tMANS - the Multi-Scale Agent-Based Networked Simulation for the Study of Multi-Scale, Multi-Level Biological and Social Phenomena

Matthias Scheutz (*), Greg Madey (*), and Sunny Boyd (**)
(*) Department of Computer Science
(**) Department of Biology
University of Notre Dame
Notre Dame, IN 46556, USA

We propose a multi-scale agent-based framework towards understanding and modeling multi-scale interdependent behavioral phenomena. This framework combines the ideas of agent-based modeling with that of hierarchies or levels of organization found in nature and allows for multiple levels in the model to interact at various time scales. We first summarize our rationale for pursuing agent-based models (rather than equation-based models) and then describe the proposed multi-level, multi-scale agent-based modeling framework formally, followed briefly by a discussion of how biological phenomena at different levels could be modeled in the framework. We list some of the requirements and desirable properties for a software simulation tool that can implement such multi-scale, multi-level models, briefly pointing to work in progress on the development of our "tMANS" tool ("the Multi-scale Agent-based Network System").

Keywords: multi-scale, multi-level, agent-based, distributed heterogeneous processing, collaboratory

INTRODUCTION

The functioning and behavior of complex, multi-cellular organisms can be analyzed at various levels, from the neuro-chemical level of molecular functions within and among neurons, to the level of neural pathways and neural networks, to the level of individual behavior based on sensory and motor capabilities, to the dynamics resulting from interactions among individuals. Each of these levels focuses on unique aspects of behavior, and computational models for the various levels comprising specific states that are defined at these levels.

While such models often can explain causal intra-level relationships, they are not well-suited to answer questions regarding inter-level relations. And while multiple models (for different levels) can be used to detect correlations, possibly even causal dependencies, among state variables at different levels, they do not allow for causal explanations, i.e., the determination of causal chains that explain why a particular higher-level behavior occurs given a low-level configuration, or why organisms always end up in particular low-level configurations after certain high-level exchanges. These questions can only be answered by a multi-scale, multi-level approach that takes the effects among and interactions between levels into account.

We propose a *multi-scale agent-based framework* towards understanding and modeling multi-scale interdependent behavioral phenomena. This framework combines the ideas of agent-based modeling with that of hierarchies or levels of organization found in nature and allows for multiple levels in the model to interact at various time scales. After summarizingour rationale for pursuing agent-based models (rather than equation-based models), we describe the proposed framework more formally, followed briefly by a discussion of how particular biological phenomena at different levels could be modeled in the framework. We list some of the requirements and desirable properties for a software simulation tool that can implement such multi-scale, multi-level models, briefly pointing to work in progress on the development of our "tMANS" tool ("the Multi-scale Agent-based Networked Simulation").

WHY AGENT-BASED SIMULATIONS?

The central idea of agent-based models is to view some of the entities in the model domain as agents that can act independently in the environment based on their inputs. Consequently, the changes over time modeled in an agent-based system are the state changes of each agent, which are given by rules, and the effects of the agents' actions. This is different from "equation-based models", where the changes over time are given by difference or differential equations that relate different state variables.

Both approaches have their advantages and disadvantages. Agent-based models allow for extreme heterogeneity (both in agent attributes and behaviors) and more flexible modeling of the spatial environments of the agents, while equation-based models are often capable of characterizing the bulk or average properties of very large systems.

Many of the differences between the two modeling paradigms (e.g., their theoretical approach or their particular mathematical formulation) vanish in actual implementations, where continuous differential equations have to be discretized to be implementable on standard von Neumann computers. Yet, one fundamental difference remains: while equation-based models presuppose a mathematically precise formulation of the various relationships among all state

variables (in terms of equations)—otherwise one cannot even meaningfully speak of an equation-based model—agent-based models only presuppose laws that guide the behavior of individual agents, but do not require knowledge of how these agent states themselves might be related. Consequently, agent-based models may be applicable (i.e., they can be formulated and investigated) even if the precise relationships (i.e., dependence and determination) among state variables are unknown. As such, agent-based models can actually help discover the equations and the formally precise relationships among state variables of agents if they are not known. In fact, this is exactly the reason why many agent-based models have been and are proposed.

We believe that agent-based models will be able to play out this *forte* even more in the domains, where multiple levels of organization and different temporo-spatial scales have to be integrated in an effort to investigate inter-level interactions, causal dependencies, and bottom-up vs. top-down constraints. What is needed is a way to extend the standard agent-based approach to allow for the simultaneous definition of different levels and temporo-spatial scales of organization, which can mutually influence each other, without imposing a theoretical commitment on the model designer on how upper-level entities relate to lower-level entities. The reason is that multi-scale modelers often do not want to, do not need to, or should not have to take a stances about the theoretical relationship among the entities they use in their models. Hence, an approach that forces them to be reductionist will *apriori* limit the modeling endeavor to a subclass of possible models. Rather, a modeling framework should be neutral on the question whether and how upper levels can be reduced to lower levels. Philosophers of science use the notion of "supervenience" here (e.g., see [Kim 1996]) to allow for dependence relation between upper and lower levels that may or may not be reductive. Roughly, the supervenience of entities at level i+1 over entities at level i means that changes in states of i+1-level entities are always accompanied by states changes of i-level entities (i.e., no change can occur at an upper level without a change at a lower level). However, the upper level changes may or may not be systematically related to lower level changes (if they are, the upper level entities and their properties may be reducible to lower level entities and their properties, if they are not, the upper level entities and their properties "emerge" from the lower level entities and their properties).

A MULTISCALE AGENT-BASED SIMULATION FRAMEWORK

We propose to generalize the idea of agent-based models to hierarchies of such models, where each level in the hierarchy is intended to model a different spatio-temporal scale. Different types of environments will be defined for each level together with the kinds of agents that populate them. Each agent (at each level) consists of a body and a control system. The body of a higher-level agent contains lower-level environments, thus connecting levels in a mereological (i.e., part-whole) manner (analogous to spatio-temporal levels as they are investigated by the special sciences such as physics, chemistry, biology, etc.). In the following, we will briefly sketch the theoretical framework (for space reasons, it is not possible to provide details here, e.g., such as the exact formal definitions of the respective update functions).

Different from most agent-based approaches, we do not take agents to be the basic entities, but rather "bodies" and "control systems" for three main reasons: (1) a notion of spatial extension of the physical representation of an agent might be crucial to capturing an understanding a particular phenomenon (independent of the level of description), (2) different control systems can control the same body and vice versa, (3) an explicit notion of agent body will allow us to connect multiple spatio-temporal scales in a natural way.

A *body* $B = \langle S_B, G_B, Int_B, E_B, U_B \rangle$ consists of sensors S_B, a geometry description G_B of its shape, internal states Int_B (e.g., energy sources), effectors E_B, and a mapping U_B from sensors, geometry, and internal states to effectors, geometry, and internal states (e.g., U_B could be given by a set of differential equations or difference equations).

A *controller* $C = \langle In_C, Comp_C, Out_C, U_C \rangle$ has input In_C, output Out_C, and computational states $Comp_C$ and a mapping U_C from input and internal to output and internal states (e.g., the mapping could be specified via condition-action rules or feedback equations from control theory).

An *agent* $A=\langle Body_A, Cont_A, Perc_A, Act_A, U_A \rangle$ then consists of a $Body_A$, a controller $Cont_A$, and two mappings $Perc_A$ and Act_A connecting them, which map sensory and internal bodily states to controller inputs and controller outputs to internal bodily states and effector states, respectively.

An *environment* $E=\langle Ext_E, Agents_E, Loc_E, U_E \rangle$ consists of external environmental states Ext_E, a set $Agents_E$ of possible agents for the environment, Loc_E the set of functions from $Agents_E$ to unique environmental states in Ext_E (the "agents' locations"), and the environmental update functions U_E which maps environmental states, a set of agents and their locations to new environmental states, a (possibly new) set of agents and updated agent locations.

Based on the mappings U_A defined for all agents in the environment, their computational and bodily states, and the environmental state at time t, it is then possible to determine the next state of the whole environment. Note that this definition of the environmental update encompasses many specific changes that can occur in agent-based models in one update step: (1) agent can cease to exist or come into existence (e.g., the set $Agents_E$' could be a subset or superset of $Agents_E$, or it could be in no containment relation to it), (2) action may be blocked by the environment (e.g., an agent's attempt to move forward even though there is wall in front), and (3) agent properties change based on environmental properties.

A standard agent-based model $M = \langle E, Init_E \rangle$ consists of an environment E and an initial state $Init_E$ of that environment.

While the above is only one of many possible formal characterization of standard agent-based models, it permits a generalization to multi-scale models (not easily achieved in other formalisms) by considering hierarchies of environments $E_1,E_2,..,E_n$, where each E_i is an environment defined for a spatio-temporal level i: A multi-scale agent-based model M can then be defined as $M=<E_n,Init_E>$, where E_n is a well-founded set of environments E_i (with initial states given by $Init_i$ each defined for a spatio-temporal level i ($0<i<n+1$) such that the following condition holds:

$$\forall E_i, E_{i+1}, A_{i+1} \in Agents_{E_{i+1}} : E_i \subseteq Intern_B$$

where $Intern_B$ is the set of internal states of body of agent A_{n+1} (i.e., the i-level environment is part of the body of the i+1-level agent).

The bodily update U_{Bi+1} is consequently based on the environmental update U_{Bi} and possibly additional factors. Hence, the theoretical approach leaves open whether a model is *reductive* in that all higher-level properties (i.e., states of the higher-level entities) are fully determined *and* reducible to lower-level properties, as it is possible to add additional states at each level that are not modeled in terms of lower-level states. Moreover, it is possible to introduce higher-level states for "emergent phenomena" and construct models to investigate behaviors that are typically characterized as "emergent" without having to solve the current widely debated foundational questions connected to "emergence" (e.g., how to theoretically define and formally construe the notion of emergence (e.g., see [Wimsatt 1997]).

From a theoretical view, the proposed framework is similar to other approaches that describe systems to be modeled in terms of system states and state transitions (e.g., discrete event systems [Zeigler 2003]). Yet, the proposed framework differs from common approaches in at least two respects: (1) it combines hierarchical models of agents and environments (by explicitly incorporating models of environments as part of a higher-level agents), and (2) it does not make any assumptions about the cardinality of the set of states at any level nor about whether time should be modeled in terms of intervals (e.g., between events) or in terms of periodic updates at a particular update frequencies (both options are possible at different levels). Note that the exact differences of the proposed models to models in other formalisms will depend on the details of the model (e.g., it might be possible to translate a multi-scale agent-based model into Pi-calculus if agents are treated as control processes without bodies and physical interactions).

From a practical point of view, we believe that the proposed framework will allow for and foster collaborations between researchers from different fields or different subfields within a field who otherwise would not interact much due to their difference in (1) domains, (2) explanatory aims, and (3) modeling approaches.

In the domain of cognitive science and economics, for example, it will be possible to investigate the decision-theoretic principles underlying group behaviors and the cognitive mechanisms that give rise to them (e.g., when people need to make decisions in time-critical situations, where the outcome affects the whole group and are thus subject to group pressure).

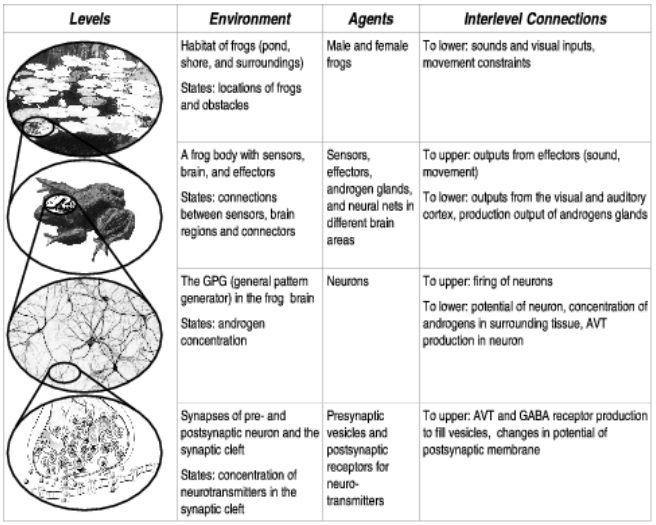

Figure 1: A four-level biological model used for the study of the effects of low-level synaptic and neuro-chemical processes on social interactions in bull frogs.

In the domain of biology, for example, it will be possible to integrate individual-based neural models of behavior (as proposed by neuroscientists) with population dynamics (as proposed by ecologists). In the simplest case, a two-level approach can investigate the neural effects of behavior in the context of a group of individuals. In such a setup, individuals are higher-level agents that consists of "neural environments", in which neurons (i.e., the lower-level agents) reside. It is then possible to investigate the effects of particular neural circuits, which give rise to individual behavior, on a group of individuals (e.g., the effects of repeated vocalizations on conspecifics).

Conversely, it is also possible to study the effects of group pressures on neural systems (e.g., the evolutionary adaptation of neural circuits that deal with alarm calls).

Figure 1 shows a specific biological model intended to span four levels of organization in bullfrogs: the low neuro-chemical level of synaptic interactions, to the level of neurons and neural network control, the level of organs and organization of body parts and bodies of animals, to the level of social interactions among animals. For each level the figure lists the employed "simulation agents" and their environment at that level as well as ways in which levels are connected by virtue of external environmental states.
For the sample model shown in Figure 1, the underlying biological question to be addressed is the mechanism of control of vertebrate social behavior. Complex interactions at multiple scales ranging from movement of individual molecules to social interactions among members of a

population ultimately determine the display of behaviors. Causal relationships across levels are poorly understood. This has implications for our understanding of basic behavioral processes, as well as perturbations critical to animal and human health.

We thus propose to use a simplified amphibian model system where a wealth of experimental data exist (e.g., [Boyd 1994,1997], [Hollis and Boyd, 2003], [Hollis et al. 2004]). Neurochemical, cellular, and behavioral processes in amphibians are nonetheless typical of vertebrates so the multi-level model will be broadly useful. At the lowest level, the model includes representatives from each of the three primary classes of chemical messengers: (1) fast-acting neurotransmitters, (2) peptides, and (3) steroids. At the next level, individual neurons integrate this chemical input and ultimately produce action potentials. Groups of neurons then perform specific functions, such as pattern generation (for the species-typical vocalization of the animal) and control of muscles in the larynx for production of sound. At the level of the individual organism, a variety of internal and external stimuli influence behavior. Our model focuses on vocal behavior, which is a common component of social interactions from humans to fish. We thus incorporate auditory input and internal physiological stimuli (such as gonadal androgen production) into the model at this level. Finally, at the level of social behavior, individual animals interact with each other. The peptide chemical messengers we model are specifically known to alter spacing between conspecifics. The mechanism for these effects is unknown, however. We expect the model to make novel predictions about the relationships between these neurochemicals and their effects on behavior. Model development should thus lead to new understanding of social behavior that would not otherwise have been achieved.

IMPLEMENTATION PLATFORM

Agent-based models are intrinsically discrete—for agents are discrete entities in the simulation with discrete update rules that determine their state transitions and actions. Consequently, it is fairly straightforward to map them onto distributed computational architectures, which is particularly desirable for multi-scale simulations, where higher-level agents consists of large numbers of lower-level agents. In the following, we provide a brief overview of our proposed "tMANS" system, *the Multi-scale Agent-based Networked Simulation*, which is currently under development and consists of seven major components:

- the open-source distributed component-based virtual machine for agent-based simulations called ADE together with its distributed multi-user graphical interface (`http://www.nd.edu/~airolab/software/`)
- the open-source agent-based simulation and experimentation environment SWAGES (`http://www.nd.edu/~airolab/software/`)
- the open-source agent-based NOM simulation environment (`http://www.nd.edu/~nom/`)
- a database server based on the open-source MYSQL server (`http://www.mysql.com`)
- a statistics server based on the open-source R system (`http://www.r-project.org/`)
- a visualization server based on JFreeChart
- the open-source webserver JIGSAW (`http://www.w3.org/Jigsaw/`)

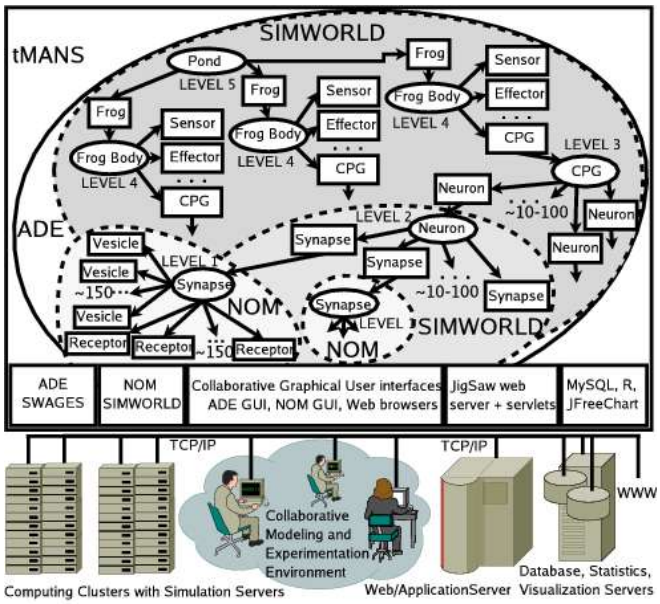

Figure 2. The proposed setup of tMANS for the biological model described in Figure 1. The lower part shows the hardware infrastructure, the upper part the distribution of agents over tMANS simulation components.

All parts of tMANS are implemented in JAVA and are thus platform-independent (to the extent that JAVA runtime environments exist for a given platform).

ADE

ADE is a distributed agent architecture development environment for virtual and robotics agents [Scheutz 2005, Andronache and Scheutz 2005, Andronache and Scheutz 2004a]. As such it allows for a very fine-grained level of distribution, namely the level of components of an agent architecture, which is much finer than possible in common multi-agent systems (such as JADE, RETSINA, AGENTFACTORY and others), of agent-based simulation environments (such as SWARM, REPAST, STARLOGO, EVO, SIMPACK, and others).

While this level of granularity is not critical to most common agent-based simulations as they typically deal with simple agents, which do not need to be distributed to speed up their computations, multi-scale models as proposed in this

paper consists of agents that recursively *contain* complete environments with agents. Consequently, part of a complex agent's control system deals with components that are themselves agent-based models (which may or may not contain further agent-based models). As such, the distribution of architectural components may become necessary to avoid computational overload of processors or lengthy simulation runs.

Consider, for example, the two-tier model of individual-based neural networks within a simulation of population dynamics (mentioned in the previous section). In such as setup, each agent might consist of thousands of neurons and their connections (in addition to sensors, effectors, and other bodily components). Hence, it might be beneficial to run the low-level neural network environment (with all the neural agents) of each high-level agent on a separate CPU (or host), while the remaining components of a high-level agent (together with the high-level environment) are run together on one CPU (or host).

The ADE virtual machine allows for such separations by virtue of system configurations, where each components in the system (e.g., the low-level agents, the low-level environments, the high-level agents, etc.) are associated either with a particular host or a set of hosts together with a set of constraints that determine where these components can instantiate and run. These configurations can be dynamically altered (e.g., based on availability of computational resources) and components can be moved from one host to another, should a reconfiguration of a running system be required (e.g., to achieve load balancing).

ADE also provides a distributed graphical user interface, which allows multiple users to define, start, test, modify, and terminate a distributed simulation. Effectively, each GUI instance connects to a registry server, which contains information about all existing instances of the ADE virtual machine. It gathers information about all existing components from all ADE virtual machines and displays them in a common workspace. The components can then be inspected and modified even while a simulation is running.

ADE has already been used successfully in several agent-based projects [e.g., Scheutz and Andronache 2004b].

SWAGES

SWAGES consists of two major components: (1) an artificial life simulation environment together with a set of agent models for 2D and 3D simulations, and (2) an experimentation server that can schedule, supervise, and run simulations on remote hosts. Both components consist of various subcomponents, each of which will be made available within the ADE environment (and can thus be arbitrarily distributed). That way the existing functionality of SWAGES will be preserved, while allowing users to access the functionality within the ADE graphical user environment (in addition to the web-based access, which is currently available).

The SIMWORLD simulation environment used by SWAGES is a general purpose 2D or 3D discrete or continuous environment that allows for continuous space, discrete time simulations. Agents can either be defined and run internally or alternatively they can exist external to the environment, in which case they will either have to connect via sockets or, if SIMWORLD is run within ADE, they can use ADE message passing mechanisms to connect. In the context of tMANS, this means that *i*-level agents (which are ADE components) will connect to the specific *i*-level SIMWORLD instance (another ADE component), in which they reside. Updates in higher-level SIMWORLD instances will thus involve updates in multiple lower-level SIMWORLD instances, which will be distributed over a network of hosts so as to minimize the update interval.

The experimentation server effectively provides a grid environment (e.g., in the sense of CONDOR and other grid engines) by virtue of maintaining a list of possibly heterogeneous hosts that can be used for simulations, updating their state, and starting simulations on them. Moreover, the experimentation server monitors the progress of simulations and is able to restart them if they should have terminated prematurely. Different from CONDOR (or SUN's grid engine), SWAGES does not require any demon to run on participating hosts, but rather uses standard secure shell connections to those hosts to check availability and to start simulations. This allows for easy maintenance of the simulation grid and, furthermore, facilitates the dynamic formation of heterogeneous ad-hoc clusters of computers that can be used for simulations (e.g., a set of computers might become available only from midnight to 8 a.m.).

Taking again the two-tier example of neural individual based-models in the context of population-based models of behavior, each neural network is run in a separate SIMWORLD instance together with all the neuron agents. Hence, for networks with n neurons at least $n+1$ ADE components will be required to run on each host. Moreover, for k higher-level agents at least $k+1$ ADE components are needed for the higher-level environment also running in SIMWORLD. Hence, the overall update cycle of the high-level environment is given by the update times of the lower-level environments, the communication overhead and time lag across the network connection, the computational overhead in ADE, and most importantly the distribution of the simulation over multiple hosts. If k is smaller than n and assuming that the high-level updates are not significantly more expensive than the lower-level updates, $k+1$ hosts would allow for an effective distribution of the computation (more complex distribution schemes are, of course, possible).

NOM

The NOM (Natural Organic Matter) server is a web-based collaboratory (a collaborative laboratory) designed and used to support environmental scientists studying the complex properties and behaviors of microorganisms and organic molecules in the soil [Xiang, et al 2004, Huang et al. 2005, Huang, Xiang, and Madey, 2004]. It does this by providing a

centralized cluster of simulation servers and web-based collaboration services. Scientists do not need to download, build, and install the applications or related software on their own computers, which can be a tedious task. Scientists can share the expensive computational resources, such as large-scale databases and specialized data mining tools, which may not be readily available to small research groups. Deployment of the simulators on the Web promotes their use as a collaboratory for geographically separated scientists and engineers. In order to support collaborative work, the NOM collaboratory is built using the Sun Java 2 Enterprise Edition (J2EE) and relational database technologies. The NOM collaboratory includes Web-based simulators, data analysis packages, simulation configurations, and communication tools such as discussion board and chat room. Scientists can access these Web-based simulators through a Web browser. They choose a particular simulation program and provide the input parameters that are then stored in the remote database. After they submit their configuration, simulations are invoked at a remote computer. When simulations are finished, users are notified by Email. They then can access the data output from the referred site. These data results not only include the raw data but also the graphic results that are generated by the data analysis packages using statistical and data mining technologies. The built-in functionalities of the NOM collaboratory allow scientists to share all their simulation results, data, and information with others.

The tMANS database server
The tMANS database server will be based on the NOM collaboratory, which provides the following functions:

Distributed computational resource utilization: Users can configure and invoke their simulations through a Web interface. Computational resources on remote servers are allocated transparently by a job manager.

Data analysis: Users can view their simulation data, generated by the NOM simulations, from a Web-based interface. These data and information are represented in various types of graphs (bar charts, pie charts, line charts) and statistical reports by employing data query and data mining technologies.

Information sharing: Users can share the results of their simulation, the molecule definitions, and the simulation configurations through web interfaces and a search engine.

Data repository: The databases are used to store the internal data that are generated from the NOM simulators. Additionally, external data, including publications, technical reports, and other forms of dissemination, which are uploaded by scientists, are also stored in the database.

Secure access: Users do not have the same level of access privileges to all the tools in the NOM collaboratory. Some tools, such as the "Molecule validator" and the "NOM simulator", can only be accessed by authorized persons. Users have access to their own simulations, and other users cannot access data that have not been authorized for public usage.

The tMANS statistics server
The tMANS statistics server will be implemented as an ADE component wrapper around the open-source statistics software R. The server will accept a set of predefined commands that will allow users to new create tables dynamically from existing tables in the database (via the database server) and perform operations that are specified in R. The numeric or graphical output from R can then be displayed or saved.

The tMANS visualization server
The tMANS visualization server will allow for various 2D and 3D visualizations of database tables or statistical results from the statistics server using Jigsaw, Jakarta Tomcat for servlets, and the open source JFreeChart visualization classes.

JIGSAW
The httpd webserver JIGSAW will be used as the default interface to the tMANS system. It is particularly intended for casual or unexperienced users who do not themselves *design* agent-based models (thus do not need a direct interface to the ADE system), but might run, configure, or adapt existing models (e.g., by changing the initial conditions, distribution of agents in the simulation environment, etc.). It will be based on the current web-based interfaces to SWAGES and NOM and will allow users to (1) setup, schedule, and run whole sets of multi-scale agent-based simulation experiments, and (2) view, analyze, and archive their results using a standard web browser (e.g., Mozilla or Netscape). Jigsaw will use the open source Jakarta Tomcat servlet engine to provide servlet wrappers for the statistics and visualization servers.

RELATED WORK
There are various other projects that are aimed at supporting collaborations among scientists by providing a web-based portal for sharing data and information across a heterogeneous network of computers.

The ESP2N Earth Science Partners' Private Network (`http://dml.cs.ucla.edu/projects/dml_esip/`), for example, uses a virtual private network to allow researchers to query a data base securely over the Internet using the high-performance, flexible query system Conquest (e.g., see [Shek et al. 1996]). In addition, ESP2N employs the semantic markup language SEML to define and capture experiments. Such a language can have great utility in providing a standardized format for specifying input-output relationships and sharing them in a unified way independent of the details of particular experiments. A similar approach is currently considered for tMANS to allow different components to exchange data, as currently experiments in

SIMWORLD and NOM use their own internal formats to represent and store experimental data. However, rather than changing the internal formats in retrospect and imposing a general language for all tMANS components, each component will have to implement a translator that converts from and to the common experiment representation format.

Different from ESP2N, which does not include a simulation environment, the JAS Sim2Web project is built around the JAS agent-based simulation environment (`http://wf.econ.unito.it/sim2web`). Sim2Web is intended for web-based economic and financial simulations and uses the JAS libraries for simulation and Zope for the web-based interface and user management. JAS is a discrete-event simulation engine. As such it is in some important ways different from the proposed simulation framework (as described earlier) and does not directly support the implementation of hierarchical agent-based models.

CONCLUSION AND FUTURE WORK

We have proposed a multi-scale agent-based framework to model phenomena at different levels of organization even if the exact dependence and determination relations are not known. Such models provide insights into the inter-level dynamics of complex systems and might help scientists to discover and formulate equation-based models for multi-scale phenomena, which would otherwise be difficult (if not impossible) to detect.

We are in the process of defining a detailed four-level model for a biological domain, which spans the range from neurochemical to social levels of organization and behavior. To support the implementation of such model on computers and experiments with them, we have proposed the comprehensive simulation tool tMANS, which allows for the definition, implementation, and experimentation with fully distributed multi-scale agent-based models.

Current work on tMANS is focused on a tight integrating of previous and existing platforms (as described in the previous section) in an effort to standardize and test the interfaces among components and verify that the overall system design is viable and will scale to large agent-based models. A first prototype version of tMANS is expected for late 2005.

ACKNOWLEDGEMENTS

The material presented in this paper is based in part upon work supported by the National Science Foundation under grant ITR/AP-DEB 0112820 for the second and grant IBN-0235903 for the third author.

REFERENCES

Andronache, Virgil and Scheutz, Matthias, 2005, “ADE - An Architecture Development Environment for Virtual and Robotic Agents”. IJAIT (forthcoming)

Andronache, Virgil and Scheutz, Matthias, 2004a, “Integrating Theory and Practice: The Agent Architecture Framework APOC and its Development Environment ADE”. In Proceedings of AAMAS 2004, ACM Press.

Boyd, S.K., 1994, “Arginine vasotocin facilitation of advertisement calling and call phonotaxis in bullfrogs.” Hormones and Behavior 28:232-240.

Boyd, S.K., 1997, “Brain vasotocin pathways and the control of sexual behaviors in the bullfrog”. Brain Research Bulletin 44: 345-350.

Hollis, D.M. and Boyd, S.K., 2003, “Characterization of the GABAA receptor in the brain of the adult male bullfrog”, Rana catesbeiana. Brain Research 992:69-75.

Hollis, D.M., Goetz, F., Roberts, S., and Boyd, S.K., 2004, “Acute neurosteroid modulation and subunit isolation of the GABAA receptor in the bullfrog, Rana catesbeiana.” Journal of Molecular Endocrinology 32:921-934.

Huang, Y., Xiang, X., Madey, G., and Cabaniss, S, 2005, “Agent-based Scientific Simulation Using Java/Swarm, J2EE, RDBMS and Autonomic Management Technologies”, IEEE Computing in Science & Engineering .

Huang, Y., Xiang, X., and Madey, G., 2004, “A Self Manageable Infrastructure for Supporting Web-based Simulations”, 37th Annual Simulation Symposium at the Advanced Simulation Technologies Conference 2004 (ASTC'04), Arlington, VA, April 2004

Kim, Jaegwon, 1996, Philosophy of Mind. Westview Press, Boulder and Oxford.

Scheutz, Matthias and Andronache, Virgil, 2004b, “Architectural Mechanisms for Dynamic Changes of Behavior Selection Strategies in Behavior-Based Systems”. IEEE Transactions of System, Man, and Cybernetics Part B: Cybernetics, Vol. 34, No. 6.

Scheutz, Matthias, 2005, “ADE - Steps Towards a Distributed Development and Runtime Environment for Complex Robotic Agent Architectures”. Applied Artificial Intelligence (forthcoming).

Shek, E.C., Muntz, R.R., Mesrobian, E., and Ng, K., 1996, “Scalable Exploratory Data Mining of Distributed Geoscientific Data”, Second Intl Conference on Knowledge Discovery and Data Mining, Portland.

Wimsatt, William C., 1997, “Aggregativity: Reductive heuristics for finding emergence”. Philosophy of Science 64, 372-384.

Xiang, X., Huang, Y., Madey, G., Cabaniss, S., Aurthurs, L., and Maurice, P., 2004, “Modeling the Evolution of Natural Organic Matter in the Environment with an Agent-based Stochastic Approach”, Natural Resource Modeling Journal, December 2004.

Zeigler, Bernard P., 2003, “DEVS Today: Recent Advances in Discrete Event-Based Information Technology.” MASCOTS.