Spezifikation der Simulation der Struktur und Dynamik von Pflanzenbeständen und Tierpopulationen mit sensitiven Wachstumsgrammatiken

Christian Hüning1, Jason Wilmans2, Nils Feyerabend3, Thomas Thiel-Clemen4

Hochschule für Angewandte Wissenschaften Hamburg

Department Informatik, Berliner Tor 7, 20099 D-Hamburg

1[christian.huening@haw-hamburg.de](mailto:christian.huening@haw-hamburg.de), 2[jason.wilmans@haw-hamburg.de](mailto:jason.wilmans@haw-hamburg.de)

3[nils.feyerabend@haw-hamburg.de](mailto:nils.feyerabend@haw-hamburg.de), 4[thomas.thiel-clemen@haw-hamburg.de](mailto:thomas.thiel-clemen@haw-hamburg.de)

www.mars-group.org

Abstract

The usage of Individual Based Modelling (IBM) or Agent Based Modelling (ABM) in ecological science is well accepted for 25 years now. However a lot of experience on when and how to use IBM has been collected over time (Filatova et al., 2013; Grimm, 1999; Huston et al., 1988) as well as new approaches, methods and technologies in computer science have emerged (Bellifemine et al., 2008, 2007; Grimm et al., 2006, 2010; Le et al., 2008; Ralha et al., 2013; Vigueras et al., 2013).

Together with these improvements, new obstacles and problems have arisen in the various domains of ecological science. Some of the major challenges are the integration of different models and almost arbitrary data into combined simulation models (Thiel-Clemen, 2013), execution performance of simulations and the need for large scale scenarios, while at the same time be able to visualize the simulation.

The coupling of ecological, social, economic and political systems creates a huge complexity to the overall model and simulation. Creation and usage of multi-agent based simulation systems has proven to be a great tool to explore and investigate such models (Le et al., 2008; Ralha et al., 2013).

The MARS Group (Multi Agent Research and Simulation) of the Hamburg University of Applied Sciences is developing a distributed and highly scalable framework for use in research and education. MARS is not a single program, but consists of a multitude of processes and tools chained together to provide an approach to most if not all of today’s simulation requirements. In our presentation the current state of development is demonstrated. Future research and development topics as well as concrete scenarios will also be shown.

# Introduction

The usage of Individual Based Modelling (IBM) or Agent Based Modelling (ABM) in ecological science is well accepted for 25 years now. During this time a lot of experience on when and how to use ABM has been collected (Filatova et al., 2013; Grimm, 1999; Huston et al., 1988), as well as new approaches, methods and technologies in computer science have emerged (Bellifemine et al., 2008, 2007; Grimm et al., 2006, 2010; Le et al., 2008; Ralha et al., 2013; Vigueras et al., 2013).

Together with these improvements, new obstacles and problems have arisen in the various domains of ecological science. Some of the major challenges are the integration of different models and almost arbitrary data into combined simulation models (Thiel-Clemen, 2013), execution performance of simulations meeting the need for large scale scenarios, and at the same time being able to visualize the simulation.

The coupling of ecological, social, economic and political systems creates a huge complexity to the overall model and simulation. Creation and usage of multi-agent based simulation systems has proven to be a great tool to explore and investigate such models (Le et al., 2008; Ralha et al., 2013).

The MARS Group (Multi Agent Research and Simulation) of the Hamburg University of Applied Sciences is developing a distributed and highly scalable framework for use in research and education. MARS is not a single program, but consists of a multitude of processes and tools chained together to provide an approach to most if not all of today's simulation requirements. ~~In this paper we present an overview of relating literature and results (Chapter 2), the current state of development of the MARS SYSTEM (Chapter 3) and first results from prototypic implementations (Chapter 4). The last chapter features a discussion as well as future research and development topics.~~

# Requirement for modern simulation

### Modularity / Reusability

As more and more ecological models were created and programmed over the years, more and more paradigms and ways of implementation of these models emerged. With that another interesting aspect of ABM came along, the integration of different models. The idea is to connect and integrate domain specific models from domain specific experts to create a new super model of a certain domain. If for example one would want to create a large scale model of the ecosystem of a national park in south Africa, it would be very helpful, if one could use existing models of elephants, cheetahs etc.. Actually doing that, turns out to be much more difficult, since every group of scientists working on a model uses another, individual paradigm, architecture, programming language, data format and so forth. Villa (2001) proposes his Integrating Modelling Architecture (IMA) for hat purpose. He singles out three characterizing dimensions for connecting different models:

* **Representation** A unified semantic relating to the depiction of space, time and behavior in every respective model is needed.
* **Domain** A clear distinction between the domain spaces of each sub-model must be made. In particular this relates to the input and output parameters which are valid for each sub-model.
* **Scale** Data, which is exchanged between models, must be compatible or translated in space and time dimensions.

~~A recent contribution to the Scale dimension has been made by Thiel-Clemen (2013), who proposes a data warehouse based information integration process on the simulation data.~~  
   
These dimensions target the difficulty when technically connecting different models. A more functional view has been made by Liu et al. (2007) who take a look at the complexity of coupled human and natural systems. Their integration efforts aim at taking interdisciplinary research on a broader scale into account, as well as exceeding local and temporal boundaries when modelling certain ecological system. As shown by their findings, almost every ecosystem today is tightly coupled with its neighboring economic or social systems and thus these need to be taken into account when watching the evolution of that ecosystem. Filatova et al. (2013) move even further and demand that the corresponding aspects of ecological systems like economy, social systems and bio-physical dynamics need to be integrated into the representation of a heterogeneous landscape representation.   
The discussion today circles around the fields of model re-usage (Holst (2013)), model integration (Filatova et al. (2013), Le et al. (2008), Liu et al. (2007), Villa (2001)), which makes distributed, parallel simulation execution (Cicirelli et al. (2010), Wang et al. (2009), Wang et al. (2012), Bellifemine et al. (2007), Thiel (2013), Vigueras et al. (2013)) necessary and the question of spatial-temporal information integration (Thiel-Clemen (2013), Filatova et al. (2013)) is raised.

### Scale

As we know from certain modelling work related to our research (Pereki et al., 2012), we need to be able to simulate large numbers of agents to produce realistic results. This holds especially true, when a model is used to predict / forecast the future development of tis real world counterpart.   
Since the above mentioned ideas produce a lot of computing complexity, the need for appropriate simulation tools and frameworks arises. Over the past years there have been quite a lot approaches to this field, which will be further examined in detail in the next chapter.

model's elements as simple as possible, still probably highly intense computation (collision avoidance, path finding, learning algorithms and so on)

imagine draughts or checkers, 10x10 fields = 100 fields. same resolution, doubled size, 20x20 = 400 fields. in 2 dimensions doubled size = 4 times computational effort, in 3 dimensions even 8 times

also: area of interest in really big (Krüger national park or more)

also: limited budget for research, cost efficiency required

### Ease of use

### Information Integration

### Focus on scalability

# Related work

A huge number of MAS frameworks and domain specific implementations have been created over the past years. Since I strive to create yet another framework, it makes perfect sense to look at the previous work and evaluate their capabilities and usefulness.

## Simulation Frameworks

### JADE

One of the most famous frameworks is JADE (Bellifemine et al., 2007) which allows to execute a simulation distributed across several JADE container processes or just locally in a single container. JADE was developed in Java to create a reference implementation of the FIPA agent specification ([http://www.fipa.org)](http://www.fipa.org/). The performance of JADE has been extensively investigated by Mengistu et al. (2008). Their findings show that JADE has significant performance issues in the fields of communication and agent migration due to the usage of the LDAP protocol and slow message transport services. JADE’s Lookup-Directory-Service also is measured to be slow, which is caused by not using local caching on the respective nodes. Mengistu et al. (2008) propose improvements to both mechanisms and present promising results from experiments they conducted. However a more recent investigation of JADE’s performance seems appropriate, given that the paper is almost 6 years old.

### GAMA

GAMA (Amouroux et al., 2007) is a modeling and simulation framework which is based on RepastJ. It features a nice model description language, called GAML, which allows nonprogrammers to create complex models. GAMA is written in JAVA and thus executable on all java enabled systems. A very strong feature of GAMA is its visualization feature, especially when it comes to using GIS data. An easy import function allows to quickly create a scenario’s environment and visualization from a GIS file and thus allows for a quick integration of that kind of data.

The downside of GAMA is, that it’s not possible to distribute the system and that it does not scale well across multiple CPU cores. In fact when testing GAMA, it actually used only just up to 4 cores while running on a 24 core machine. While testing I found GAMA to have a perfomance threshold around 80.000 agents, with one simulation step taking more than 800ms on the aforementioned machine.

### WALK

Also from 2013 comes a solution with a strong focus on evacuation scenarios which has been developed here at the Hamburg University of Applied Sciences and is called WALK (Thiel, 2013). It features a dynamic (re)partitioning and distribution of agents across several compute nodes and is thus capable of running simulations with hundreds of thousands agents on commodity hardware. In fact Thiel (2013) showed in his final tests that WALK can run a 300.000 agent random walk simulation in near real time. Also remarkable about WALK is, that its agents pass the RiMEA tests and thus provide a pretty good behavior. As a recent addition Stefan Münchow added support for leadership models and social behavior to the agents implemented in WALK. These additions show very promising results and create a very high interest in re-using the agent implementation from WALK in the new system whenever human agents are explored.

### Vigueras

Another interesting architecture (Vigueras et al., 2013) proposes an almost completely asynchronous, distributed simulation execution to implement interactive simulations, that may be visualized in near real-time. The only time Vigueras et al. (2013) synchronize the execution of their agents is, when they happen to act or move beyond the boundaries of their respective environment patch.

When it comes to visualization of the simulation Vigueras et al. (2013) utilize visualization nodes (VS) that also act asynchronously on the distributed nodes. Each VS has a camera-style definition of its field of view and may thus only ask those nodes for information containing parts of the environment, which is in that field of view. This is very contrary to other visualization approaches (e.g. GAMA, NetLogo), since it does not attempt to visualize the whole simulation at once.

Considering the amount of agents and the sheer sice of simulated space in our upcoming scenarios, this approach might become very valuable.

## Case Specific Implementations

### LUDAS

LUDAS (Land-Use Dynamic Simulator) (Le et al., 2008) implements a social-ecological, landuse/cover change (LUCC) model featuring four components, which implement human population including behavior, the environment, various policy factors with focus on land-use choices and lastly a decision making procedure which integrates the first three features. The model simulates "a watershed in Vietnam for integrated assessments of policy impacts on landscape and community dynamics". The implementation has been done in NetLogo and thus does not provide a very high performance, but showcases the scenario pretty nice.

It is not performance nor distribution which makes LUDAS interesting, but the great integration of LUCC components into a working simulation scenario. If that model can be translated into a larger, more capable software architecture, it could provide some very decent results in future, larger scale LUCC simulations.

### MASE

MASE (Ralha et al., 2013) is another LUCC simulation which targets the development of robust land-use strategies. The showcase features a region called Cerrado in Brazil. Whats remarkable about MASE is, that it utilizes a methodical, empirical parameterization process for human behavior, which has been developed by Smajgl et al. (2011). The implementation has been done with JADE (Bellifemine et al., 2007) and Matlab.

# Architecture proposal

## Agent Shadowing

Assuming an architecture where agents live in three dimensional layers each distributed across one or more container nodes, the problem of synchronization and communication arises, when it comes to agent interaction or movement across the boundaries.

Specifically whenever an agent wants to communicate (that is call a method) with another agent, it first has to check whether the desired agent is present locally or remotely and, if remotely, obtain a reference through which it may perform the actual communication.

If agents may move around their environment or are moved by a load balancing partitioning mechanism, it may well happen, that an agent crosses the virtual border of a container node’s part of the layer and thus has to be moved to another container node instance. If that happens the communication reference of that agent has to be updated, whenever another agent holding an old reference wants to communicate.

### Concept

Agent Shadowing is the depiction of an agent living on layer A1 drawn onto layer A2, where it is not actually instantiated, but instead is represented by a stub-like object as in remote communication concepts like RPC/RMI.

In RPC/RMI each agent’s methods are callable by third parties through its stub object. Usually a stub just provides the capabilities to establish an interface-bound communication with the remote object. If the remote reference changes, in classic RPC/RMI the stub simply becomes useless since its reference is not updated. The protocol then has to notice the broken link and re-establish a new one.

A shadow agent stub (SAS) is extended by the ability to hold cached attributes like its position or any other attribute. Both, the attributes and the remote reference, may be updated by the real agent object whenever a change occurs. These updates may be delivered via multicast when in LAN to reduce the amount of traffic. The initial remote references can be provided when the overall system is initiated since some kind of distribution information has to be provided at that state.

This results in each container node containing the full environment as well as all 10.000 agents, but with the difference, that only 5.000 agents are really instantiated (and thus have to be computed). The other 5.000 agents are only instantiated as SASs and thus do not contain any agent behavior logic. An increase in container nodes would reduce the amount of agents per node that have to be actively computed, while the memory footprint per node would also potentially decrease, assuming that a SAS consumes less RAM than a full fledged agent.

### Hypotheses:

1. This data-binding mechanism significantly reduces the amount of (duplicated) network communication / traffic between agents, because heavily used attributes may be cached in SAS.
2. Lookup of remote references is not necessary anymore, since each agent is virtually present at each container node and may be accessed through its usual interface, with the stub-object binding taking care of the remote reference[[1]](file:///E:/Master/Shadow%20Agents/ShadowAgents.docx#_ftn1).
3. Distribution of agents is transparent to the programmer.
4. No single-point-of-failure since no central directory for lookup or routing is necessary. Furthermore if a container node crashes, its state might be recreated by another node.
5. Massive traffic resulting from multiple simultaneous SAS updates, can be reduced by aggregating these updates into one large batch update.
6. The system is limited by the maximum amount of RAM per node .
7. This limitation can be compensated by introducing lazy loading of SASs, utilizing potential locality of agent interaction and a garbage collection for SASs which have been unused for too long[[2]](file:///E:/Master/Shadow%20Agents/ShadowAgents.docx#_ftn2).

# Outlook

**References**

Bellifemine, Fabio, Caire, Giovanni, Poggi, Agostino, & Rimassa, Giovanni. 2008. JADE: A software framework for developing multi-agent applications. Lessons learned. Information and Software Technology, 50(1-2), 10–21.

Bellifemine, FL, Caire, G, & Greenwood, D. 2007. Developing multi-agent systems with JADE.

Filatova, Tatiana, Verburg, Peter H., Parker, Dawn Cassandra, & Stannard, Carol Ann. 2013. Spatial agent-based models for socio-ecological systems: Challenges and prospects. Environmental Modelling & Software, Apr., 1–7.

Grimm, Volker. 1999. Ten years of individual-based modelling in ecology: what have we learned and what could we learn in the future? Ecological Modelling, 115(2-3), 129–148.

Grimm, Volker, Berger, Uta, et al. 2006. A standard protocol for describing individual-based and agent-based models. Ecological Modelling, 198(1-2), 115–126.

Grimm, Volker, Berger, Uta, DeAngelis, Donald L., Polhill, J. Gary, Giske, Jarl, & Railsback,Steven F. 2010. The ODD protocol: A review and first update. Ecological Modelling, 221(23),

2760–2768.

Huston, M, DeAngelis, D, & Post, W. 1988. New computer models unify ecological theory.

BioScience, 38, 682–691.

Le, Quang Bao, Park, Soo Jin, Vlek, Paul L G, & Cremers, Armin B. 2008. Land-Use Dynamic Simulator (LUDAS): A multi-agent system model for simulating spatio-temporal dynamics of coupled human–landscape system. I. Structure and theoretical specification. Ecological Informatics, 3(2), 135–153.

Ralha, Célia G., Abreu, Carolina G., Coelho, Cássio G.C., Zaghetto, Alexandre, Macchiavello, Bruno, & Machado, Ricardo B. 2013. A multi-agent model system for land-use change simulation. Environmental Modelling & Software, 42(Apr.), 30–46.

Thiel-Clemen, Th. 2013. Information Integration in Ecological Informatics and Modelling. Pages 89 – 96 of: Wittmann, J., & Müller, M. (eds), Simulation in Umwelt- und Geowissenschaften, Workshop Leipzig. Leipzig: Shaker.

Vigueras, Guillermo, Orduña, Juan M., Lozano, Miguel, & Jégou, Yvon. 2013. A scalable multiagent system architecture for interactive applications. Science of Computer Programming, 78(6), 715–724.