1.0 Introduction

Scholarly interpretations of archaeological evidence from any given period are traditionally based upon static approaches that look at a fairly limited number of factors affecting social change. Reliance on such approaches is understandable given the temporally fragmented nature of data from the past. Regardless of whether or not researchers use excavation materials, textual sources, survey results, or other sources of data to interpret past social behaviors, these techniques can only give a limited picture of how societies may have functioned in a given period. However, understanding the root mechanisms that create the social trajectories and outcomes that scholars of the past interpret requires a more sophisticated analysis that looks at the structural factors and dynamic processes that shape a society in any given period. Over the past few decades, researchers have been increasingly investigating the interactions of social processes (e.g., economic exchange, marriage patterns, agricultural practices) and natural processes (e.g., hydrology, climate, human disease), or what can be termed socioecological interactions, in great part due to the works and influence of Braudel (1980), Butzer (1982) and others. Given this need to address socioecological interactions, along with growing computational power over the past few decades, archaeologists and anthropologists have incorporated computer modeling and simulation in analytical techniques that attempt to explain how observed social phenomena may have emerged (Reynolds 1986; Lansing 1991; Doran et al. 1994; Kohler et al. 2000).

In addition, it is clear that those who study the past have varied interests and theoretical perceptions that may require a simulation or modeling approach that entails examination of diverse, concurrent processes across a range of scales and levels of detail. Valid questions within this range might, for example, relate to the aggregate social development of large regions over multi-generational time spans down to details of participation of individuals in household activities, over a span of days or weeks, for specific settlements. Given such diverse scales and varied theoretical perspectives, it is a significant technological challenge to be able to faithfully represent important simultaneous processes and their interactions at different social, spatial, and temporal scales using computational approaches. In response to this challenge, social scientists at the University of Chicago and the University of Edinburgh, along with researchers at the Argonne National Laboratory's Decision and Information Sciences Division, have embarked upon a project that seeks to develop a new simulation software framework to model key socioecological processes and process interactions for Bronze Age (3000-1200 B.C.) settlement systems and societies, for both the northern rain-fed and southern irrigated regions of Mesopotamia. We have named our new simulation system "ENKIMDU," after the ancient Sumerian god of agriculture and irrigation. An overview of the ENKIMDU system's design philosophy, capabilities and limitations, and range of applicability is presented below, along with a brief discussion of some of the underlying advanced simulation technologies that have made ENKIMDU's development feasible.

Although the intended primary audience of this paper is anthropologists and archaeologists, our discussion will attempt to explain how the underlying technological architecture of ENKIMDU, and particularly the software infrastructure provided by

Argonne's general-purpose simulation frameworks, facilitates its use as a tool for simulating complex social and natural systems. For scholars comfortable with the Java computer language, computer code examples of some of the principal elements of the simulation system are provided in an appendix. The language in the main text, however, will be kept as free of computer science jargon as possible without sacrificing meaning.

2.0 Overview of the Simulation System

In many respects, ENKIMDU reposes upon well-established software engineering design precedents and upon advances made in prior groundbreaking social simulation studies (e.g., Kohler and Van West 1996; Epstein and Axtell 1996; Kohler et al. 2000). From a computer science perspective, our general approach can be categorized as both "object-oriented" and "agent-based." It is also a "process-oriented" example of a "discrete event simulation." It can be classified both as a simulation and as a simulation software "framework." ENKMDU is implemented entirely in standard Java. The key simulation software system attributes and characteristics implied by this categorization can be briefly summarized as follows:

2.1 <u>ENKIMDU as an Object-oriented Simulation</u>

The use of software "objects" to partition and simplify computer program data structures and information flows dates back several decades, to the SIMULA software language (Dahl and Nygaard, 1966). Under current "object-oriented" software development practice (e.g., Wirfs-Brock et al. 1990), as applied to a simulation, the specific region, epoch, etc., to be studied, or "problem domain," is decomposed, or subdivided, into objects representing the principal elements that comprise the domain. Information describing the domain in space and time is then assigned to attributes of these "domain objects." The resulting logical structure is known as a "domain object model." A domain object model for a simulation of a settlement system and the landscape in which it is situated might include objects representing such elements as villages, dwellings, persons, households, fauna, vegetation cover, soil layers, water bodies, etc. Each object maintains information on its current state, and regulates access to its state variables. In fully object-oriented simulation systems such as ENKIMDU, the domain objects are also responsible for expressing their own dynamic behaviors; for socioecological simulations these behaviors may include such potentially complex functions as formulating and implementing adaptive responses to external stimuli.

2.2 <u>ENKIMDU as an Agent-based Simulation</u>

An "agent" in a simulation is an entity characterized by autonomous behavior, usually performed with reference to local rules, and in response to local conditions. Working definitions of agent-based models (ABMs) diverge substantially past that point. By the most liberal definition, virtually any fully object-oriented simulation (see above) could be considered an ABM, with the domain objects in the roles of agents. Thus, for example, if an object representing a soil layer for a segment of a simulated landscape were to implement the behavior of evolving its own state variables (e.g., vertical moisture profile, compaction, etc.) forward in time using its own behavior formulation (perhaps using its own "embedded" hydrology and erosion models), and with reference to local weather, vegetation cover, etc., it could be considered an agent. Our simulation system is rich in examples of this sort of non-social "agent-like" behavior. In the context of social simulations (e.g., Epstein and Axtell 1996), however, it is customary to restrict the definition of agent further. Here, cognitive agents, or social agents, in a simulation are typically used to represent individual persons or organizations. They are imbued with the ability to form and respond to perceptions, and adapt to, and possibly learn from, changing local environmental conditions and social stimuli. In common with most prior agent-based social simulations, our cognitive social agents exhibit "bounded rational" behavior (Simon 1957), in which agents make decisions and take actions that are at best locally optimal, informed by incomplete and imperfect perceptions.

2.3 <u>ENKIMDU as a Process-oriented Discrete Event Simulation</u>

ENKIMDU employs the software mechanism of time-ordered discrete events, or messages, to initiate, regulate, and track dynamic processes, to carry communications between agents, and to notify agents and other domain entities of significant environmental occurrences. Since these events can be specified to occur at any simulation time, rather than having to occur only at uniformly spaced time steps, the progress of each of the various modeled dynamic processes can be tracked at whatever temporal resolution is best suited to it. This is a very important consideration for highly heterogeneous simulation domains such as those addressed by ENKIMDU, in which diverse social and environmental processes typically proceed at very different rates.

Simulations that make use of discrete events are generally classifiable as either "event-driven" or "process-oriented." In "event-driven" simulations, event messages passed among entities convey contextual information on some occurrence of interest to a receiving entity, <u>and</u> specify what action the receiving entity should take in response. In contrast, process-oriented discrete event simulation systems such as ENKIMDU use events to convey contextual information among agents and other domain entities, but do not attempt to prescribe the actions that the receiving entities should take; that is up to the agents themselves, in keeping with the spirit of agent-based, object-oriented software design.

2.4 <u>ENKIMDU as Computer Simulation and as Simulation Framework</u>

The ENKIMDU system in its present prototype configuration is capable of performing detailed socioecological simulations of relatively simple agricultural/pastoral settlements in ancient Mesopotamia. It has been "tuned" to this particular problem domain in order to address our specific research questions about the development and sustainability of ancient Mesopotamian society. In addition to addressing this specific simulation domain, ENKIMDU can serve as a foundation for building new simulation systems that address other complex domains.

ENKIMDU was designed, in accordance with standard best practices for object-oriented software development, to exploit commonalities in domain data structures and processes. Also, as will be discussed in detail later in this paper, ENKIMDU has been constructed using common components and design patterns from two general-purpose object-oriented simulation frameworks developed at Argonne. The idiosyncrasies of ancient Mesopotamian society and of the natural and anthropogenic landscapes of the ancient Near East are dealt with in the ENKIMDU design as modulations upon relatively stable, common design themes. These design themes are expressed via ENKIMDU's high-level object schema: a structured collection of formal domain object definitions for major classes of societal and environmental entity. In light of the flexibility afforded by this design approach, we see a role for ENKIMDU as a foundation upon which other diverse socioecological simulations can be built. We have already taken the first steps in that direction, via a pilot study in which ENKIMDU is being extended to address fine-scale agroeconomic sustainability issues for villages in modern Thailand.

2.5 Enabling Technologies for ENKIMDU Framework Development

ENKIMDU software system development has been supported by two advanced modeling and simulation technologies developed at Argonne National Laboratory. This system's basic software structure has been implemented with the help of Argonne's Dynamic Information Architecture System (DIAS; Christiansen 2000a). DIAS is a generic object-oriented Java computer simulation framework that makes it feasible to build and run complex simulation scenarios that involve thousands of heterogeneous domain objects interacting via scores of concurrent dynamic processes.

The FACET framework (Christiansen 2000b) is an object-oriented Java software toolkit for constructing agent-based, object-oriented models of simple to highly complex societal processes. We have used FACET to construct models of persistent social behavior patterns for individuals and organizations that operate within the social and environmental context of larger DIAS-based holistic simulations. Several detailed examples of application of both DIAS and FACET capabilities in the construction of the ENKIMDU simulation system are provided later in this paper.

2.6 Distinguishing Characteristics of the ENKIMDU Framework

As the preceding discussion mentions, the ENKIMDU framework shares some fundamental structural characteristics with several other simulation software systems. ENKIMDU tends to differ from most agent-based simulation systems that have been used to date in socioecological modeling investigations in its more extensive employment of, and support for, simulation of close coupling of heterogeneous dynamic processes. ENKIMDU also garners advantages in flexibility, scalability, and ease of code re-use, as a direct consequence of its DIAS underpinnings.

Under the domain object-centric simulation paradigm supported by DIAS, and in accordance with a basic tenet of the object paradigm, the objects that represent the problem domain are responsible for expressing their own dynamic behaviors which they accomplish by invoking appropriate simulation models. These simulation models, which can include proven, existing models and/or new, purpose-built codes, are not embedded directly in the software objects but are instead linked to their “owner” objects at run-time, for execution by the owners as they are needed, based on simulation context. Under DIAS, each model is formally defined as being able to implement a specific *abstract* behavior called out in the definition of a class of domain object. When the abstract behavior is invoked by the object, the model that is dynamically linked to the behavior will be triggered to run. Thus, various alternative model formulations, perhaps embodying different hypotheses or different levels of detail, can be invoked “painlessly,” without changing simulation source code, at run-time, a key advantage in flexibility and ease of use as a research tool.

Each model in a DIAS-based simulation converses with the simulation in the language of the relevant domain object attributes; a model may interact only with its owner object and with the objects that its owner object “knows;” models never need to interact directly with other models. This approach pays major dividends in simulation scalability as more and more process models are added to a simulation framework, since there is no need to maintain an exponentially growing set of model-to-model linkages and data protocols as models are added. Each new model need only know about the limited subset of domain objects that it can affect, and/or those objects from which it draws its inputs.

ENKIMDU carries explicit, fine-scale representations of the dynamics of several key natural processes, operating concurrently with, and interacting with, fine-scale dynamics of the social processes carried out by the social agents. ENKIMDU simulations do employ social agents, but these agents operate within a “decision space” that includes a substantial fine-scale environmental component in addition to the obligatory societal component, thus providing an especially rich context for formulating and expressing agent behaviors. By the same token, while ENKIMDU simulations encompass key entities and dynamic processes pertaining to natural landscapes, these simulated landscapes are also subject to perturbations, deliberate or otherwise, due to the fine-scale modeled activities of the individual social agents. This balanced approach has allowed us to make good use of some of the key features of the DIAS framework for modeling fine-scale interactions and feedbacks between heterogeneous processes. The subsets of domain object attributes used as model inputs and outputs may easily <u>overlap</u> from model to model. Interactions and feedback loops involving multiple concurrent dynamic processes therefore ensue implicitly, wherever such overlaps occur, and need not be explicitly called out in the simulation design.

3.0 Constructing the ENKIMDU Dynamic Domain Object Model

3.1 <u>Software Object Representation of the Problem Domain</u>

Our modeling representation of the domain of interest is composed of (a) objects representing the components, both concrete and abstract, of the real-world domain, and (b) simulation models that implement the dynamic behaviors of these domain objects. The discussion that follows will first identify and describe some of the key classes of object in the modeling domain, and then move on to discuss the object behaviors and the simulation models that have been built or adapted to implement them.

The ENKIMDU system prototype as it currently stands contains formal Java class definitions for over 300 Java language object classes specific to ENKIMDU, in addition to the hundreds of DIAS and FACET infrastructure library software objects supporting the simulation. These object classes represent essentially all of the software objects that comprise specific components of the real-world domain – ancient Mesopotamian society and its infrastructure, embedded in its natural environment – that we are attempting to simulate. This collection of domain objects represents both concrete instances (e.g., fields and plow teams) and abstractions (e.g., perceptions and strategies).

A partial, simplified view of our domain object decomposition is provided in Figure 1, which focuses on the natural, or environmental, component of the domain, and Figure 2, which deals with the societal component. These figures use the formalism of Unified Modeling Language (UML) static structure diagrams to show interrelationships among objects.

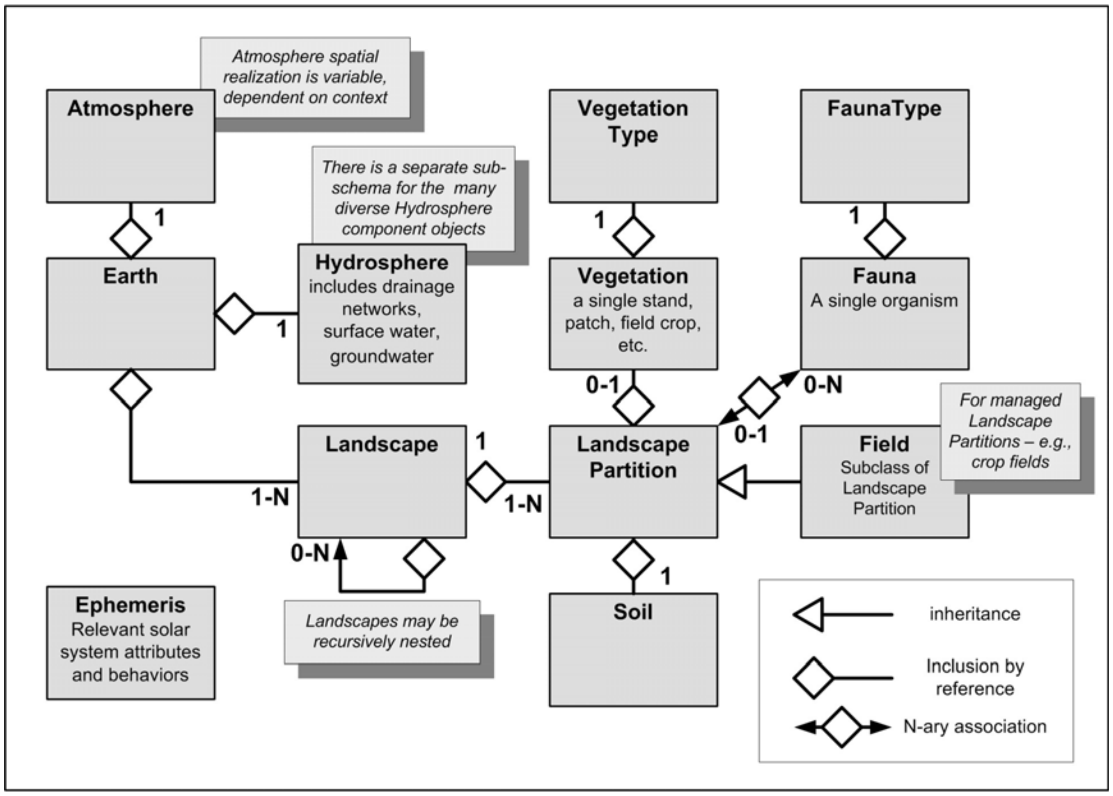

Atmosphere spatial realization is variable, dependent on context
Atmosphere
There is a separate sub-schema for the many diverse Hydrosphere component objects
Vegetation Type
FaunaType
1
1
1
Earth
Hydrosphere includes drainage networks, surface water, groundwater
1
Vegetation a single stand, patch, field crop, etc.
Fauna A single organism
0-1
0-N
0-1
For managed Landscape Partitions – e.g., crop fields
Landscape
1
Landscape Partition
Field Subclass of Landscape Partition
1-N
1-N
0-N
1
Ephemeris Relevant solar system attributes and behaviors
Landscapes may be recursively nested
Soil
inheritance
Inclusion by reference
N-ary association

Figure 1. Simplified partial schema for ENKIMDU natural systems objects.

The components of the natural landscape are portrayed in Figure 1 as included by reference in a single Earth object. The term, "inclusion by reference," here signifies that an object, or an instance of an object class, carries a reference, or pointer, to another object, so that it can, for example, send messages to that object. As Figure 1 indicates, any instances of the Earth object class, there will only be one in this specific case, will carry references, or pointers, to other objects representing the Earth's atmosphere and hydrosphere, as well as references to a recursive hierarchy of Landscape objects representing major spatial "compartments" on the Earth's surface. Landscapes can be further subdivided into Landscape Partition objects which represent fairly homogeneous landscape mosaic "patches" with characteristic vegetation and soil types. Figure 1 includes one example of object inheritance. The Field object class is a subclass of the Landscape Partition class. This means that all instances of the class, Field, automatically "inherit" all of the attributes of the Landscape Partition class. Field objects carry additional attributes, that is information associated with tillage, land tenure, etc., that are *not* found in Landscape Partitions. The "N-ary," or "many-to-many" type of object association identified in Figure 1 simply refers to cases where instances of a particular type of object may carry references to several instances of another class of object, such as a Landscape having references to any and all Landscape Partitions that lie within its geospatial boundaries.

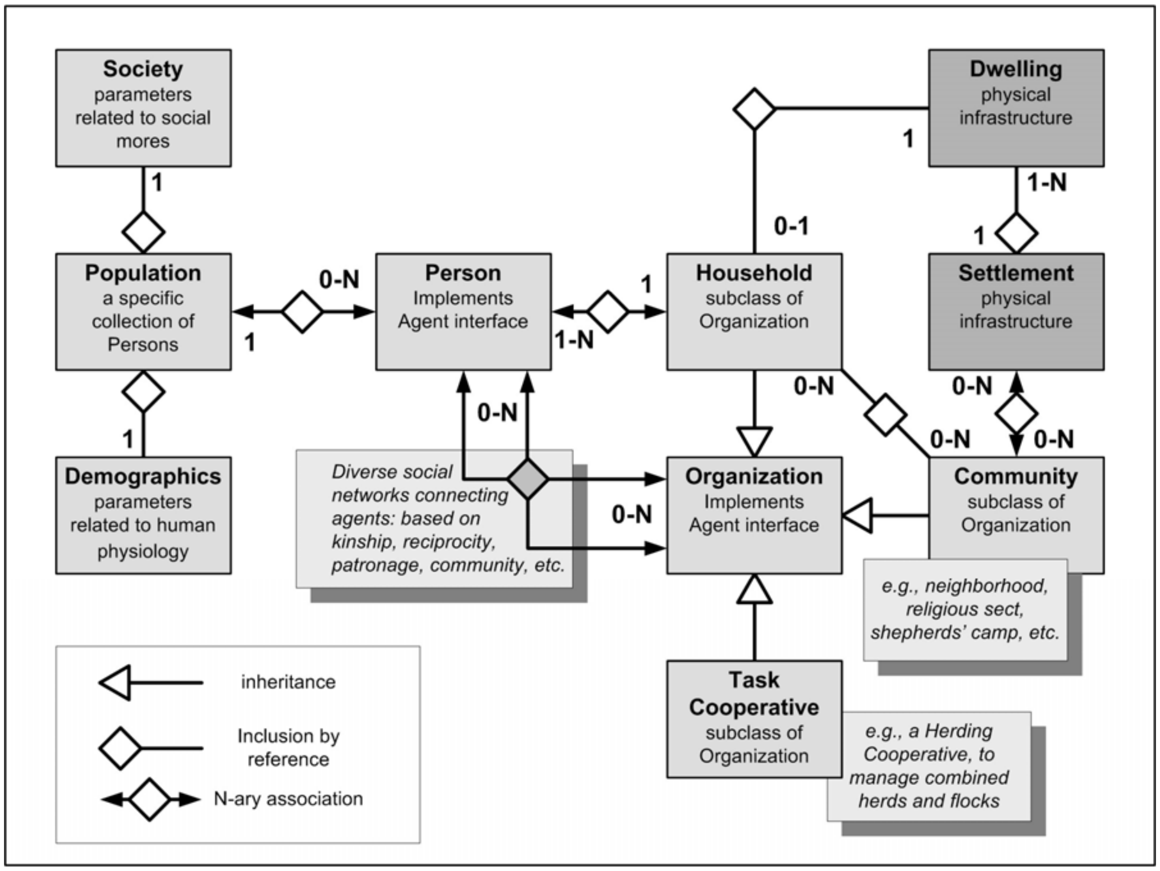

Society
parameters related to social mores
Dwelling
physical infrastructure
1
1
1-N
0-1
1
Population
a specific collection of Persons
0-N
Person
Implements Agent interface
1
Household
subclass of Organization
Settlement
physical infrastructure
1
1-N
0-N
0-N
0-N
0-N
0-N
1
Demographics
parameters related to human physiology
Diverse social networks connecting agents: based on kinship, reciprocity, patronage, community, etc.
0-N
Organization
Implements Agent interface
Community
subclass of Organization
e.g., neighborhood, religious sect, shepherds' camp, etc.
inheritance
Inclusion by reference
N-ary association
Task Cooperative
subclass of Organization
e.g., a Herding Cooperative, to manage combined herds and flocks

Figure 2. Simplified partial schema for ENKIMDU societal systems objects.

Figure 2 illustrates some of the key object components of the ENKIMDU societal systems representation. In the present ENKIMDU design, both Organizations and individual Persons implement the Agent interface, which means that they are capable of carrying out a range of behaviors reserved for social agents. These include such activities as adaptively responding to stress and to opportunity, participating in reciprocal exchanges with other agents, etc. Several specialized subclasses of Organization are defined to deal with cooperative behaviors at various scales (e.g., household vs. community vs. state) and in various modes. Each Person object carries a reference to a Population object instance, which represents a population, or micro-population, that has a distinctive set of social mores represented within a Society object and particular settings for population dynamics parameters such as birth and death rates that are represented within a Demographics object. Figure 2 also calls out the multiple overlapping social networks based on kinship, reciprocity, coercion, community, etc., that link the ENKIMDU social agents.

The structure of the object representation of the ancient Mesopotamian subject domain having been sketched out, we now turn to specification of the dynamic processes by which these objects affect each other and evolve over time.

3.2 Adding Dynamics to the Domain Object Model

As noted earlier, the DIAS infrastructure has provided the development team with a means of ensuring that the ENKIMDU simulation platform will remain flexible, manageable, and extendable as it grows in scale, scope, and depth over the course of its useful lifetime. The mechanisms responsible for this advantage are as follows:

- Abstraction of domain object behaviors. DIAS Entity objects declare their behaviors abstractly. Linkage of these abstract behaviors to appropriate modeling functionality is a context-dependent, run-time activity. Specific domain object classes can use this DIAS feature simply by declaring themselves to be subclasses of DIAS Entity, so that they can inherit the Entity class attributes and capabilities.

- Encapsulation and formalization of model functionality. DIAS interface objects formally define and isolate models and other applications that implement domain object behaviors, and map them to specific domain object behavior aspects. In a DIAS-based simulation, models may communicate only with domain objects, and never directly with each other. This has the effect of reducing the number and complexity of model linkages needed and improves code reusability.

DIAS Entity-based domain objects, such as Household and Landscape objects in the ENKIMDU prototype, carry a collection of “Aspect” objects that *abstractly* represent the specific behaviors for the subclass. If a particular aspect of a domain object’s behavior is needed in a given simulation context, a DIAS “Process” object standing as proxy for a specific model implementation is linked at run time to that Aspect to allow the domain object to exhibit the behavior. This indirect, flexible, context-driven linkage is illustrated in Figure 3.

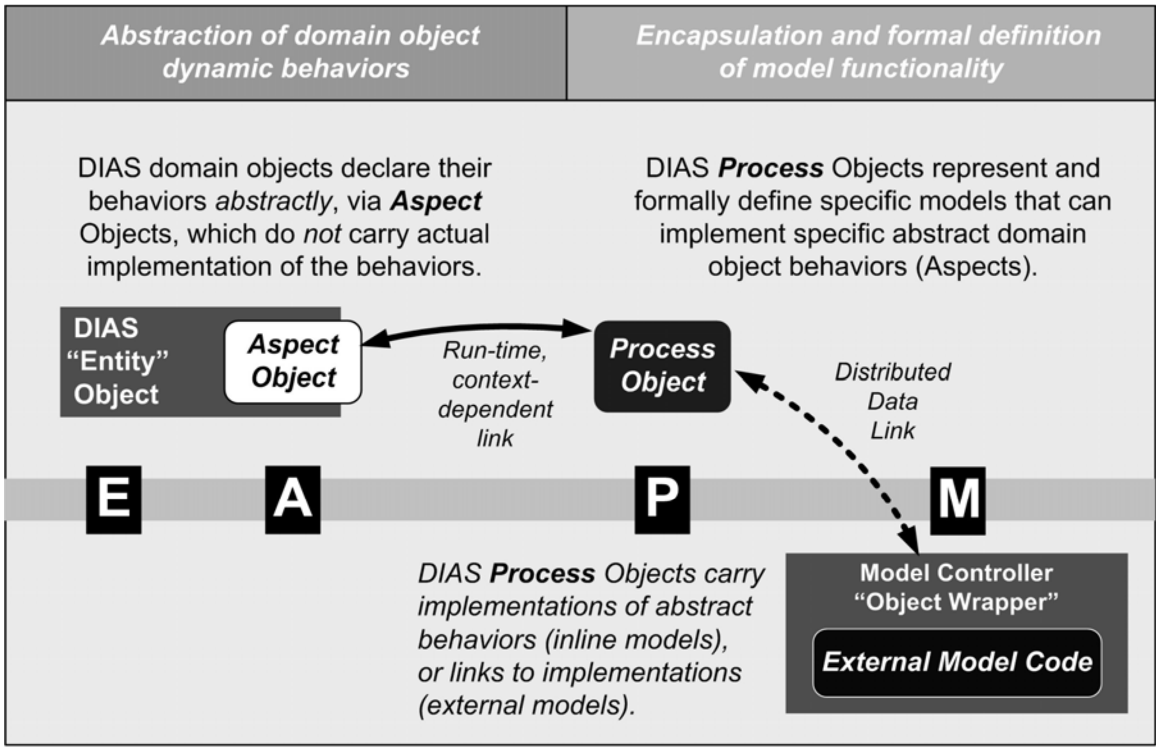
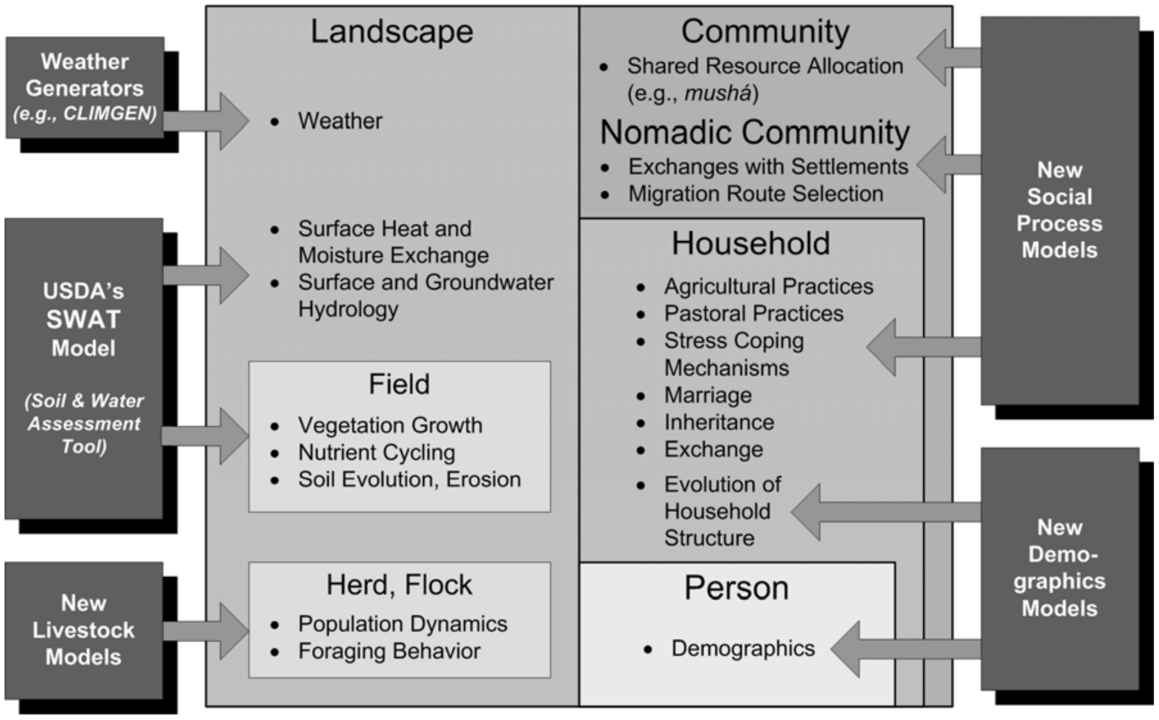

Abstraction of domain object dynamic behaviors
Encapsulation and formal definition of model functionality
DIAS domain objects declare their behaviors abstractly, via Aspect Objects, which do not carry actual implementation of the behaviors.
DIAS Process Objects represent and formally define specific models that can implement specific abstract domain object behaviors (Aspects).
DIAS "Entity" Object
Aspect Object
Run-time, context-dependent link
Process Object
Distributed Data Link
E
A
P
M
DIAS Process Objects carry implementations of abstract behaviors (inline models), or links to implementations (external models).
Model Controller "Object Wrapper"
External Model Code

Figure 3. DIAS architecture for dynamically linking models to domain objects.

The DIAS Process objects carry formal definitions of the capabilities and limitations of specific, actual model implementations that are available to the simulation framework. DIAS Process specifications identify *which* object behaviors, as represented by the abstract Aspect objects, the application is qualified to implement. Each DIAS Process also carries a formal specification of the input and output data dictionaries of the application it represents, along with a link to the actual application code itself wherever it resides in a distributed network. Each actual model or other application is accessed by its DIAS Process object via a "Model Controller" object, which acts as an object wrapper for the model, hiding model internal details from the rest of the simulation system. Model Controllers are responsible for translating DIAS object state variables into model-specific input terms and for translating model-specific output back into the form of updates to DIAS object attributes. This Entity – Aspect – Process – Model (or E-A-P-M) approach greatly facilitates plug-and-play flexibility for context-sensitive specification of alternative model formulations that address the same domain object behaviors but in different ways.

Figure 4 presents a simplified schematic representation of many of the classes of software object that make up our ancient Mesopotamian simulation domain, linked to a suite of simulation model components that have been assembled to provide dynamic behavior. The major classes of domain entity (Field, Household, etc.) are shown as the large blocks occupying the center of the figure. Modeled dynamic behaviors of these simulation entities are called out in the bulleted lists within each entity block. These entity behaviors are implemented by the ensemble of simulation models depicted as shadowed blocks at the left and right margins of the figure.


Weather Generators (e.g., CLIMGEN)
USDA's SWAT Model (Soil & Water Assessment Tool)
New Livestock Models
Landscape
• Weather
• Surface Heat and Moisture Exchange
• Surface and Groundwater Hydrology
Field
• Vegetation Growth
• Nutrient Cycling
• Soil Evolution, Erosion
Herd, Flock
• Population Dynamics
• Foraging Behavior
Community
• Shared Resource Allocation (e.g., mushá)
Nomadic Community
• Exchanges with Settlements
• Migration Route Selection
Household
• Agricultural Practices
• Pastoral Practices
• Stress Coping Mechanisms
• Marriage
• Inheritance
• Exchange
• Evolution of Household Structure
Person
• Demographics
New Social Process Models
New Demo-graphics Models
Figure 4. Simplified view of ENKIMDU domain objects and simulation models.

The simulations for the Tell Beydar settlement simulation studies described later in this paper address natural processes (weather, crop growth, hydrology, soil evolution, population dynamics, etc.) and societal processes (farming and herding practices, kinship-driven behaviors, trade, etc.) interacting daily across simulation runs that span decades to centuries. Software objects representing the key components of the simulation domain are resolved and modeled at the level of individual persons and households, individual agricultural fields and individual herd animals. Each of the decision-making "agents" in the simulation domain – each person, household, or other organization – governs its own behavior in the simulations based on its own local rules and in response to its own perceptions, preferences, capabilities, and goals.

Figure 4 illustrates in very broad terms the linkages between ENKIMDU domain objects and simulation models that the DIAS framework supports. One of the nuances that such a high-level figure cannot adequately portray is the ability of an abstract behavior aspect of a domain object to be served by any of a set of alternative models that all address the abstract behavior but in different ways. A more detailed example may help to make this point clearer. Consider the natural (physiological) behavior of milk production, or lactation, for the animals in a flock or herd. One simple natural behavioral model for lactation can allow a Fauna object representing, say, a ewe, to lactate a fixed daily quantity of milk for a period of time after the birth of its lamb. However, a more complex lactation model could look at such factors as sheep diet, body mass, and overall health to determine the quantity of lactation each day. In both models, the outcome of the lactation process (say, milk produced today, in kilograms) would be a parameter that can be defined in the object. Figure 5 highlights this example with two different lactation models being implemented by a Fauna object. Both models can be declared in ENKIMDU, but the specific model that is chosen in a given situation will be the one that is most appropriate for a given simulation context, as specified by the user in defining the context for a simulation. Here, the process object acts as the interface between the domain entity and model objects, processing both the entity input data that drives the model and output data produced by the model. Ultimately, it is the output produced by the model that evolves the parameter states of the entity object.

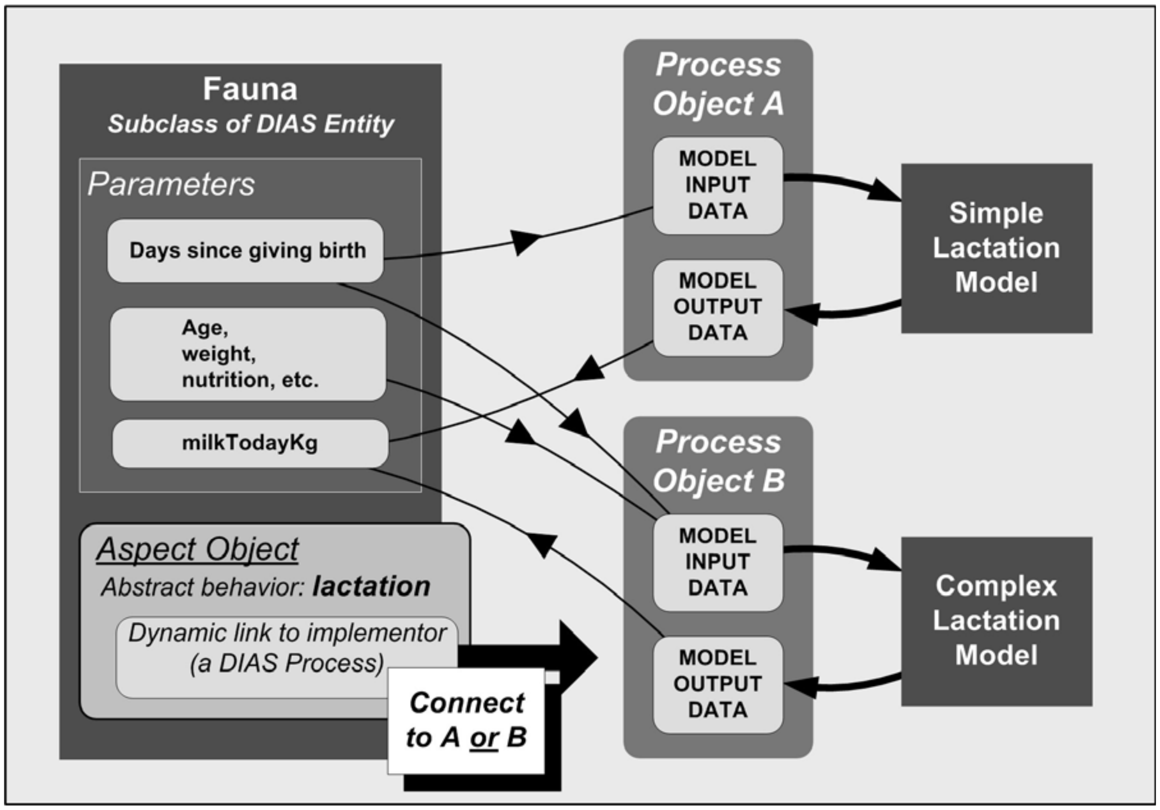


Figure 5. Example of alternative processes and models to address the same abstract aspect of behavior.

For complex real-world problem domains, the effects of various heterogeneous dynamic processes may manifest themselves in the domain at widely varying temporal scales and granularities. For example, “slow” processes, such as deterioration of a textiles in a warehouse or grain in a granary, may need to check in with the simulation only every month or so, to update the condition of the stored commodities, and possibly to recompute rates of deterioration based on slow changes in environmental conditions in the storage facilities. But “fast” processes, such as reciprocal exchange among agents at a local market, must update daily or even more frequently, since each market transaction can be completed relatively quickly and each transaction can substantially change an agent’s *subsequent* trading behavior.

The DIAS software infrastructure provides explicit support to allow entities to express their dynamic behaviors at the native temporal scale and tempo appropriate to each such behavior. First, since DIAS is a discrete event simulator, events delineating the tempo of change for each process can be arbitrarily spaced in time, and can even be spaced or bunched in time at irregular intervals if appropriate; there is no “native” simulation time step that forces process representations into tempos that may not suit them. A second, more subtle but more powerful means of support for temporal heterogeneity is provided by the DIAS “process trigger” mechanism. Each DIAS Process, representing a dynamic behavior of a type of simulation entity, can specify the preconditions that must be met in order for it to launch a model to express that behavior. These preconditions can include requirements that certain entity state variables of the owning entity, or of any other entities that are declared to be of interest to the owning entity in the context of this behavior, have timestamps indicating a specified degree of “freshness.” Thus, for example, a social model of daily irrigation practices might require freshly updated channel water levels and flow rates before it can launch. If these updates are in fact to be provided by a surface hydrology model, then the hydrology model is implicitly constrained to update before the irrigation practices model can run. Note that in this situation there is absolutely no need for either of the models to have any knowledge of the other one, because each model’s needs are always expressed purely in terms of domain object attributes. This mechanism makes it possible for differently paced heterogeneous processes to automatically cue each other to run in an appropriate temporal sequence, without any need for the researcher to specify or script such a sequence. This inherent ability to allow diverse process representations to self-organize to properly regulate the flow of information in DIAS-based simulations such as ENKIMDU has proven to be extremely valuable. It has meant that we are spared the chore of adding escalating volumes of procedural control code to our simulation engines whenever simulation complexity increases.

An Appendix has been included to show brief examples of the entity-aspect-process-model paradigm. The actual linkages between the agents and models can be inserted at run and build time. Appendix A.1 shows one technique that links the agent with a behavioral model in the instantiation of a simulation scenario, while other examples (Appendix A.2 and A.3) indicate what the entity, aspect, and process objects contain. An example model that is linked with the entity-aspect-process sequence is described in Appendix A.4, and will be discussed shortly.

In order to help facilitate simulation of interactions among different social agents, Argonne has augmented DIAS with a social simulation toolkit called FACET: the Framework for Addressing Cooperative Extended Transactions (Christiansen 2000b). FACET can be used to construct simulation models of transient or persistent social behavior patterns in which individuals and organizations interact over time. These patterns can be drawn from a variety of societal settings. For example, for various simulation applications we have built at Argonne, we have used FACET behavior pattern models to represent business practices, government and corporate policies, military doctrine and operating practices and tactics, and, as in the ENKIMDU examples we will present momentarily, complex, persistent pursuit of agricultural and pastoral practices by ancient Northern Mesopotamian households.

FACET is presently implemented as a separate Java package within the DIAS project; many of the main FACET modeling software objects are implemented as subclasses of high-level DIAS simulation infrastructure objects. Thus, for example, FACET models are implemented using the same DIAS Model and Process objects described earlier, and are each invoked by launching a behavior Aspect of a DIAS Entity object, just like any other DIAS model. FACET models that we have built to date have all been internal models, not requiring a Model Controller software wrapper, though they could be built as distributed, external models if that mode of interaction were needed. The FACET package could in principle be adapted to be used without DIAS; it would need an alternative discrete event posting and distribution facility, and surrogates for a few object superclasses presently provide by DIAS to be usable in stand-alone mode or from within other simulation software environments.

FACET-built models of social processes tend to behave quite a bit differently than, say, models of natural processes such as weather, subsurface water flow, etc., which may be following very highly repetitive patterns in evaluating continuous, smoothly varying differential equations for conservation of mass, etc. Social processes, both in real-world situations and as modeled via FACET, tend to exhibit different patterned behaviors: proceeding at different timescales, interruptible and redirectable by a host of exogenous influences, and often requiring some degree of cooperation and coordination among multiple simulation entities such as persons and organizations. FACET models capture complex societal interactions via a mutable directed graph of course-of-action steps, which are essentially behavioral sub-models. Each of these steps must generally occur in a specified locale, and require a finite amount of time, during which the participating agents are partially or fully preoccupied with the action in the step and are thus unavailable for other activities. FACET action steps typically require resources of one or more of the participants, and result in changes to the states of some of the participants, and, in the agricultural examples we will be discussing shortly, in changes to elements of the natural environment as well. Step participants, or agents, and their resources can be carried over from step to step, or can be different in various steps, depending on the nature of the social process being modeled. Specific agents fill roles that are called out in an abstractly in the step definition. For example, in some of our detailed health care simulations (Christiansen 2000b), different persons fill the roles of Patient, Nurse, Receptionist, Physician, and so on in FACET models of outpatient visits; the same person might take on the role of Physician in one model and Patient in another. The modeled social patterns thus present a series of templates, with slots for abstract participants filling the various specific roles called for in each action step.

In order for Agent objects to take active roles in FACET societal models, they must first implement Java interfaces that give each of them access to their own instances of the FACET Participant and ResourceManager object classes, to which the agents can then delegate much of the reasoning and ancillary processing regarding participation in social activities. For example, through the Participant delegate object, an agent can consult an agenda of requests that have been made for it to join various activities and make specified resources available in various capacities (i.e., roles), so the agent can prioritize the unsatisfied requests on its agenda and make appropriate responses. If an invited participant is unwilling to join in an action step, or unable to do so because some of the requested resources are not available, then the step will be delayed until the requisite level of cooperation is achieved. Timeout intervals can be specified so that planned activities can be cancelled if the needed participants and resources are not forthcoming; alternative behavior pathways can be specified in the FACET model to deal with such occurrences, as well as for any number of other exogenous events that might preempt, cancel or redirect a planned sequence of social activities. Appendix A.4 shows a code example of FACET model implementation for ENKIMDU. In order to simplify the example, the creation of only one of the model's action step is shown.

An example of the complex behavioral mechanisms that have been implemented using FACET models is provided by the collection of agricultural practice for dry-cropping of grain in the rain-fed northern region of Mesopotamia modeled in the ENKIMDU simulation system (Christiansen and Altaweel 2006). Figure 4 shows the directed graph of action steps for two ENKIMDU FACET agricultural practice models, the Field Crop Management Model and Dispose Harvest Model. Within ENKIMDU, a separate instance of the Field Crop Management Model is launched for each agricultural field being tended by each household in the current simulation year. Similarly, a separate instance of the Dispose Harvest Model is launched, for each household that has been tending grain crops in the current simulation year, as soon as its first grain sheaves are brought home to its threshing floor.

Throughout the time period covered by these two models, roughly nine months from September through May, each agricultural field must be maintained, through execution of the sequence of agricultural activities depicted in the model's steps, by human agents in order to produce a crop. These agents apply their own labor, employ their own assets, such as ard plows, along with other assets such as plow teams borrowed from the community in a cooperative arrangement, and expend their own consumables (e.g., seed), in the modeled steps. At the same time, ongoing environmental processes such as plant phenology, percolation of soil moisture, and daily weather fluctuations affect the state parameters of both field and crop. The current states of both field and crop are known to the tending household when its members visit the field; this knowledge is essential in the farmers' ability to synchronize their activities with the crop's own internal timetable. For example, the ENKIMDU crop management model incorporates the farmer's understanding that harvest can commence once the crop has reached a specific phenological stage, in this case somewhat senesced and defoliated, with grain ripeness assessed as it is in the USDA's SWAT crop model, in terms of the accumulated fraction of the nominal number of growing-degree days needed to ripen each variety of field crop.

We believe that it is essential to capture the interplay among field, crop, *and farmer* if a settlement system's agricultural activities are to be portrayed with any degree of fidelity, and particularly in pre-modern times, when the pervasive occurrence of critical bottlenecks and shortages of labor and materiel, and a farming household's ability to devise means to fend off such difficulties, could prove decisive in determining crop performance, and thus the sustainability of the household. We cannot stress strongly enough that crop performance is not a purely biological issue that can be decoupled from the social/natural system interactions that bring the crop into being and see it through to harvest. Even in our still relatively simple simulations, we have observed examples of this at every turn. Farmers in our simulations who have to wait too long for their turn to use community-managed plow teams may not get finished with that activity, and thus be able to move on to sowing of seed, until the winter rains are nearly over, putting the crop at a significant initial disadvantage from which it may not recover. Households that lose key workers at critical points in the agricultural year such as harvest time, due to deaths or simply to other unavoidable social commitments, may not get all of their harvest in before it withers in the mercilessly hot and dry Mesopotamian summer. In such cases, the crop weather and soil conditions may have been perfect, and the crop planted may have been vigorous and perfectly suited to local conditions, but the crop failed because the social component of the overall dynamic was not sufficient to the task at some critical juncture.

The ENKIMDU agricultural models for ancient Mesopotamia explicitly reflect Mesopotamian conditions. For instance, in both modern and ancient Mesopotamia, the harvest season is in the early summer, requiring crops to be harvested before they wilt under the increasing heat, lack of rain, and changing soil conditions during the harvest action step (Buringh 1960). Other labor processes associated with the crop cycle include threshing and storing grain, or post-harvest labor activities (Murray 2000). As noted above, these post-harvest behaviors were represented in a separate ENKIMDU FACET model, called the Dispose Harvest model, not directly linked to the crop model, but invoked through the participating entities (Figure 6).

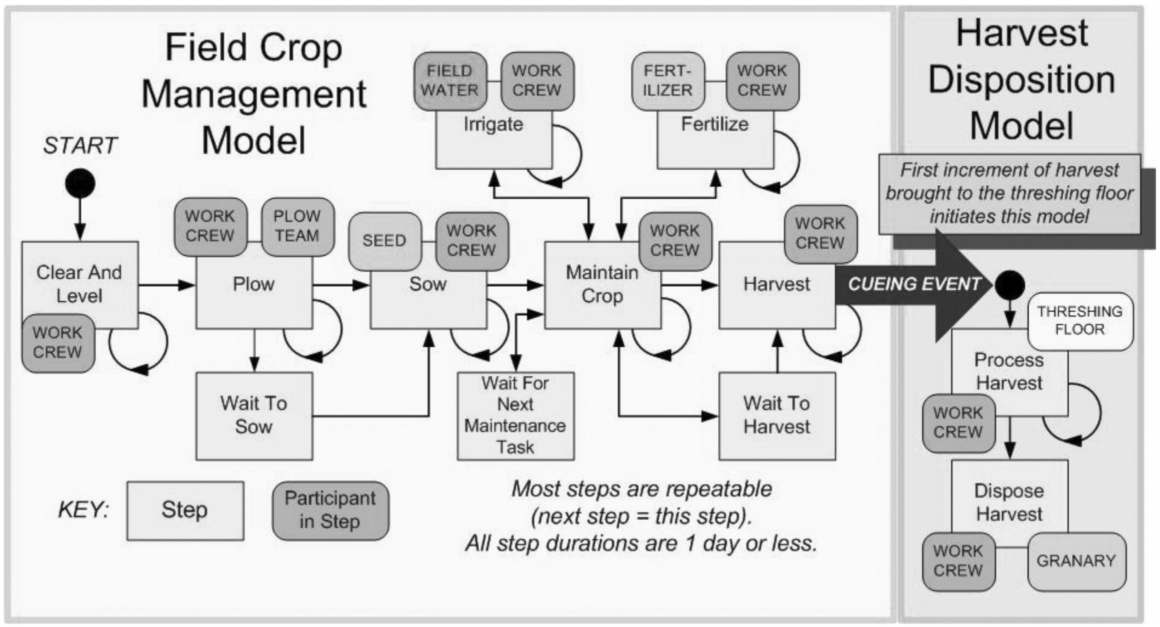


Figure 6. FACET social behavior pattern models for Mesopotamian dry farming practices.

The Manage Herd Local Model in ENKIMDU addresses a community’s cooperative livestock management activities, undertaken by groups of households that keep their sheep and goats within their own household compounds overnight, but pool their herds and flocks each day. This pooling allows the combined herds and flocks to be efficiently taken out each day to forage pastures that are near enough to allow them to return to the settlement before nightfall. The Manage Herd Local Model works similarly to the previous example. However, because the relevant behavior pattern must be repeated on a daily basis, the total timeline for performance of all of the model's action steps is far shorter: one day, compared to the crop models' nine-month run. A schematic diagram of the herding model's daily steps is shown in Figure 7. This model was largely based on an ethnographic study that describes a pastoral system that may have also existed in past Mesopotamian societies (Sweet 1974). As was the case for agricultural practices, simulated decisions regarding herding practices are dynamic and continuously evolving as circumstances change for each day. For instance, with the advent of the agricultural season, herds need to be moved to locations away from the cropped fields. Factors such as field biomass and distance from the settlement determine which areas these animals will be sent to graze by the herder agents. Livestock herds themselves can begin to affect where they graze by their trampling, foraging, and excrement deposition that influence plant growth on the fields they traverse and graze. The herding cooperative social agents continuously monitor local pasturage conditions to help decide where to forage in subsequent time. Also, because each animal in the herd is evolving its own physiological parameters over time in the simulations, undergoing specific processes such as gestation, lactation, etc., and responding to population dynamics (e.g. birth, death) events (Redding 1981; Blaxter 1967), the herd's state parameter values also evolve. This affects the specific impacts the herd has on the pastures and other landscape partitions that it traverses in its daily foraging activities.

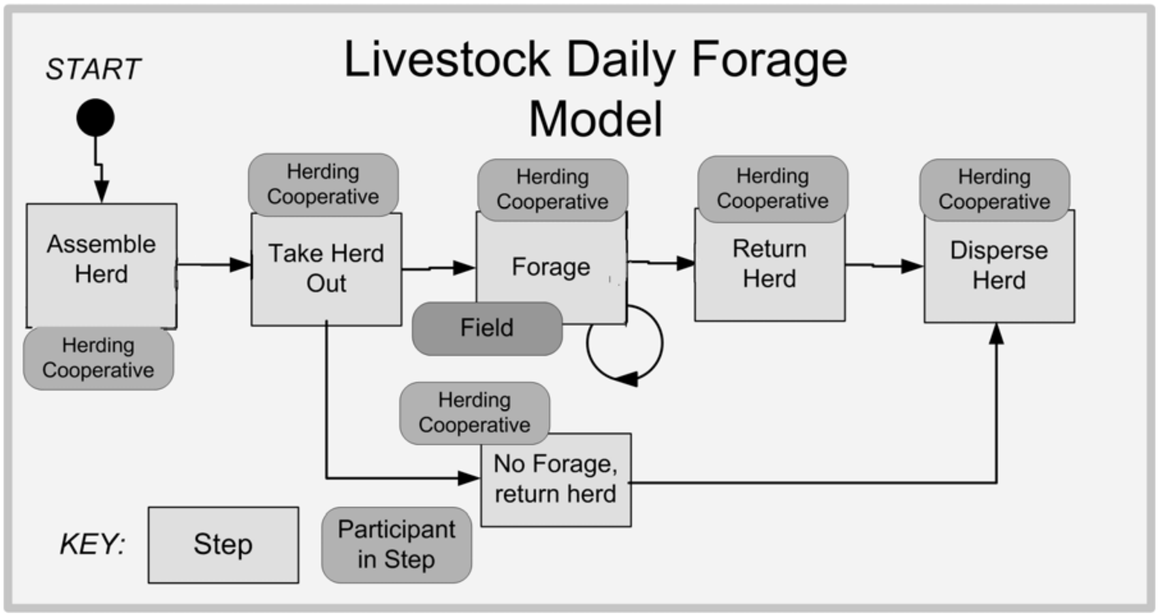


Figure 7. A FACET social behavior pattern model for Mesopotamian pastoral practice.

Certainly these FACET model examples from ENKIMDU represent a fairly limited perception of how agropastoral activities may have been practiced in ancient Mesopotamia. Thus, it would be worthwhile to see variations of these models as well as entirely different agropastoral models incorporated as alternatives. Using the DIAS approach outlined earlier, such alternative formulations will generally be mapped to the same specific domain entity types and abstract behaviors as the original models, and can simply be selected and swapped in at runtime, so there would be no need to explicitly account for every possible combination of model alternatives in the ENKIMDU code.

4.0 Incorporation of Landscape Dynamics and External Models

All societies operate within landscape contexts, making it essential to incorporate landscape environmental dynamics when investigating social mechanisms relevant for a particular society. Furthermore, agents should in general possess the ability to sense, experience, and reason upon evolving conditions in their own local environments over the course of a simulation, if they are to be able to behave in a realistic manner.

Given that the creation of highly complex natural landscape models can be a major project in its own right, above and beyond the effort required to construct a suite of social-behavioral models, simulation platforms that aim to address a very wide range of natural and social processes can benefit from the capability to access relevant, capable external modeling systems, even legacy or older models built in different computer languages. DIAS incorporates such capabilities by building upon Java code interfaces and adding controls that allow a platform such as ENKIMDU to communicate effectively with other modeling and simulation software packages. For example, in ENKIMDU we have incorporated the Soil and Water Assessment Tool (SWAT) modeling suite developed for the United States Department of Agriculture (USDA) to study managed natural landscape dynamics for agricultural, ecological, and pollution control purposes (Arnold et al.1998). Many critical landscape parameters such as soil moisture and plant phenology are evolved through processes incorporated within SWAT (Neitsch et al. 2002). Key dynamic processes that SWAT simulates include the following:

Processes Simulated by SWAT

Hydrology:

- Surface runoff, ponding
- Drainage routing at individual field to watershed scale
- Percolation, water table dynamics
- Lateral subsurface flow
- Evaporation and snowmelt,
- Soil temperature dynamics
- Precipitation recharge

Meteorology: Daily agricultural weather, from daily observations or via a Markov Chain stochastic weather generator that is driven by data tables of climatological means and extremes.

Soil Evolution: Water and wind erosion, deflation, etc.

Nutrient Cycling Dynamics: Nitrogen and phosphorus balance

Vegetation Growth:

- Canopy and root development
- Evapotranspiration
- Water budget
- Nutrient uptake
- Growth constraints (by moisture, temperature, nutrient levels, etc.)
- Yield, with constraints as above.
- Dormancy
- Effects of pests, effects of grazing

Human Interventions:

- Tillage effects (leveling, plowing, planting, harrowing, harvesting, etc.)
- Irrigation
- Fertilizer and pesticide application (presently no pesticide effects on vegetation modeled in SWAT, only routing and fate of pesticide chemicals)

In DIAS terms, the SWAT model, via a DIAS Process proxy object, reports to the simulation that it is capable of implementing the high-level “evolve landscape” aspect of behavior called out in the definition of the ENKIMDU Landscape object class. A remote SWAT model execution is then linked at run-time with an ENKIMDU Landscape object, via the latter’s “evolve landscape” behavior aspect. For this SWAT execution, the SWAT Fortran code is running on a processor separate from the processor that holds the ENKIMDU domain object representation, including the Landscape object. A DIAS Model Controller wrapper written in Java, with some C for direct access to SWAT’s large internal common data block, facilitates two-way communication between SWAT and the domain object representation. Daily updates to soil and vegetation parameters in every partition of the landscape (including all crop fields) flows form SWAT to the domain objects, and information on tillage operations and other agricultural and pastoral interventions visited on each field by household work crews, herds, and flocks flow in the opposite direction; process feedbacks are thus automatically provided for. More details on this linkage are given below and in the Appendices.

In ENKIMDU the social agents (households, in this case) can perceive and respond to changes to the landscape that result from the operation of the SWAT. For instance, while the Field Crop Management model is functioning, human agents are concurrently able to perceive changing soil and plant properties due to the dynamic behaviors of SWAT that evolve the agricultural field agents. As noted earlier, for instance, households’ agricultural work crews can use observed maturity levels of a given crop to determine proper time to begin the harvest. The maturity state of a crop not only evolves through natural conditions, such as moisture and soil inputs, but also by anthropogenic processes that impact SWAT’s calculations. An example of this is a crop field that is being harrowed: this tillage operation will remix the top layer or so of soil, changing vertical nutrient distributions, and altering subsequent plant growth trajectories. Thus, SWAT is simply treated as another process model that evolves different parameter states, and is able to provide feedback necessary for human agents to determine their next social actions. In the current ENKIMDU prototype, SWAT is accessed through a DIAS

Model Controller object that communicates with it using a distributed Java RMI interface, allowing SWAT to physically reside in a different computer connected through a network (see Reilly and Reilly 2002 on distributed computing). Using the Java Native Interface (JNI) provided within the Java language, ENKIMDU can then access the variables incorporated and produced within the SWAT model (written in Fortran90). Appendix A.5 provides example code to show how SWAT can be incorporated within ENKIMDU using Java Remote Method Invocation (RMI) tools as well as those built by Argonne.

Landscape dynamics are driven to a great extent by weather and climate. ENKIMDU can accept daily weather updates from actual observations, from numerical weather model output or synthesized from monthly local climate summaries using a Markov process weather generator. For some ENKIMDU studies, we have chosen to use a weather generator called ClimGen in place of SWAT's internal weather generator, because it proved somewhat easier to manipulate in generating varied results (Stöckle et al. 1999). Climate outputs produced in ClimGen can be streamed into SWAT, giving ENKIMDU access to these data. As ENKIMDU development proceeds, and particularly as we simulate larger geographic regions, we plan to provide the domain object model with access to the MM5 (Anthes and Warner 1978) mesoscale weather model and/or other comparable numerical weather models to provide fine-scale spatially varied climates, using derived 3rd millennium BC surface cover estimates. Long-run general circulation model (GCM) paleoclimate analyses (e.g., Joussame and Taylor 2000; Kutzbach et al. 1996) can be used to help provide boundary conditions for the mesoscale climate simulations. Monthly average GCM results can guide selection of representative finer-scale climate analog datasets to provide the requisite boundary conditions for MM5. Alternatively, if GCM paleoclimate results are available at sufficiently fine temporal resolution, they can be used directly to drive the MM5 runs.

Regardless of the specific suite of models and behaviors incorporated within a simulation, the ability to incorporate various modeling suites and mechanisms created by different research and development efforts, including both physical and social behavior representations, is essential for simulation platforms to adequately address complex social and environmental questions. Using the DIAS framework to wield this capability enables us to extend ENKIMDU to address the scope and range of modeling coverage for natural and social processes without expending a lot of effort in creating new models, when existing models may provide the functionality and domain expertise that is needed. A major strength of a given simulation platform can, therefore, be determined by how well it integrates external models, or models built by others, into an overall effort.

DIAS can also be used to integrate existing software tools that can perform as standalone applications. In our current simulation efforts, one such tool that has been used to supply spatial display and analysis is the GeoViewer (Lurie, Sydelko, and Taxon 2002), an Argonne developed geospatial toolkit that includes typical geographic information system (GIS) capabilities such as flexible modes of data input and spatial statistical functions. DIAS can easily interface with this tool; several of the Argonne object-based spatial utility libraries are used both by DIAS and by the GeoViewer. The

GeoViewer can also be linked to external models and coupled directly with other research software frameworks comparable to ENKIMDU to perform spatial data transfer, display and analysis tasks across variable spatial scales. Other spatial tools can be, and have been, linked to DIAS applications. However, the commonalities between the software architectures of DIAS and the GeoViewer make it particularly straightforward for us to access and spatially display attributes and behaviors of entities in our simulation domain object models. The GeoViewer tool has been coupled with ENKIMDU, and is used to display changing values of various landscape attributes, with particular emphasis placed upon evolving agricultural field and crop and forage properties, as well as to show movement of agents and of fauna across the landscape. Figure 8 presents a snapshot of an actual ENKIMDU GeoViewer animation display for a simulated Bronze Age settlement, with field and vegetation states and minute-by-minute locations and activities of all household work crews in the settlement.

5.0 Baseline Simulations for a Northern Mesopotamian Settlement

Up to this point, we have discussed in general terms how ENKIMDU works, emphasizing how it makes use of our generic DIAS and FACET simulation frameworks, and have provided some schematic and descriptive examples of different behaviors within the simulation framework. We would now like to briefly describe some of the most fundamental early ENKIMDU modeling runs, our “baseline” cases, for a single simulated northern Mesopotamian settlement. Given that other early modeling results of our ancient Mesopotamian simulation project for various interesting stress scenarios are beginning to appear in various other publications, we suggest that interested readers can access these published or soon-to-be-published resources to further explore our simulation results of fine-scale, heterogeneous social and natural system process interactions (Christiansen and Altaweel 2006; Wilkinson et al. forthcoming). These articles detail a great many of the specific behaviors and input aspects associated with actual social and natural systems in Bronze Age Mesopotamia and the Near East, and discuss how they have been represented within ENKIMDU’s dynamic domain object model. For example, these papers identify the relevant types of households in our simulations and describe their evolution over time in the ancient Near East, discuss ancient textual data used to construct Bronze Age behaviors, and provide detail on the spectrum of specific household stress coping strategies that have been modeled in the ENKIMDU framework.

For our initial simulations of ancient Mesopotamian settlement systems, we have chosen the landscape around Tell Beydar, an ancient settlement site located in the Khabur Basin of northern Mesopotamia (modern northern Syria). The simulated landscape characteristics were derived from site surveys and from satellite CORONA imagery from the 1960s. The region consists of rolling plains and low-lying basalt plateaus. Several wadi systems, or temporary riverbeds, can be found in the region near Tell Beydar (Wilkinson 2000). Input data associated with the surrounding landscape is captured in ArcGIS shapefiles and incorporated into ENKIMDU to create the data parameters for landscape entities. Data integrated into ENKIMDU’s landscape include soil types derived from neighboring regions near Tell Beydar (van Liere 2003). These site region characterization data are briefly summarized in Figure 8.

Tell Beydar Field Layout:

Surrounding the Tell Beydar settlement site to a radius of roughly 2 km:

337 fields averaging about 3 ha each
→ 1,000 ha total

Fields excluded from basalt highlands to west and from courses of major wadis

Soils: calciorthid silty loam; alluvial in wadis; thinly covered basalt formations to west.

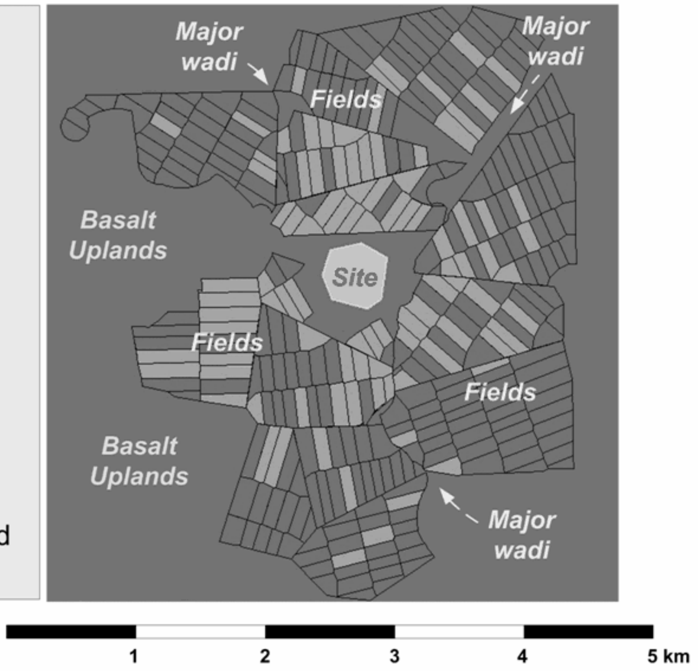


Figure 8. Spatial layout of the Tell Beydar settlement simulation region. Field colors indicate variations in vegetation cover in the settlement's agricultural fields.

The underlying assumptions and initial conditions for our "baseline" simulations for the synthetic settlement that we have based on the Tell Beydar site refer to "normal" conditions that do not deviate dramatically from expected means, and do not include any strong exogenous shocks to the coupled social/natural systems that we are modeling. For example, modeled demographic rates are not significantly altered from what are considered to be likely average rates of birth and death in the region during the Bronze Age. We did not impose any major climatic events that significantly deviated from mean values for sustained periods. However, even within a "normal" range, variability could be substantial from year to year, as in the case of modeled local annual total precipitation values ranging between 100 and 600 mm per year over a 100-year scenario (Christiansen and Altaweel 2006).

Our baseline simulations were run for 100 years of simulated time. As we have discussed, social activities and environmental process dynamics for these runs were represented at very fine spatial and temporal scales over these multigenerational runs. To highlight this point, Figure 9 provides a snapshot from an ENKIMDU user interface animation view for one of our 100-year baseline runs, showing the field and vegetation states that are updated daily and the minute-by-minute locations and activities of all household work crews in the settlement.

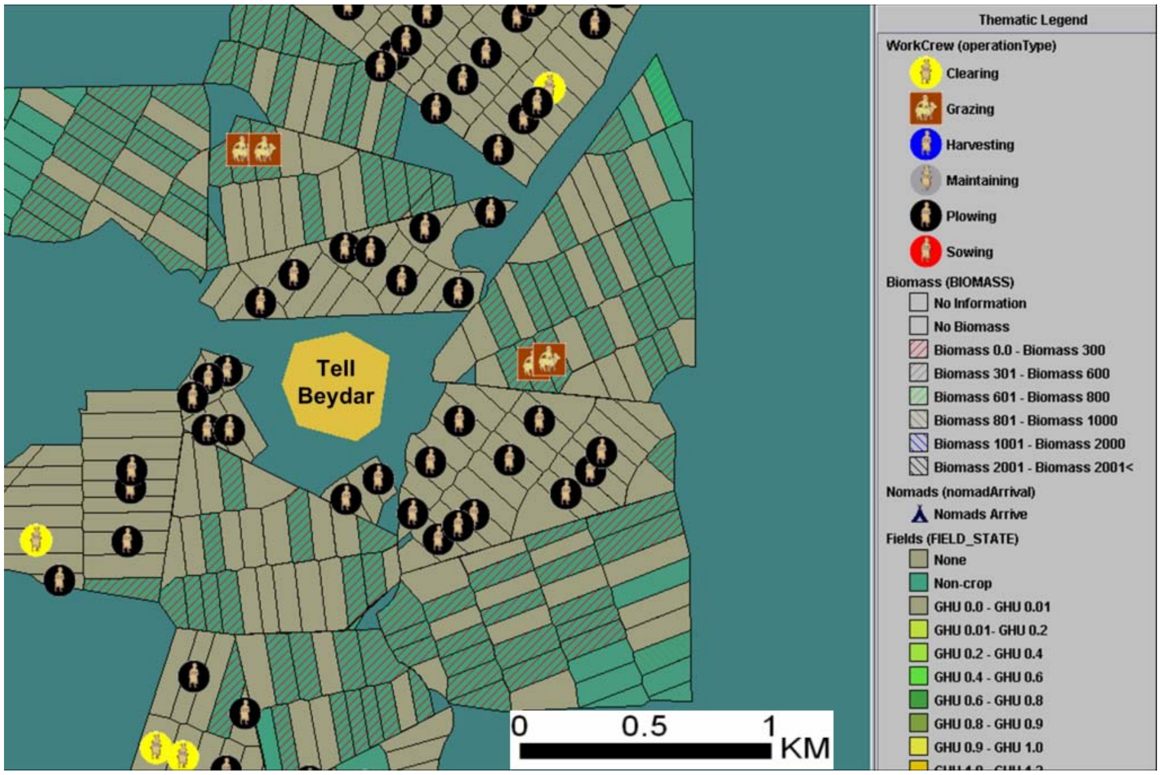


Figure 9. Snapshot from an ENKIMDU animation display, showing crop fields and vegetation states and household agricultural and pastoral activities.

In Figure 9, the phenological state and standing biomass density of field vegetation are encoded using background color and colored diagonal lines respectively. Note that the settlement's four local cooperatively tended herds shown in Figure 9 are foraging fallow fields chosen by their respective herding cooperatives: fields that are the nearest ones to the settlement with sufficient biomass to sustain today's grazing and browsing; tomorrow the chosen forage fields may be different. Work crew activities are indicated by color-coded icons in the figure; on the autumn day depicted, all crews other than those of the herding cooperatives were either clearing fields or had finished clearing and were engaged in plowing. It is illuminating to note that the households that appear to be "behind in their work," the ones that are clearing their fields of weeds and brush and are not yet plowing, are working fields that tend to be quite distant from the settlement center. Our simulated settlement is assumed to be *nucleated*; all its households' dwellings are concentrated at the center of the crop field mosaic. Thus the crews working the more distant fields are walking further to and from their tasks, giving them less time each day to devote to their actual field tasks, and thus putting their households under slightly greater sustainability stress. Because of the contextual richness of the fine-scale simulated social and environmental settings in our ENKIMDU simulations, we have found that such nuances appear quite frequently in the modeling runs, and, furthermore that, following the chain of subsequent events, they can in some cases have sustainability impacts far greater than one might reasonably have expected.

Figure 10 shows the variations in settlement population and number of active households during the 100-year baseline simulation scenario. In this figure, it can be seen that the initial population of around 500 people grew during the first 50 years, declined, and then appeared to stabilize near 600 toward the end of the scenario. The population losses in the second half of the scenario can largely be explained by higher overall emigration, with 142 emigrants in the first 50 years and 247 emigrants in the second 50 years. In the baseline runs, emigration was set up to be a household food stress coping mechanism of last resort, when the expedients of growing grain crops, obtaining grain gifts from kin, selling off or eating livestock, or borrowing grain at interest were unable to keep starvation at bay. Given that agriculture was the main source of food for all households in the simulation, a shortage of farm labor to farm, or an inability to obtain food from other agents, particularly in the form of nonreciprocal food gifts from kin members, could be disastrous for a household (Wilkinson et al. forthcoming).

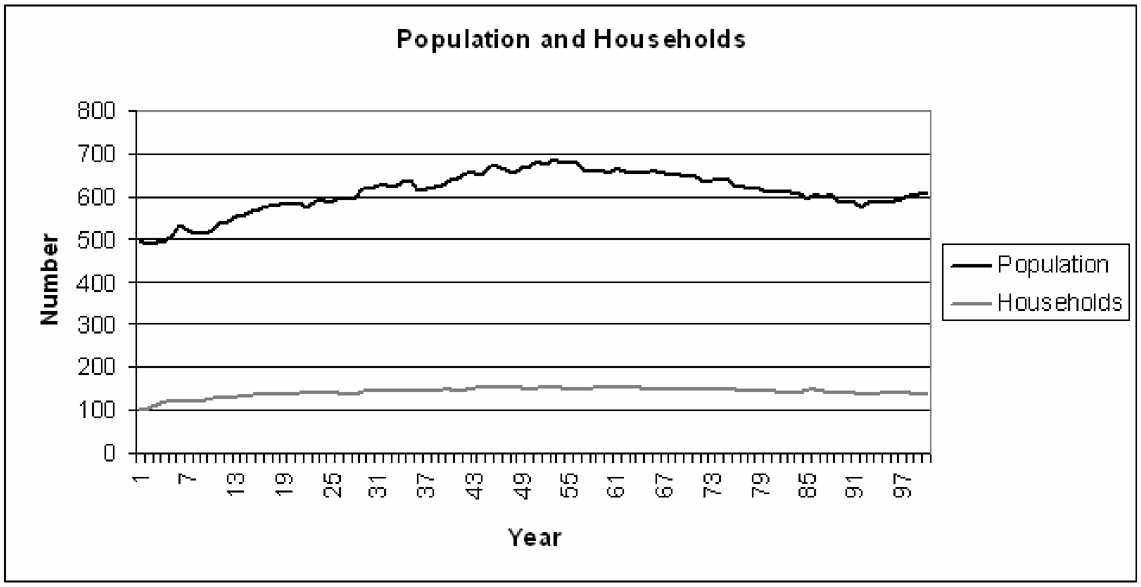


Figure 10. Graph showing population and number of active households in the virtual settlement for a 100-year "baseline" simulation.

One ENKIMDU simulation output parameter that can indicate overall community-level food stress is the net deficit measure for food requirements by members of the settlement. This measure records food shortfalls for households that fail to obtain sufficient food reserves at any point in a given year. The calculation converts all food available for a household into equivalent kilograms of grain, and then subtracts kilograms needed from kilograms available. This conversion is done because food can be obtained from not only agricultural production, but also from livestock, hunting, dairy, and the collection of wild plants. Converting all food quantities to equivalent grain kilograms offers a way to standardize the overall measurement. Therefore, if a household slaughtered a sheep, the meat produced would be converted into equivalent kilograms of barley grain.

The deficit grain graph in Figure 11 indicates that food deficits for some households did increase after the beginning of the simulation; however, this deficit did not steadily increase, as one might expect for a population that has greater emigration rates in the second 50 years of a 100-year scenario. Rather, a major factor that caused households and people to emigrate was their loss of links in their network of kinship connections over time. For instance, the overall number of kinship connections in the community for Year 35 was 349 (2.4 close kin households per household), while there were only 206 connections in Year 95 (1.5 kin close households per household). With fewer kinship connections, even a small food crisis could cause individuals to emigrate, since there would be fewer bilateral kin members on hand to help struggling individuals. Therefore, greater overall community food stress as such does not have to be present in cases of higher emigration and lower community sustainability, if there is a problem distributing the available food among households.

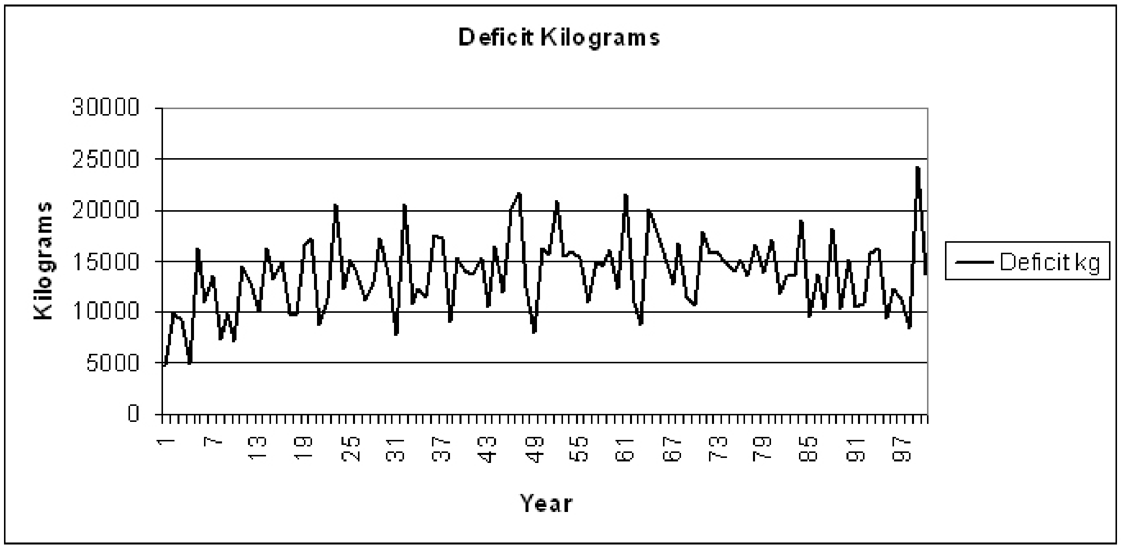


Figure 11. Graph showing deficit kilograms of grain (or food converted into grain kilograms) for the overall simulated settlement.

A factor that would certainly be expected to have an effect on the simulated settlement's overall sustainability is the agricultural yield of the surrounding crop fields. Figure 12 shows the overall average barley yield for our simulated settlement, along with the local annual mean precipitation, for the 100-year baseline scenario.

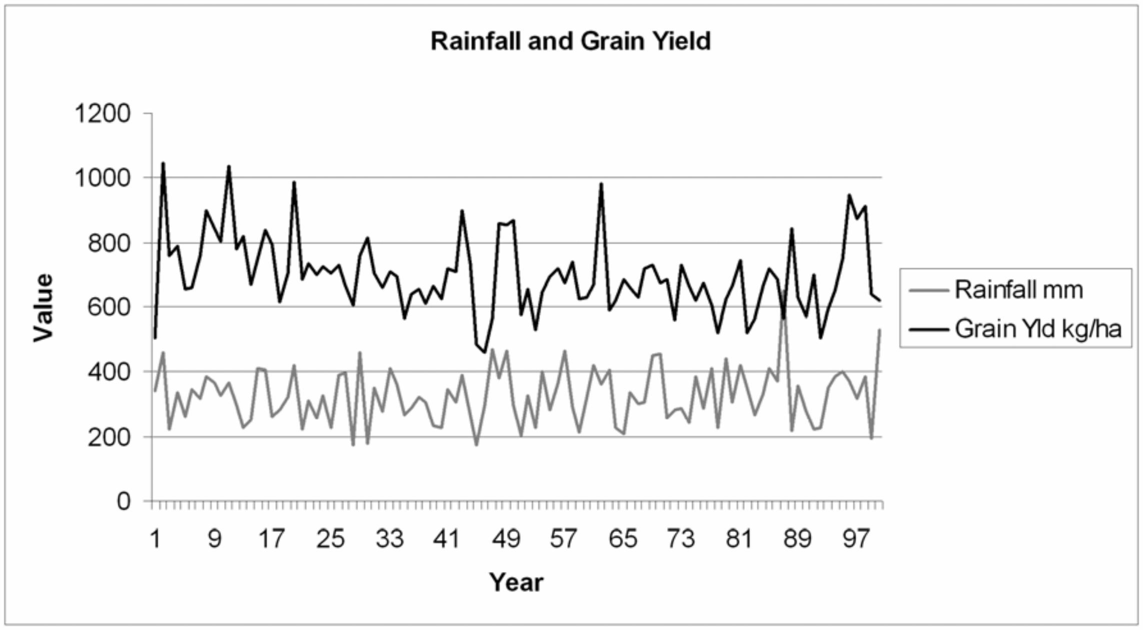
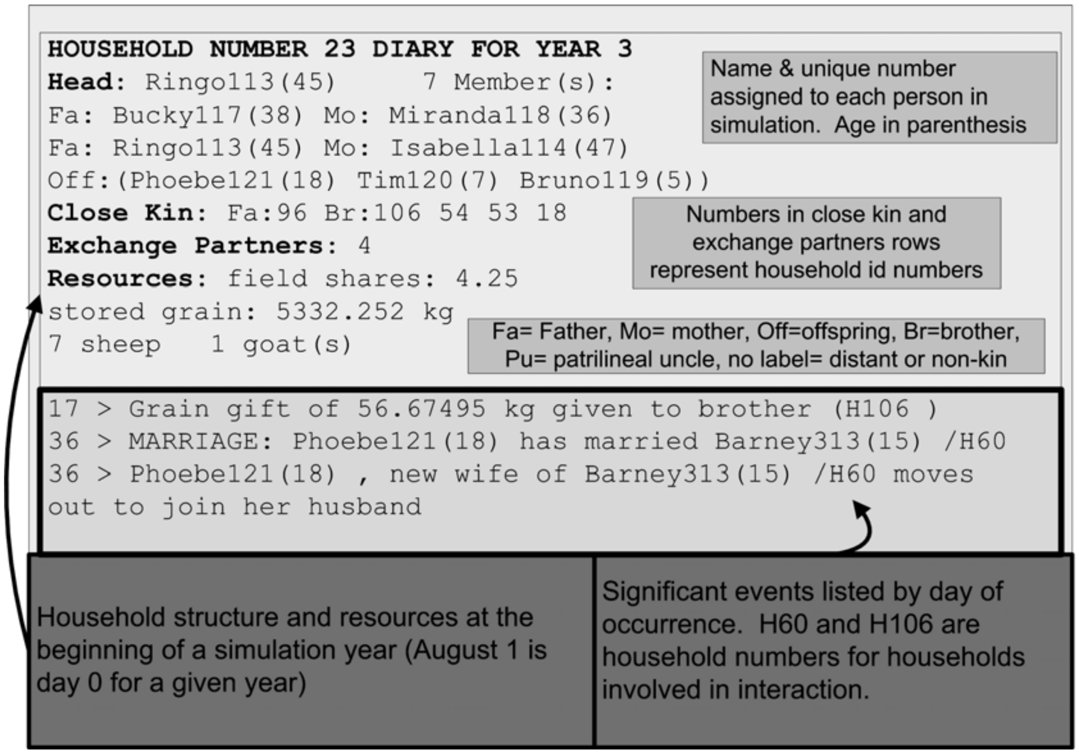


Figure 12. Overall settlement barley yield and annual mean precipitation for 100-year baseline simulation.

Two features of the graphs in Figure 12 are worth mentioning. First, the crop yield does seem to be positively correlated with the rainfall, after a slight temporal lag, as might reasonably be expected. Second, there appears to be a trend towards a slow net decline in overall yields throughout the simulation period. This cause of this apparent trend bears further investigation. It may be significant that, although all of the households in the settlement are scrupulously observing the traditional practice of biennial fallowing, no supplementary fertilizer (e.g., manure) is being systematically added to the fields in the baseline case, so nutrients may simply be being depleted by several generations of farmers planting every other year.

Although the example above can provide useful information as to why some individuals and households are forced to emigrate the community, a more detailed analysis of agents is often required to fully understand social phenomena. Looking at the results from the most fundamental agent level at any period of a simulation can help express details that are masked by the aggregate results. Therefore, ideally, a simulation needs to be able to report in detail the behaviors of its fundamental components, often over small increments of time. An example illustrating ENKIMDU's ability to provide informative fine-scale household and person level results will be given below.

An important functional element in ENKIMDU is the ability to produce "household diaries" that detail household and person behaviors over time. At present, these diaries record all significant demographic (birth, deaths, marriages, etc.) and resource-related events such as reciprocal exchanges, gifts, and loans in a given year. Figure 13 provides an example key to explain how the diaries can be used in interpreting various events and household/person situations.



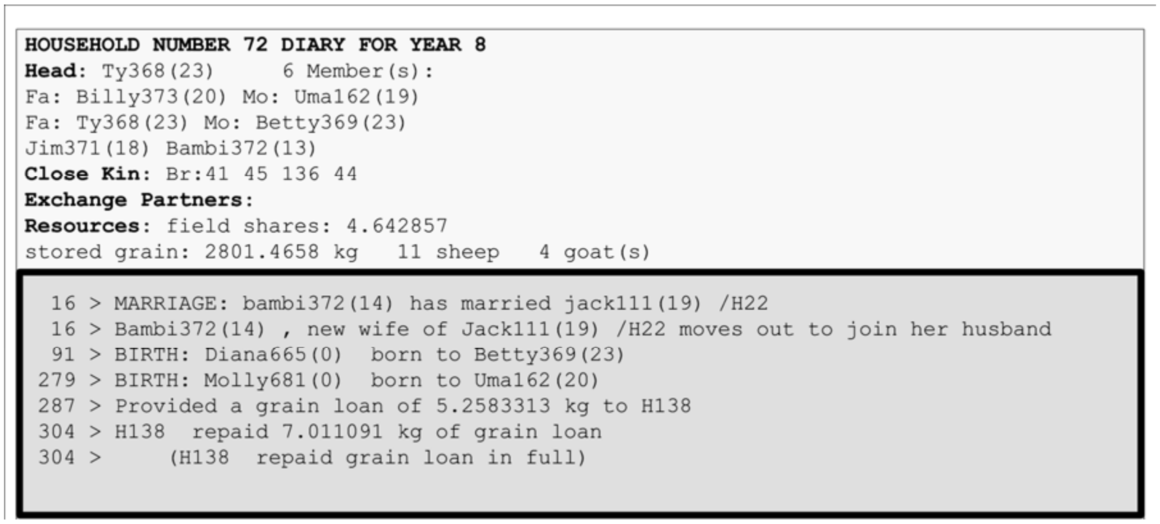
```
HOUSEHOLD NUMBER 23 DIARY FOR YEAR 3
Head: Ringo113(45)     7 Member(s):
Fa: Bucky117(38) Mo: Miranda118(36)
Fa: Ringo113(45) Mo: Isabella114(47)
Off:(Phoebe121(18) Tim120(7) Bruno119(5))
Close Kin: Fa:96 Br:106 54 53 18
Exchange Partners: 4
Resources: field shares: 4.25
stored grain: 5332.252 kg
7 sheep   1 goat(s)

17 > Grain gift of 56.67495 kg given to brother (H106 )
36 > MARRIAGE: Phoebe121(18) has married Barney313(15) /H60
36 > Phoebe121(18) , new wife of Barney313(15) /H60 moves
out to join her husband
```

Name & unique number assigned to each person in simulation. Age in parenthesis
Numbers in close kin and exchange partners rows represent household id numbers
Fa= Father, Mo= mother, Off=offspring, Br=brother, Pu= patrilineal uncle, no label= distant or non-kin
Household structure and resources at the beginning of a simulation year (August 1 is day 0 for a given year)
Significant events listed by day of occurrence. H60 and H106 are household numbers for households involved in interaction.

Figure 13. Example key explaining elements of household diary output. Each household has knowledge of its resources (e.g., grain and livestock), kinship networks, and keeps a memory of other households it has interacted with. Person names in the simulation were selected randomly from a list of modern names, and do not reflect ancient Mesopotamian names.

For this example, we begin in Year 8 of a baseline scenario similar to the settlement level example presented earlier (Figure 14). Household 72 can be seen to have six members in a multiple-family type household, a type common in the ancient Near East (Bagnall and Frier 1994). Two births can be observed in Year 8, with one member of the household marrying and moving to her new husband's household. This patrilocal event is an expected event for past Near Eastern societies (Schloen 2001). Looking at the grain reserves for the household, it appears that the members did not have any problem obtaining sufficient food, and the household was even able to provide a small grain loan to another household in the community. In addition, Household 72 had a reasonable number of livestock (15 sheep and goats) that could have been used for food stress relief.

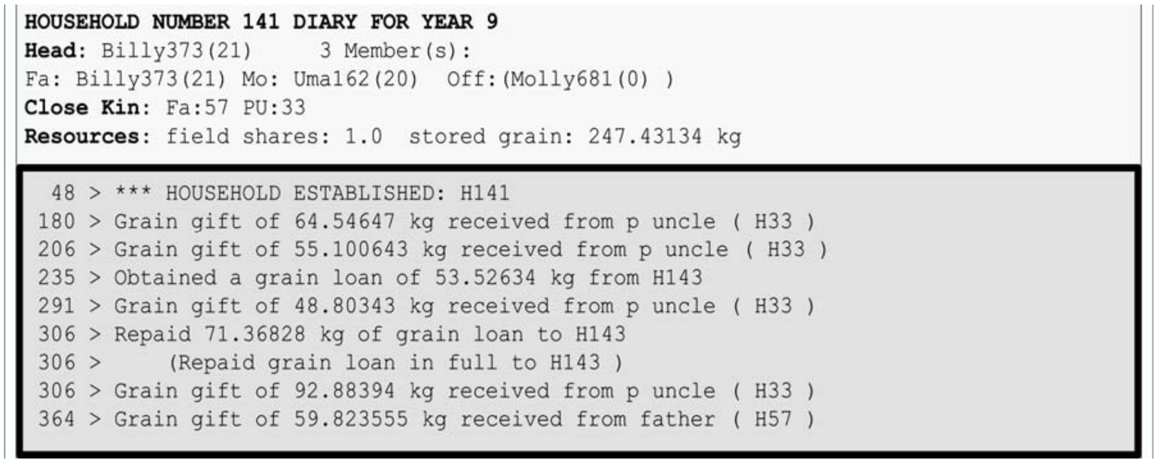


```
HOUSEHOLD NUMBER 72 DIARY FOR YEAR 8
Head: Ty368(23)     6 Member(s):
Fa: Billy373(20) Mo: Uma162(19)
Fa: Ty368(23) Mo: Betty369(23)
Jim371(18) Bambi372(13)
Close Kin: Br:41 45 136 44
Exchange Partners:
Resources: field shares: 4.642857
stored grain: 2801.4658 kg   11 sheep   4 goat(s)

 16 > MARRIAGE: bambi372(14) has married jack111(19) /H22
 16 > Bambi372(14) , new wife of Jack111(19) /H22 moves out to join her husband
 91 > BIRTH: Diana665(0)  born to Betty369(23)
279 > BIRTH: Molly681(0)  born to Uma162(20)
287 > Provided a grain loan of 5.2583313 kg to H138
304 > H138  repaid 7.011091 kg of grain loan
304 >     (H138  repaid grain loan in full)
```

Figure 14. Diary Year 8 for Household 72.

By the next year, Billy373, his wife, and child depart Household 72 to form a new nuclear family household (Figure 15). In this particular simulation scenario, large households can be highly productive by having more available members for labor tasks. Nevertheless, households with several families can also be more apt to fissure due to high social stress between the members, as this example shows. In any case, by creating a new household without access to many of the resources that belonged to their previous household, Billy373's family faced, in a relatively short period, some food stress. This food stress was not necessarily disastrous, since the household was able to depend on kinship bonds for one-way gift transfers. In day 235, the household did have to seek a grain loan from a non-kin household, a behavioral option that was much less favorable for Household 141. Certainly the household attempted to request grain gifts from its kin members, but at the time of the food request none of the kin members were able to provide assistance, forcing Household 141 to seek a grain loan from other community members. This loan, nevertheless, was not overly burdensome for the household, and it was able to repay this grain loan with interest by day 306, not too long after the harvest.

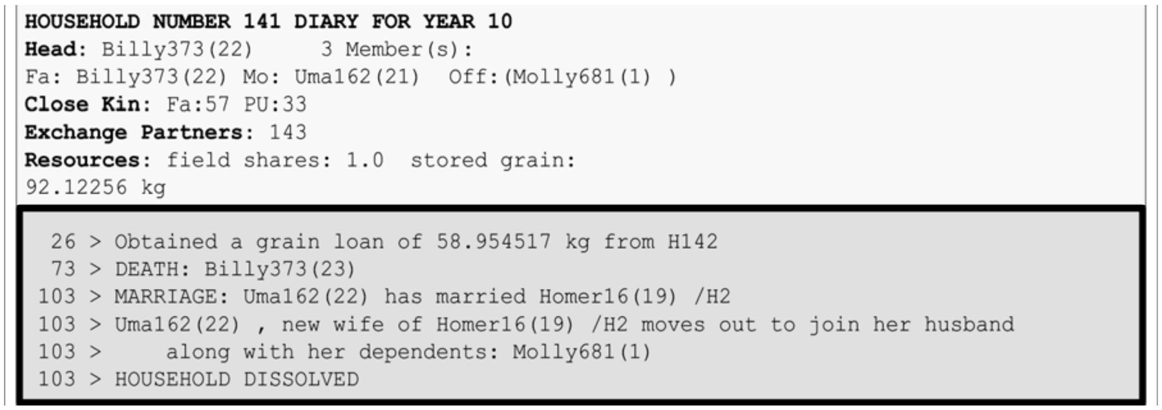


```
HOUSEHOLD NUMBER 141 DIARY FOR YEAR 9
Head: Billy373(21)     3 Member(s):
Fa: Billy373(21) Mo: Uma162(20)  Off:(Molly681(0) )
Close Kin: Fa:57 PU:33
Resources: field shares: 1.0  stored grain: 247.43134 kg

 48 > *** HOUSEHOLD ESTABLISHED: H141
180 > Grain gift of 64.54647 kg received from p uncle ( H33 )
206 > Grain gift of 55.100643 kg received from p uncle ( H33 )
235 > Obtained a grain loan of 53.52634 kg from H143
291 > Grain gift of 48.80343 kg received from p uncle ( H33 )
306 > Repaid 71.36828 kg of grain loan to H143
306 >     (Repaid grain loan in full to H143 )
306 > Grain gift of 92.88394 kg received from p uncle ( H33 )
364 > Grain gift of 59.823555 kg received from father ( H57 )
```

Figure 15. Diary Year 9 for Household 141.

The pattern of borrowing grain that began in Year 9 continued into the following year (Figure 16). At the beginning of Year 10, the household was in worse shape than it was in the beginning of Year 9 (see grain reserves), partly because it gave a portion of the harvest away to pay the previous year's grain loan. With the presence of kin members, however, the household still had potential to be sustainable by receiving labor assistance in farming tasks, a behavioral option enabled in this simulation, and grain gifts. Ultimately, what was most disastrous was the death of Billy373. The loss of an important labor resource reduced the ability of the household to produce sufficient crop yields, at least without labor assistance from other households. Significant labor assistance, however, can be financially costly and difficult to obtain in certain years. As a result, Uma162, the young widow of Billy373, and her child were in a dire situation. Faced with few options for household sustainability, Uma162's situation propelled her to choose to marry into another household, taking her child with her and ending the history of Household 141.


```
HOUSEHOLD NUMBER 141 DIARY FOR YEAR 10
Head: Billy373(22)     3 Member(s):
Fa: Billy373(22) Mo: Uma162(21)  Off:(Molly681(1) )
Close Kin: Fa:57 PU:33
Exchange Partners: 143
Resources: field shares: 1.0  stored grain:
92.12256 kg

  26 > Obtained a grain loan of 58.954517 kg from H142
  73 > DEATH: Billy373(23)
 103 > MARRIAGE: Uma162(22) has married Homer16(19) /H2
 103 > Uma162(22) , new wife of Homer16(19) /H2 moves out to join her husband
 103 >     along with her dependents: Molly681(1)
 103 > HOUSEHOLD DISSOLVED
```
Figure 16. Diary Year 10 for Household 141.

Although these brief results are based on a virtual community in a simulation, the processes determining the results were built using past data from the Near East. The household and individual behaviors as well as the demographic circumstances derive from historical examples, from the Bronze Age and other periods, indicating the realities some past societies faced under conditions of food stress (Christiansen and Altaweel 2006; Wilkinson et al. forthcoming). The cognitive powers given to the agents allowed them to evaluate different behavioral options before choosing what they believed to be their best option, such as seeking a one-way gift exchange from a kin member rather than borrowing grain from unrelated agents. What this shows is that ENKIMDU can allow agents to perceive and respond to their simulated social and natural surroundings from their own perspectives.

Certainly how agents make cognitive choices in past and present social systems is debatable. This debate even extends to computer scientists who have varying approaches in recreating systems that simulate human cognition and agency (Kauffman 1994; Plotkin 1993; Carley, Prietula, and Lin 1998). In the example presented, Billy373 made a choice of breaking away from the common Mesopotamian practice of staying in multiple-family households. In this case, human agency is not constrained by a given social structure common to a particular society. As a result, Billy373 did not behave as an automaton, and was able to make a choice that enabled non-normative results to occur, even tough some of his other behaviors conformed to common Mesopotamian practices. For the purposes of this paper, no particular socio-behavioral theory is strenuously argued; the examples simply provided a way of showing that different theoretical perspectives can be incorporated into a simulation scenario that allows archaeologists and anthropologists to test ideas and beliefs they may have. The end results of such testing can be possible insights into a particular theoretical framework through observation of how that framework operates in a virtual society.

What is significant in the DIAS-based approach described earlier is that it can place the agents as the progenitors of all action and allows for the integration of alternative modeling approaches, including models built by others in different computer languages, enabling a simulation effort to be more flexible to different perspectives in computational mechanics and theoretical application. Enabling agents to implement context-driven behavior, but not incorporating agents directly with behavioral models, also allows modelers to more easily scale-up agent behavior by making it possible for greater complexity that builds on existing work related to human choice and decision making. If Billy373 had additional and alternative behavioral models for analyzing his situation, then he may have decided that remaining in his previous household (Household 72) would have been economically more viable for his family. Perhaps he would have determined to resolve any conflict he had with Household 72's members. In any case, using the agent-aspect-process-model interaction outlined earlier, scaling-up and implementing different behavioral models are made easier. What the household diary example ultimately shows is that the tools for addressing highly complex human behavior, including but not only at the fundamental agent level, are present in ENKIMDU.

6.0 Conclusions and Future Direction

Although our simulation effort is still far from producing anything close to *all* of the relevant mechanisms affecting socioecological dynamics in ancient Bronze Age Mesopotamia, this brief overview of the ENKIMDU engine and its enabling technologies makes it clear that researchers have the ability to create complex agent-based simulations that can test varied theoretical approaches and address the numerous needs of researchers. The descriptions and examples presented highlight ENKIMDU's flexibility, scalability, and ability to produce expressive and socially plausible modeling results. In addition, these results can be relevant for varied social, spatial, and temporal scales.

At present only one settlement has been included in simulation scenarios; however, as the simulation begins to broaden its spatial scales and include numerous interacting settlements, regions, and migrant groups, it will be necessary to accurately observe all societal levels of interaction in order to better understand social phenomena. The ability of ENKIMDU to provide results relevant for micro- and macro-scales of society makes it a useful tool for testing social theory across these scales.

In addition, a well-constructed simulation structure insures that as new information, increased complexity, and different theoretical perspectives are applied, the chassis should still be able to function and provide useful insights into social-natural dynamics and emergent behavior at the desired scale. The structure of DIAS enables the simultaneous inclusion of alternative and multiple types of data and behavioral models. Furthermore, the units of time that can be simulated and capture behavior range from minutes and seconds to hundreds of years, allowing ENKIMDU to be temporally adaptable and applicable for different studies of past societies. Spatial variability can also be observed in ENKIMDU. For instance, agricultural yields for one specific field or the entire territory surrounding a settlement can be observed at any point of a simulation run. In the near future, we hope to further expand the spatial qualities of ENKIMDU to include large regional dynamics. In addition, GIS data encapsulated in common formats (e.g. ArcGIS shapefiles) can be incorporated into the DIAS-based simulation, making ENKIMDU well able to utilize existing spatial data.

Overall, the benefits of the enabling technologies used in ENKIMDU show that it is possible for scholars with varied interests and perspectives to find efficacy in creating socioecological simulations. This development helps social theorists move closer to the goal of producing platforms that are applicable for virtually every type of study regarding social dynamics in past human-environmental systems, perhaps even creating a unified approach in testing archaeological and anthropological theory. The intent of this paper is not to claim such grandiose achievements; however, it is our desire that social scientists interested in the past try to see relevance in the simulation approach we have taken in helping to achieve an effective system for testing theory. These benefits, we hope, can help others develop and build upon approaches that enable more significant progress in producing useful insights for various past ancient societies.

Acknowledgements and Further Information

This paper was made possible by the “Modeling Bronze Age Settlement Systems in a Dynamic Environment” project, a collaborative effort of the Oriental Institute and Department of Anthropology at the University of Chicago, the University of Edinburgh, and Argonne National Laboratory. We are particularly grateful to the National Science Foundation Program: Dynamics of Coupled Natural and Human Systems (Grant No. 0216548), for providing major funding for this project. Earlier exploratory work was funded by the University of Chicago/Argonne National Laboratory Collaborative Grants Program in 1998 and 1999 as well as by the University of Chicago, Advanced Technology Initiative in 2000. Further information regarding ENKIMDU can be obtained at: http://oi.uchicago.edu/OI/PROJ/MASS/Mass.htm. For information and questions about obtaining licenses for DIAS, access to specific ENKIMDU code and results, and other software tools mentioned in this paper please contact John Christiansen at jhc@anl.gov or Mark Altaweel at maltaweel@anl.gov.

Biographies

John Christiansen, Argonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL, USA 60439, jhc@anl.gov. John Christiansen is Deputy Director of the Advanced Simulation Technologies Center at Argonne National Laboratory. Mr. Christiansen’s

research over the past decade has focused on the development of flexible simulation frameworks that can be used to build very large, multidisciplinary complex adaptive system simulations, in applications that cover a wide range of domains and disciplines, including the natural and social sciences as well as health care and logistics. He is the creator of the ENKIMDU simulation engine and inventor of the DIAS and FACET modeling frameworks that support it.

Mark Altaweel, Argonne National Laboratory/University of Chicago, 9700 S. Cass Avenue, Argonne, IL, USA 60439, maltaweel@anl.gov. Dr. Altaweel received his Ph.D. at the University of Chicago in Near Eastern Archaeology, with a specialty in Mesopotamian archaeology. Currently, he is a scientist at Argonne assisting in the development of ENKIMDU and creating other social science modeling toolkits and applications relevant for modern and ancient societies. In addition to working at Argonne, Dr. Altaweel is leading a research projected related to ancient settlement development in northern Mesopotamia. His research interests include remote sensing and GIS applications in archaeology.

Appendix A: Examples of the Entity-Aspect-Process-Model Paradigm and Incorporation of an External Model

In the examples given below, all the methods do not have all their variables and method calls included. Rather, only the key portions of these methods are given to show the main concepts emphasized in the article. Readers are encouraged to contact Mark Altaweel (maltaweel@anl.gov) or John Christiansen (jhc@anl.gov) for further detail on the specific functionalities provided by DIAS and other technologies discussed.

A.1. Linking of an Entity with its Aspect, Process, and Model

Entities can be linked with behavioral models at various stages of the simulation, although often this link is created in the initialization stage. In the getModelsForRegion method, a HouseholdFieldManager entity is linked to the model called FieldCropManagmentModel. This method calls a DIAS method, called addProcessToModelCategoryMap, that takes the entity, aspect, process, and model classes and links them together. A Map instance is created and returns the connections for the simulation.

```
public java.util.Map getModelsForRegion(anl.dias.simulation.RegionDefinition region)
{
    Map categories = new HashMap();

   addProcessToModelCategoryMap(categories,  //map of models
       "Evolution",       //string category
       "FieldCropManagementModel",  //model name
       HouseholdFieldManager.class,           //entity class
       "fieldCropManagement",          //aspect in entity
        FieldCropManagementProcess.class,   //process class used
        FieldCropManagementModel.class);   //model class to be called

        return categories;          //categories map of entity-aspect-process-model returned
}
```

A.2 The Entity Object and its Parameters and Aspect

Appendix 1.1 shows how the connections for the entity-aspect-process-model are made; however, all of these objects are instantiated at an earlier stage. The example below shows the entity object mentioned in Appendix A.1. The static call registers this subclass with the superclass Entity. The other static methods create the actual aspects and parameters used for this entity. The method initializeMetaParameters creates a household and field parameters, while the initializeMetaAspects creates the aspect that has a string value that matches the aspect keyword in Appendix A.1. This aspect string allows the model and process classes to be linked to the entity class. The aspect string can also be used to reference alternative models.

```
public class HouseholdFieldManager extends Entity
{

//this registers the object
  static
  {
        initializeEntityMetaData(HouseholdFieldManager.class);
  }

//initialize the parameters
public static void initializeMetaParameters()
  {
     //the class to be called
     Class cls= HouseholdFieldManager.class;

      //the parameters for this class are initialized below
      addMetaParameter( cls, "household" ); // reference to a dwelling
      addMetaParameter( cls, "field" );
  }

//initialize the aspect
public static void initializeMetaAspects()
 {
     Class cls= HouseholdFieldManager.class;

     addMetaAspect( cls, "fieldCropManagement", "Routine agricultural tasks for a Crop
     Field" );
 }
}
```

A.3 The Creation of a Process Object

In this example, the process class extends the superclass CourseOfActionProcess, used for FACET type models. The static block registers the subclass, while the buildProcessInputData and initializeOutputParameters allow the mapping between the entity's variable parameters and associated model. The buildProcessInputData creates a Map instance that will store the involved variables referenced by the parameter values of the objects involved in the model. In this case, the process can be used to launch a model at any time, while other types of processes can be used to launch models when parameter variables are at the appropriate state for the model to function. The string value in the unpackageProcessOutputData method references the entity object to its specific parameter value that evolves as a result of the behavior. Each process object instance has to be associated with a specific model instance. Not all involved objects and parameters are shown in this example.

```
public class FieldCropManagementProcess extends CourseOfActionProcess
{

  protected Object buildProcessInputData()
   {

      // Process input data map in FACET models is as follows:
      // All contents of the map are available in the
      // participants to each Step.
      // Also in that map are (a) Step thisStep, and
      // (b) TimeInterval thisStepDuration.  In addition,
      // for Action Steps, the participants are there.

      Map data= new HashMap();  //data inputs
      HouseholdFieldManager hfm= (HouseholdFieldManager)getEntity();
      Household h= (Household)hfm.getValue( "household" );
      data.put( "household", h );  //puts the entity into the data map

      HouseholdOperations hops=
         (HouseholdOperations)h.getValue( "operations" );

      PendingCrop pc= hops.takePendingField();

      Field f= pc.getField();
      data.put( "field", f );

      data.put( "crop", pc );

      return data;
   }
//method to process output data from the actual model
   protected void unpackageProcessOutputData(Object data)
   {


            Field field =(Field)((Map)data).get("field");

           //links the output for this process back to
           //the parameterized entity

           connectOutputParameter(“cultivationState”,field);
         }
  }
```

A.4 FACET Model FieldCropManagementModel

In this FACET model, the constructor of the object is used to define the model steps. Local variables and their initial values used in the model can be defined in the constructor. The ResourcePackage class is used to hold the resources of the entities involved in the interaction. The rest of this constructor method creates action steps and participants in those steps (only one step shown in the example). The strings in the ActionStepTemplate object identify the name of the step, the timeout of the step, the next step after this step, and the duration of the step. Each of these strings reference methods that will be used specifically in the model. For example, the duration of the step can be determined by available resources of an entity that is calculated in the model. The next portion of the code refers to the participants and resource manager (i.e., the household) reserved for the step. Finally, the step is added to the overall model, and it is indicated this step is the first in the model. Similar to the other examples, comments are given in the constructor to facilitate interpretation of the methods called.

```
public FieldCropManagementModel()
   {
      super();


          addLocalVariable( "field", null );    //variable’s initial value

         //resources to be used in this model
      ResourcePackage rp= new ResourcePackage();
      RangedRequirement rr= new RangedRequirement(    RangedRequirement.MIN,
         "HeavyLaborer", 1 );

      rp.getResources().put( "HeavyLaborer", rr );

      //general exceptions in model are created here
```

```
        ExceptionStepTemplate generalTimeout= new ExceptionStepTemplate(
            "GeneralTimeoutProcessing", this,
            "nextStepAfterStepTimeoutGeneral", "stepTimeoutGeneral" );

        ActionStepTemplate action;  //the step object
        StepParticipantTemplate spt;  // the participant in step

        // Step 1: Clear and level the field
        action= new ActionStepTemplate("ClearAndLevel", this,  //title and object model
             "nextStepAfterClearAndLevel",        //next step after this step will be declared
             "clearAndLevel",  //name of the step that will be called
             "clearAndLevelTimeout",  //timeout method
             "clearAndLevelDuration" );  //method that will determine duration of step

        spt= new StepParticipantTemplate( "workCrew",  //the participant in the step
              "WorkCrew", new Integer( 1 ),  //role type and attention number
               new Requestee( "household" ),  //the manager of the resources used
               true, false );  //true refers to dismiss participants after step, false indicates only
                               //one resource object is used in the step

        action.addParticipant( spt );  //participant in step is added

         //this call below adds an exception to the model
         action.addPreemptiveEventHandler( "COAStepTimeout",     generalTimeout );

         addStep( action );      //step is then added to the model

        setEntryStep( "ClearAndLevel" );  //this sets the first step in the model
}
```

A.5 External Model Incorporation

This main method is run to initialize the remote external model, which is the SWAT model in this case. When the SWATModelControllerImpl is instantiated, DIAS methods will be called to initialize and register the model. For more on RMI see Reilly and Reilly (2002). The second method, the executeForTime method, resides in an object that wraps and calls the remote model to execute.

```
public static void main(String[] args)
{
            // this will load properties from the specified property files
            PropertiesUtils.loadProperties();

            SWATModelControllerImpl mc;
            initializeLogging();  //start logger object that will register messages from SWAT
```

```
        // An RMI registry is created/started so that the Remote external model can
        // register with it.  The model will be looked up by ENKIMDU when it is
        // run.

        //the port used for RMI
        int port = Integer.parseInt(System.getProperty("rmiregistry.port"));
        RMIUtilities.startRMIRegistry(port);

        //instantiate the external model control
        try {
                mc = new SWATModelControllerImpl(true); //true = remote model

            } catch (RemoteException e1) {
                e1.printStackTrace();
        }
}

//this method calls the execution of the external model within an object model that wraps
//the external model

public void executeForTime(RMIModelState modelState) {
        try {
                  Object output =
        ((SWATModelController)modelState.getModelController()).executeForTime((S
        WATStepData)modelState.getInputData());
                        modelState.setOutputData(output);

           } catch (RemoteException e) {

                        e.printStackTrace();
           }
        }
```

References

Anthes, RA, and Warner TT, Development of Hydrodynamic Models Suitable for Air Pollution and Other Mesometeorological Studies. *Monthly Weather Review*. 1978; 106:1045-1078.

Arnold JG, Srinivasin R, Muttiah RS, and Williams JR, Large area hydrologic modeling and assessment. Part I: Model development. *Journal of the American Water Resources Association*. 1998; 34(1):73-89.

Bagnall RS, and Frier BW, *The Demography of Roman Egypt*, Cambridge: Cambridge University Press; 1994.

Blaxter KL, *The Energy Metabolism of Ruminants,* London: Garland Publishing; 1967

Braudel F, *On History*, Chicago: University of Chicago Press; 1980.

Buringh P, *Soils and Soil Conditions in Iraq*, Baghdad: Ministry of Agriculture; 1960.

Butzer KW, *Archaeology as Human Ecology,* Cambridge: Cambridge University Press; 1982.

Carley KM, Prietula MJ, and Lin Z, Design versus cognition: The interaction of agent cognition and organizational design on organizational performance. *Journal of Artificial Societies and Social Simulation*. 1998; 1(3) http://ideas.repec.org/a/jas/jasssj/1998-7-1.html.

Christiansen JH, A flexible object-based software framework for modeling complex systems with interacting natural and societal processes. *Proceedings of the 4th International Conference on Integrating GIS and Environmental Modeling, Banff, Alberta, Canada, Sept. 2-8;* 2000a. http://www.colorado.edu/research/cires/banff/pubpapers/

— FACET: A simulation software framework for modeling complex societal processes and interactions. *Proceedings of the 2000 Summer Computer Simulation Conference of the Society for Computer Simulation International, Vancouver, British Columbia, Canada, July 16-20;* 2000b. http://www.scs.org/confernc/scsc/scsc00/text/scsc2000_author_kit.html

Christiansen JH, and Altaweel M, Simulation of natural and social process interactions: An example from Bronze Age Mesopotamia. *Social Science Computer Review*. 2006; 24(2):209-226.

Dahl O-J, and Nygaard K, SIMULA: an ALGOL-based simulation language. *Communications of the ACM*. 1966; 9(9):671-678.

Doran JM, Palmer M, Gilbert N, and Mellars P, The EOS project: Modeling Upper Paleolithic social change. In Gilbert N and Doran J, eds. *Simulating Societies*: *The Computer Simulation of Social Phenomena*, London: UCL Press; 1994:195-221.

Epstein J, and Axtell R, *Growing Artificial Societies*: *Social Science from the Bottom Up*, Cambridge: MIT Press; 1996.

Joussaume S, and Taylor K, The Paleoclimate Modeling Intercomparison Project. In Braconnot P, ed. *Proceedings of the Third PMIP Workshop, Canada, 4-8 October 1999*, WCRP-111, WMO/TD-1007; 2000: 271.

Kauffman SA, *The Origins of Order*: *Self Organization and Selection in Evolution*, Oxford: Oxford University Press; 1993.

Kohler TA., Kresl J, West CV, Carr E, and Wilshusen R, Be there then: A modeling approach to settlement determinants and spatial efficiency among late ancestral Pueblo populations of the Mesa Verde region, U. S. Southwest. In Kohler TA and Gumerman GJ, eds. *Dynamics in Human and Primate Societies*: *Agent-Based Modeling of Social and Spatial Processes*, New York: Oxford University Press; 2000:145-178.

Kohler TA, and West CV, The calculus of self interest in the development of cooperation: Sociopolitical development and risk among the northern Anasazi. In Tainter JA and Tainter BB, eds. *Evolving Complexity and Environment*: *Risk in the Prehistoric Southwest*, Reading, MA: Addison-Wesley; 1996:169-96.

Kutzbach JP, Behling P, and Selin R, *CCM1 General Circulation Model Output Data Set*. IGBP PAGES/World Data Center-A for Paleoclimatology Data Contribution Series # 96-027, NOAA/NGDC Paleoclimatology Program, Boulder, Colorado; 1996.

Lansing JS, *Priests and Programmers*: *Technologies of Power in the Engineered Landscape of Bali*, Princeton, NJ: Princeton University Press; 1991.

van Liere W, *Survey of Soil, Present Land Use and Land Capabilities of the Jezireh*, London: UCL Press; 2003.

Lurie GR, Sydelko PJ, and Taxon TN, An object-oriented geographic information system toolkit for web-based and dynamic decision analysis applications. *Journal of geographic information and decision analysis*. 2002; 6(2):108-116.

Murray MA, Cereal production and processing. In: Nicholson PT and Shaw I, eds. *Ancient Egyptian Materials and Technology*. Cambridge: Cambridge University Press; 2000:505-536.

Neitsch SL, Arnold JG, Kiniry JR, Srinivasan R, and Williams JR, *Soil Water Assessment Tool User's Manual*, Temple, TX: Texas Water Resource Institute; 2002.

Plotkin H, *The Nature of Knowledge*: *Concerning Adaptations, Instinct and the Evolution of Intelligence*, London: Penguin Press; 1994.

Redding RW, *Decision making in subsistence herding of sheep and goats in the Middle East*, Ann Arbor, MI: PhD dissertation, University of Michigan; 1981.

Reilly M, and Reilly D, *Java™ Network Programming and Distributed Computing*, Reading, MA: Addison-Wesley Professional; 2002.

Reynolds RG, An adaptive computer model for the evolution of plant collecting and early agriculture in the Eastern Valley of Oaxaca. In Flannery KV, ed. *Guila Naquitz*:

Archaic Foraging and Early Agriculture in the Oaxaca, Mexico, New York: Academic Press; 1986:439–500.

Schloen D, *The House of the Father as Fact and Symbol*: *Patrimonialism in Ugarit and the Ancient Near East*, Winona Lake, IN: Eisenbrauns; 2001.

Stöckle CO, Cambell GS, and Nelson R, *ClimGen Manual*, Pullman, WS: Washington State University; 1999.

Simon HA, *Models of Man*: *Social and Rational*, New York, Wiley; 1957.

Sweet L, *Tell Toqan*, *a Syrian Village*, Ann Arbor, MI: University of Michigan Press; 1974.

Wilkinson TJ. Archaeological survey of the Tell Beydar Region, Syria 1997. In Lerberghe KV and Voet G, eds. *Tell Beydar Environmental and Technical Studies*, Subartu 6. Turnhout: Brepols; 2000:1-37.

Wilkinson TJ, Gibson M, Christiansen JH, Widell M, Schloen D, Kouchoukos N, Altaweel M, Ur J, and Hritz C. Modeling settlement systems in a dynamic environment: Case studies from Mesopotamia. In Kohler TA and Leeuw SD, eds. *Modeling Socioecological Systems*, Santa Fe, NM: SAR Press; forthcoming (2007).

Wirfs-Brock R, Wilkerson B, and Wiener L, *Designing Object-oriented Software*, Englewood Cliffs, NJ: Prentice Hall; 1990.