a particular system should have an explicit representation at any time step. Especially, this state should comprise an explicit representation of the entire information that is individually available to every information processing entity of that system.
3. *Focus on interactions*. The model should facilitate the description of complex interactions between information processing entities including aspects

of nontrivial addressing and delayed interactions.

4. *Expressiveness.* The perceived concepts of a complex system, i.e., its different entities, their possibly dynamic relations, and further characteristics of interest, should intuitively translate into corresponding elements of the model.
5. *Minimalism.* The interference between concepts and structure inherent to the model itself and those perceived of the complex system that is modeled should be as minimal as possible, i.e., the model should mitigate the phenomenon known as "Maslow's Hammer" (Maslow and Wirth, 1966).

The subsequent description of the model is structured as follows: Section 3 introduces the four major building blocks of the model and describes how theses building blocks relate to one another and how they can be combined. Section 4 explains how time is represented in the model, it defines what comprises a state and describes how state transitions are performed. Finally, section 5 reviews the presented model with respect to the objectives stated above.

To illustrate the description of the proposed model we will roughly sketch out the modeling of a simple *boid* simulation as an example along the way. Boids (bird-like objects) were introduced by (Reynolds, 1987) as a simple model of flocking behavior. In essence, the movement behavior of a boid is governed by three simple rules[1]:

1. *Separation.* Steer to avoid crowding local flockmates.
2. *Alignment.* Steer towards the average heading of local flockmates.
3. *Cohesion.* Steer towards the average position of local flockmates.

In addition to these basic rules the model can be extended with similar rules to avoid obstacles or to stay in a certain region. Despite this very simple set of rules a flock of boids can exhibit a surprising variety of motion patterns.

3 BUILDING BLOCKS OF OUR MODEL

The proposed model uses an object-oriented approach. It's main idea is to model a complex system as a combination of four basic building blocks (*proceties*[2], *messages*, *address filters*, and *procety attributes*) which will be described in detail in this section. First of all, it is important to note that these building blocks do not necessarily correspond to individual entities of a complex system such as single ants in an ant colony. (In terms of object-orientation the building blocks are *interfaces* not *base classes*.) Thus, individual entities of a complex system can be described by multiple building blocks at once, e.g., an entity of a complex system could be a *procety attribute* and a *procety* at the same time. In this regard, our model differs from most object-oriented modeling approaches for complex system, e.g., agent-based models[3].

The first building block of the proposed model are information processing entities, or *proceties* in short. A procety is any element of a complex system that is able to send and receive information, to create new proceties and remove existing ones. In case of the boid simulation, each boid can be interpreted as a procety: it has to receive information about the positions and headings of nearby flockmates and it has to send its own position and heading. Less obvious, possible obstacles in a boid simulation would also be modeled as proceties that actively send out information about their presence. As there is no common representation of an environment in our model, even "passive" elements have to be modeled explicitly as sources of information, i.e., proceties, if they are to be perceived by other entities of the system. At first glance this property of the model may seem to be a serious drawback, but contrariwise, this property helps to achieve the second objective stated in section 2.

The second building block are *messages*. They encapsulate the information that is exchanged between proceties. The information that is transferred by a message can be arbitrary. The recipients of a message are determined by its address. The address has the form of a set expression which allows to describe the recipients of a message on an abstract and conceptual level. It is a key component of the model in order to achieve the third objective stated in section 2.

The set expression used to specify the recipients of a message is composed of common set operators like union or intersection, and sets of proceties that are filtered by *address filters* – the third building block of our model. If an address filter is applied to a set of proceties the filter decides for every procety in the set if the procety should remain in the set or not. The information on which the address filter bases its decision is provided by a set of *procety attributes* that

[1] See also http://www.red3d.com/cwr/boids/ for a detailed description

[2] short form of *information processing entities*

[3] For an overview of agent-based models see (Salamon, 2011; Allan, 2010; Railsback et al., 2006).

are exhibited by the individual proceties. Procety attributes are the last building block of the proposed model. They provide a way to describe attributes that are shared among a set of proceties. In case of the boid simulation such a shared attribute could be the position and the heading of a boid. Based on this information, a corresponding address filter could then select all boids that are within a certain distance of the boid that originated the message – effectively restricting the flow of information from one boid to its local neighborhood only. In general, shared attributes are the basic prerequisite for the definition of global relations among proceties. These global relations can then be represented as address filters and as such be used in a set expression to specifiy the recipients of a message.

In addition to this addressing scheme, messages feature another important property. They can have an arbitrary long *time to live* (TTL) once they were sent by a procety. This property is the basis for a broad variety of time-dependent interactions between proceties. As a simple example, the messages sent by "passive" proceties like the aforementioned obstacles in the boid simulation can have an unlimited TTL and thus must be sent only once. More sophisticated uses are achieved in combination with appropriate, time-dependent address filters. For example, the propagation of a message over time can be described by an address filter that selects different sets of recipients depending on the age of the message, i.e., depending on how far the message has "travelled". In the boid simulation we could use this to model a message of type "boid cry" that expands radially over time from the boid that uttered the cry.

The four building blocks of the proposed model can be summarized as follows:

1. *Proceties* are "information processing entities" of a complex system. As such they process information which they send and receive in form of *messages*. Furthermore, proceties can create new proceties and delete existing ones.
2. *Messages* represents information that is transferred between proceties. The address of a message has the form of a set expression which allows to specify the recipients of a message on a high, conceptual level. Furthermore, messages have a TTL (time to live). That means, they can exist independently of any procety for an arbitrary number of time steps after they have been sent.
3. *Address filters* are used as a mechanism in the message addressing scheme of the model. They represent global relations among sets of proceties that share one or more procety attributes.
4. *Procety Attributes* represent common attributes that are shared among a set of proceties. They can be used by address filters to determine if the owner of an attribute should receive a particular message or not.

As stated at the beginning of this section, the described building blocks of our model do not necessarily have an exclusive one-to-one relationship to entities of the complex system that is being modeled. Instead, an element of a complex system can be represented by several building blocks at once. For example, the procety attribute of a procety could itself be a procety in its own right that modifies the attribute values according to the information it receives. A practical example could be the model of a biological neuron with its ion channels. In this case, the neuron as well as the ion channels could be modeled as proceties, where at the same time the ion channels would also be procety attributes of the neuron. In such a configuration, the ion channels could independently alter their behavior in reaction to ion channel specific signals, i.e., messages, that are directly processed by the ion channels themselfes and not by the neuron as a proxy.

4 TIME, STATES, AND STATE TRANSITIONS

The proposed model operates on a discrete time scale with steps $t \in \mathbb{N}$. At the beginning of each time step t the *state* of the model is given by the current states of all building blocks in the current model. This state comprises the states of all proceties, messages, address filters and procety attributes. The transition of the state at time t to the state at time $t+1$ is performed by the following three substeps:

1. Use the *procety scheduler* of the model to generate a processing schedule that governs which proceties are executed in substep 2.
2. Prompt each procety in the processing schedule to process the messages in their local message buffers.
3. Evaluate the addresses of all messages and distribute the messages according to the resulting procety sets to the local message buffers of each procety.

The use of a *procety scheduler* in the first step of the state transition allows to precisely control, how the proceties are updated during the state transition. The default procety scheduler of the model facilitates the synchronous updating of all proceties, i.e., every procety is prompted to process its messages in every time step. However, there are several models for complex systems that prefer asynchronous updating, e.g., many agent based models (Caron-Lormier et al., 2008). In these cases a custom procety scheduler can be defined to accurately emulate the updating procedure of the particular model.

The separation of the processing of messages in step 2 and the delivery of messages in step 3 effectively implements a double buffering scheme. This means that messages that were sent at time t will be processed earliest at time $t+1$. It also guarantees, that the order of the proceties inside the processing schedule has no effect on the behavior of the model.

5 REVIEW OF THE OBJECTIVES

The proposed model was specifically designed to support the process of exploratory modeling. In this regard we defined five objectives to guide the development of the model. In this section we review the model in relation to these objectives.

1. *Generality*. There are two main components of the model that provide the ability to model a wide variety of different complex system. First, the addressing scheme based on procety attributes and address filters facilitates the description of virtually any kind of static or dynamic relation between the elements of a system. Secondly, the ability to provide a custom procety scheduler allows to fully control if and when the elements of a system get updated.
2. *Explicit state representation*. As described in section 4 the state of a system comprises all building blocks of the model. In addition, the information that is individually available to a procety in that time step is represented by the messages inside the local message buffer of that procety. This state information is especially usefull to extract a communication graph for every time step that represents which proceties exchange information with one another. This graph representation opens up the possibility to use common graph measures to characterize the dynamic structure of the interactions occuring in the complex system. An extensive review on this topic can be found in (Newman, 2003).
3. *Focus on interactions*. As already stated above, the addressing scheme provides an effective way to describe a wide variety of interactions between the proceties. Furthermore, the ability of messages to exists for arbitrary long periods of time and the possibility to define time-dependent address filters yields a whole new area of interactions that can be modeled, e.g., the spatial propagation of signals. The modulation of messages was not explicitly addressed. However, as the building blocks of our model can be freely combined, a message can also be a procety or an address filter for instance. The latter, for example, would allow for the manipulation of the message content while it is propagated over time.
4. *Expressiveness*. The building blocks of the model were designed in the spirit of object-orientation to encourage the encapsulation of local knowledge. For example, when proceties interact by exchanging messages they use a set expression for addressing other proceties. This set expression contains only address filters which typically represent high level concepts about some relation between the proceties. Thus, a procety does not need to know much about the other proceties they are interacting with. The same holds true for the address filter itself. The filter just relies on the information provided by specific procety attributes.
5. *Minimalism*. The model uses only four building blocks which have a very precise and small "conceptual footprint". As the building blocks can be freely combined, there should be no need to "forcefully" fit a perceived concept of the complex system onto a single building block of the model.

6 SIMULATION EXAMPLE

We created a software library using C++ and the ROOT data analysis framework developed at CERN (Brun and Rademakers, 1996) to support the implementation of simulations that use our model. As the ROOT framework is designed to handle and analyse large amounts of data, it is well suited for the data intensive process of exploratory modeling.

As a first test case for our model we implemented a boid simulation as described in section 2. In addition to the three basic rules of a boid we added a rule that keeps the boids confined to a local area.

One of the parameters that influences the observed flocking behaviour very strongly is the viewing range of the individual boids. Is the viewing range to short, e.g. only 10 units, then no observable flocking occurs (s. Figure 1).

If the viewing range is increased to some medium value, e.g. 35 units, then the emergence of several independent flocks can be witnessed (s. Figure 2).

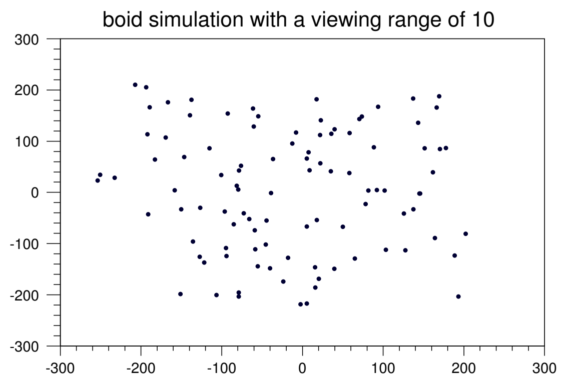

Figure 1: A representative distribution of boids with a short viewing range of 10 units.

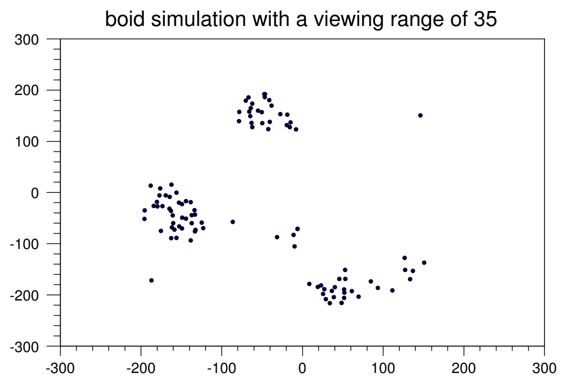

Figure 2: A representative distribution of boids with a medium viewing range of 35 units.

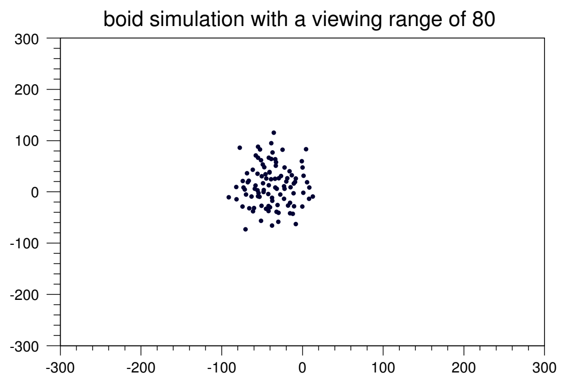

Figure 3: A representative distribution of boids with a long viewing range of 80 units.

However, if the viewing range is increased further, e.g. to a value of 80 units, then the dynamic behavior breaks down and all boids accumulate in a single cluster (s. Figure 3).

In order to analyse the effect of the viewing range on the emergence of flocking behaviour in this simulation, we ran a series of simulations in which we increased the viewing range stepwise from 5 units to 90 units. As a measure of "flocking" we used the average number of clusters that occured within 1000 time steps. Figure 4 shows the result of this experiment. With increasing viewing range the number of clusters drop rapidly starting with 100 clusters at a viewing range of 5 to 10 clusters at a viewing range of 35. Starting with a viewing range of 60 the simulation averages on one single cluster, i.e. no flocking occurs anymore.

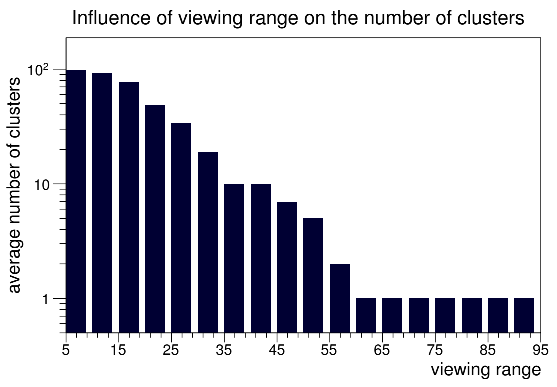

Figure 4: The viewing range of the individual boids influences the average number of clusters that emerge during a simulation run.

7 CONCLUSIONS

In this paper we presented a new model for the description and simulation of complex information processing systems. We designed this model specifically to support the *exploratory modeling* of complex information processing systems. Among its characteristics we would like to point out the following properties:

- The model provides a set of building blocks which can be freely combined to model the particular complex system. This contrasts existing models where the different model components are isolated from each other, for example NetLogo (Wilensky, 1999) or repast (North et al., 2006).
- The state of the model has an explicit representation. This facilitates the creation and application of measures that describe global, dynamic aspects of the system which is being analyzed.
- The model focuses on the interactions between its information processing entities. The building blocks of the model are designed such that the modeling of interactions is facilitated.
- The model allows messages to exist over arbitrary long time intervals which, e.g., enables modeling of the spatial propagation of a message.

REFERENCES

Allan, R. (2010). Survey of agent based modelling and simulation tools. Technical report, STFC Daresbury Laboratory, Computational Science and Engineering Department, Daresbury, Warrington WA4 4AD.

Bankes, S. (1993). Exploratory modeling for policy analysis. *Operations Research*, 41(3):435–449.

Brun, R. and Rademakers, F. (1996). Root - an object oriented data analysis framework. In *AIHENP'96 Workshop, Lausane*, volume 389, pages 81–86.

Caron-Lormier, G., Humphry, R., Bohan, D., Hawes, C., and Thorbek, P. (2008). Asynchronous and synchronous updating in individual-based models. *Ecological Modelling*, 212(3-4):522–527.

Dennett, D. (1998). *Brainchildren: Essays on Designing Minds, 1984-1996*. Representation and Mind Series. Mit Press.

Evans, K. M. (2001). Larger than life: Digital creatures in a family of two-dimensional cellular automata. In Cori, R., Mazoyer, J., Morvan, M., and Mosseri, R., editors, *Discrete Models: Combinatorics, Computation, and Geometry, DM-CCG 2001*, volume AA of *DMTCS Proceedings*, pages 177–192. Discrete Mathematics and Theoretical Computer Science.

Gardner, M. (1970). Mathematical games: The fantastic combinations of John Conway's new solitaire game 'Life'. *j-SCI-AMER*, 223(4):120–123.

Holland, J. (1998). *Emergence*. Oxford University Press, New York.

Maslow, A. and Wirth, A. (1966). *The psychology of science: a reconnaissance*, volume 8 of *The John Dewey Society lectureship series*. Harper & Row.

Newman, M. (2003). The structure and function of complex networks. *SIAM Review*, 45(2):167–256.

Newman, M. E. J. (2011). Resource letter cs-1: Complex systems. *American Journal of Physics*, 79:800–810.

North, M., Collier, N., and Vos, J. (2006). Experiences creating three implementations of the repast agent modeling toolkit. *ACM Trans. Model. Comput. Simul.*, 16(1):1–25.

Railsback, S., Lytinen, S., and Jackson, S. (2006). Agent-based simulation platforms: Review and development recommendations. *Simulation*, 82(9):609–623.

Reynolds, C. (1987). Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '87, pages 25–34, New York, NY, USA. ACM.

Salamon, T. (2011). *Design of Agent-Based Models : Developing Computer Simulations for a Better Understanding of Social Processes*. Academic series. Bruckner Publishing, Repin, Czech Republic.

Wilensky, U. (1999). *NetLogo http://ccl.northwestern.edu/netlogo/*. Center for Connected Learning and Computer-Based Modeling, Northwestern University., Evanston, IL.