Design for a Multilayer Model of Financial Stability:
Exploring the Integration of System Dynamics and Agent-based Models

Ignacio J. Martinez-Moyano[1] **David L. Sallach** **Mark J. Bragen** **Prakash R. Thimmapuram**

Decision and Information Sciences Division
Argonne National Laboratory
9700 South Cass Ave.
Argonne, IL 60439

630.252.8824	630.252.5760	630.252.5235	630.252.9291
imartinez@anl.gov	sallach@anl.gov	bragen@anl.gov	prakash@anl.gov

Abstract

Modeling for enhanced understanding of complex systems with policy-oriented implications sometimes requires that several different levels of aggregation be considered and formally included. In the system dynamics tradition, different levels of aggregation are not normally combined, which leaves certain classes of problems outside of the traditional use of system dynamics models. Agent-based models provide the ability to capture a fine level of detail of the system under study, but they lack the ability to parsimoniously and clearly link behavior to structure. This paper presents a domain in which a combined approach (hybrid model) seems to offer advantages. Additionally, two approaches to addressing the problem of integrating data from different levels of aggregation with the use of system dynamics and agent-based models are presented.

Keywords: Different Levels of Aggregation, Multilayer Modeling, Integration, Agent-based Models, Financial Modeling.

Introduction

To increase our understanding of the forces that shape the behavior observed in complex systems, we rely on both intuition and analysis (Hammond 2007). Intuition allows us to characterize behavior and its causes rapidly, but it lacks rigor and reproducibility, potentially leaving many questions unanswered. When the systems under scrutiny are complex and dynamic, analytical approaches seem more suited to providing robust answers to decision problems. In understanding complex systems, there are two (analytical) competing modeling approaches that differ in their orientation toward the level of detail to include in models of complex systems and their basic approach to understanding complexity (Scholl 2001a): system dynamics models (presenting a highly aggregated and feedback-rich view of the system using a deductive approach to understating behavior) and agent-based models (presenting a highly disaggregated view of the system in which behavior is emergent using inductive reasoning to generate it).

[1] Corresponding author at imartinez@anl.gov

System dynamics "is a perspective and set of conceptual tools that enables us to understand the structure and dynamics of complex systems" (Sterman 2000, p. vii); it focuses on an aggregate view of the system and emphasizes the role of feedback mechanisms and the endogenous nature of the observed behavior (Richardson 1991; Richardson 1996). The system dynamics approach emphasizes the use of dynamic complexity for enhanced understanding of complex systems (Forrester 1958, 1961). In the system dynamics approach, the structure of the system conditions the behavior that it exhibits (Forrester 1968; Richardson and Pugh 1981; Sterman 2000), and the way to change behavior is through changes in the structure of the system.

Agent-based modeling, in contrast, emphasizes the inclusion of as many details as possible to represent the individual characteristics of the elements of the system; it intentionally avoids the identification of feedback effects or endogenous mechanisms. The agent-based approach to modeling emphasizes the use of detail complexity to enhance understanding of complex systems. Agent-based models aim to examine global consequences of individual interactions as emergent characteristics of the system.

Both system dynamics and agent-based modeling have been used successfully to explore and understand complex systems with different characteristics. Researchers have explored the idea of comparing and contrasting the use of these two modeling approaches—aggregate models (system dynamics) and disaggregate models (agent-based)—as means to achieve enhanced understanding of complex systems (Scholl 2001a; Schieritz and Milling 2003; Pourdehnad, Maani, and Sedehi 2002). Some place this exploration in the broader social science agency/structure debate (Lane 2001), and some venture that "integrated research designs may have a high potential for result triangulation adding to model validity and result robustness" (Scholl 2001b). Furthermore, Kim and Juhn (1997) present a case for the use of system dynamics as a modeling platform for multi-agent systems.

In this theoretical strand of the literature, differences between the approaches are described in eight main elements. According to Schieritz and Milling (2003), system dynamics and agent-based modeling differ in their basic building blocks (feedback loops vs. agents), unit of analysis (feedback structure vs. agent rules), level of modeling (macro vs. micro), modeling perspective (top down vs. bottom up), adaptation process (change of dominant structure vs. change of structure), handling of time (continuous vs. discrete), mathematical formulation (integral equations vs. logic), and origin of dynamics in the system (levels vs. events).

A different strand of literature covers an applied view of how to combine the two modeling approaches. In this literature, a wide variety of combinations are presented. These combinations can be identified as part of a conceptual continuum in which one side advocates for the use of a system dynamics framework to create agents, and the other advocates for the use of an agent-based framework to capture feedback and aggregate dynamics. The two extremes try to exploit the usability of the approach to include the benefits of the other. Very few studies, however, present a mix of approaches (hybrid modeling) as a solution to deal with complex phenomena (Hines and House 2001). Akkermans (2001) uses the system dynamics framework to create agents and explore the dynamics of supply chain networks, while Schieritz *et al.* (Schieritz 2002; Schieritz and Größler 2003; Größler, Stotz, and Schiertz 2003) use the agent-based approach for macro modeling and the system dynamics approach for the agents (micro modeling). Wakeland

et al. (2004) and Parunak *et al.* (1998), alternatively, compare the two approaches (system dynamics and agent-based) by using both to simulate the same problem (nonequilibrium ligand-receptor dynamics over a broad range of concentrations and supply networks, respectively).

Research has been conducted comparing and contrasting system dynamics and agent-based modeling, and some of this research has been geared toward exploring under what circumstances it makes sense to use each one of them (for an example, see Rahmandad and Sterman 2004). The exploration of hybrid approaches as means to enhance understanding and insight generation of complex phenomena is not prevalent in the literature. Additionally, modeling for enhanced understanding of complex systems with policy-oriented implications sometimes requires that several different levels of aggregation be considered and formally included. In the system dynamics tradition, different levels of aggregation are not normally combined, leaving certain classes of problems outside of the traditional use of system dynamics models. Agent-based models provide the ability to capture a fine level of detail of the system under study, but they lack the ability to parsimoniously and clearly link behavior to structure. This paper presents a domain problem in which a combined approach seems to offer advantages. Additionally, two approaches to addressing the problem of integrating data from different levels of aggregation with the use of system dynamics and agent-based models are presented and discussed.

The paper is organized as follows; first, a domain problem is presented to clarify the context that catalyzed the idea of tackling the problem of integration of different levels of aggregation in modeling and simulation by using system dynamics and agent-based models. Second, two approaches to address the problem of integration are presented and discussed: a system-dynamics-centric approach and a controller-centric approach. Finally, conclusions and future work are discussed.

A Multilayer Model of Financial Stability

Financial infrastructure modeling is the domain problem that motivated us to think about integrating different levels of aggregation in models of complex systems. In our work, we model different critical infrastructures and their interrelationships by using the system dynamics approach in a project called the *Critical Infrastructure Protection Decision Support System* (CIP/DSS). To keep a central CIP/DSS financial infrastructure system dynamics model simple and integral while, at the same time, representing well the complexities of the banking and finance domain, we hypothesize that a hybrid model will effectively capture both the aggregate view of the infrastructure and the detailed indicators that can change its behavior abruptly.

If we were to have a combined financial model, we would want a detailed domain model[2] able to characterize the working financial context (phase)[3] in which a particular type of disruption, including complex, coordinated attacks, would propagate. We would want it to define the nonlinearities (context shifts) of possible trajectories (including aggregate flows) implied by a range of possible threat scenarios.

[2] Agent-based model.

[3] "Financial context" is meant to suggest the broader financial climate that may be indicated by a wide range of markets, indices, prices, and economic indicators. "Phase" is used to refer to the metric used to capture the range of possible contexts.

Dynamic Systems and Domain Models

Domain models can provide parameters that system dynamics models need to generate aggregate trajectories (e.g., market index patterns). Minimally, these parameters include the range and standard deviation for each variable used to define a phase distinction. More sophisticated parameters might include selected frequency distributions and/or specification of stochastic persistence (e.g., rescaled range analysis, Feder 1998, pp. 149-199). Additionally, domain models can help define: 1) accessibility and probability of phase shifts, 2) when a phase shift occurs, and 3) what new parameters accompany a phase shift. Domain (agent-based) models would act as providers of information to the system dynamics model operating at a different level of aggregation in the system.

In considering the integration of different levels of aggregation in a model, the division of labor between domain models and system dynamics models is a major design decision. Minimally, the system dynamics models should be able to generate a sequence (flow) of aggregate values that are appropriate for a given financial context and intermittently check whether, or be interrupted when, a new phase has been entered. A more extensive role for system dynamics models would include aspects of phase identification and management.

The present description has five distinct layers. The first four are represented by mechanisms representing global capital flows, exchange dynamics within the domestic economy, a postperturbation payment renewal process, and a scenario generator representing adversarial attack. The four levels interleave each other; the fifth level is the meta or integral layer that addresses the interaction of the first four layers.

Three Financial Layers

The first thee layers can be broadly conceived as differentiated by scale: global, national, and firm levels. They combine to represent large-scale financial transformations.

Global Capital Flows. A capital flow mechanism is drawn from Tirole's (2002) model of instability in emerging economies and is global in nature. Its focus is the tendency for capital to manifest disruptive flows during crises. This pattern can be observed as arising in endogenous financial dynamics and has the potential to be exacerbated during adversarial attacks. More particularly, a massive and sustained withdrawal of capital is a potential source of deep economic disruption and, accordingly, one of the fervent goals of terrorist movements.

From Tirole (2002), four moments can be identified: Foreign exchange rate depreciation, spillover and contagion, net capital outflow, and a decline in asset prices. In terms of immediacy and impact, these successive moments can be regarded as declining logarithmically. That is, a dramatic depreciation in exchange rates is not only immediately and quantitatively visible; it is a major means by which investors achieve capital relocation. Each subsequent moment is less immediate, less visible, and less relevant to a policy-oriented model.

Multiple foreign exchange markets provide investors with the ability to move into and out of national currencies. Exchange markets (in both electronic and open outcry forms) provide a vital and near real-time means of assessing immediate movements of global capital. Moving into a currency is a means of achieving economic goals such as discharging financial obligations,

purchasing inventories of various types, and acquiring or constructing durable assets. Moving out of a currency may express a desire to slow repayment, delay investment, and, in general, move to a safer investment climate. Capital flows are complex and unpredictable, and disruptions can create both investment problems and opportunities. For example, the opportunity to participate in the rebuilding of damaged facilities creates significant opportunities for capital flow. The unpredictability of capital flow is exacerbated by the fact that endogenous economic and financial dynamics are capable of producing dramatic capital flows that can mask or obscure responses to a crisis.

Although capital flows, as represented by shifts in exchange rates, have complex sources, integral structures can be identified. The primary means of identifying such structures is the use of computational taxonomy to classify existing empirical patterns into phases. From these phases, trend and volatility parameters can be identified and conveyed to the system dynamics model. This is the first and most important role of the global capital flow mechanism.

Spillover and contagion concern possible regional effects of capital withdrawal. Specifically, capital can withdraw to the region immediately surrounding the affected nation, or it can withdraw from an entire region. The issue can be seen as one of assessing the scale of disruption and constitutes a secondary focus of capital flow model.

The third and fourth moments, net capital outflow and decline in asset prices, are less visible and slower to manifest, with data less readily available. As a result, they are less useful to a policy-oriented model and will be used primarily to validate and/or confirm trends suggested by the first two moments.

Economy and Exchanges. In the same way that foreign exchanges allow insight into global capital flows, stock markets affect and represent the larger economy with its diverse and intertwined industries. From the standpoint of disruption, the two major categories of events are: 1) major market shifts, most of which are entirely endogenous, and 2) material and social disruptions.

Markets make continual adjustments in response to economic news, shifts in preferences, supply availability, and other factors. Market prices influence available liquidity not only directly (Tirole 2002) but indirectly via the mechanisms of margin calls and circuit breakers. In periods of price decline, the former are designed to defray risk by increasing collateral, while the latter impose a cooling-off period. Accordingly, in times of crisis, these two mechanisms provide alerts to possible and actual disruptive events and the opportunity to assess their economic impact.

As in the global capital flow mechanism, historical data are used to identify complex market patterns, including trends and volatility among multiple instruments and indices. The resulting parameters should be conveyed to a system dynamics model where they are used to determine rates of change.

The exchange component represents the flow of trading activity through the execution, clearance, and settlement phases. Trading activities are processed only during normal (user-specified) operating hours on normal operating days subject to the availability of the exchange. This

availability can be limited by infrastructure and workforce availability as well as automatic shutdown due to market conditions.

The margin call component represents the flow of margin trading activity. Two types of margin trading that should be captured are:

1. Margin trading that takes place during stable market operation
2. Margin calls that take place during excessive market volatility

The stable market portion uses historical data to generate activity during the model run. The market volatility portion monitors the market index, comparing its changes to three user-specified limits. Additional margin call trades, based on the specified margin call reaction parameters, are generated when the limits are surpassed.

Market dynamics are inherently complex and cannot be reliably predicted. The CIP/DSS simulation model represents only market institutions under stress. In particular, the mechanisms of margin calls and circuit breakers are used to identify dramatic shifts in exchange-traded instruments, calculating the consequences that are likely to ripple through the larger economy as a result of a diverse range of adversarial scenarios.

Payment Resumption and Emergent Liquidity Risk. There are two kinds of potential robustness: that related to physical infrastructure and operations, and that related to institutional responses. As an example of the latter, an adversarial attack on the international financial infrastructure designed to catalyze systemic risk can be exacerbated, albeit inadvertently, by a reluctance of institutions to resume payments until they have begun to receive a flow of payments owed them. Because there is a densely connected network of obligations, each delayed response, measured in hours, has the potential to create, and then intensify, a liquidity crisis, correlatively deepening the danger of national and global systemic risk.

It is necessary to understand what motivates institutions and pertinent decision makers to continue payment flows under difficult circumstances, at a particular time. Relevant factors might include the credit market and the credit quality profiles of situated obligatees, how various sectors of the market have been affected by the disruption, whether there have been multiple coordinated attacks and/or a troubling pattern of attacks, the way in which the responses of financial institutions are influenced by the actions of downstream customers, the strategic responses of variously situated businesses, and other widely diverse incentives.

Each of these factors can be incorporated into a fine-grained model with the potential to clarify prospective institutional responses to multiple interacting risks. The model would require representation of: banks and businesses at diverse scales; the skein of relationships among the various institutions, including densely correlated networks of interbank loans; credit quality by sector, region, and, ultimately, institution; and the nature, pattern, and consequences of the specific attack.

Layer Integration. Taken together, the three financial layers represent different scales of interaction and variegated types of risk. The capital flow mechanism is global in scope and

places national issues in the context of international investment decisions. Because the dollar is the standard unit of exchange for those wishing to invest in the United States, because the foreign exchange markets express changes in the strengths and weaknesses of the dollar on a real-time basis, and because currencies are exchanged on a number of electronic markets (thereby minimizing the role of physical disruption), this is the simplest of the three layers. However, the capital flow mechanism still faces the difficulties of inferring capital flows from a set of relatively valued currencies in the context of diverse financial conditions, while still providing an intuitive indicator to the system dynamics model.

The domain model should calculate a Hurst phase exponent (HPE) by using computational taxonomy to identify the underlying phases of currency interaction. The HPE should be passed to the system dynamics model as a persistence ("stickiness") indicator for the relevant phase. This persistence indicator can then be used (by either the system dynamics model or the domain model) to generate a sequence of values that indicate the general climate of capital dynamics. With the use of a mixed initiative system, this stickiness parameter can be checked by the system dynamics model, either periodically or when the flow values approach a recognized threshold. Alternatively, the domain model can use its far more detailed structure as a basis for interrupting the system dynamics model to provide it with updated stickiness parameters.

The economy and exchanges mechanism is national in scope and more complex in focus. The exchanges are mostly physical in form, and thus can be the target of attack with resulting disruption. In periods of nonlinear change, the exchanges have protective mechanisms of circuit breakers and margin calls, which are explicitly modeled. It has payment and clearing infrastructures that can be targets of attack and thus be disrupted, causing further downstream effects. Finally, the economy incorporates diverse industries with varied geographical distributions and effects.

Although these structures can certainly give rise to important economic and financial interactions, they are too detailed to be directly represented in the system dynamics model, which is one reason for maintaining a complementary domain model. However, as with the capital flow model, it is necessary to provide parameters that allow the system dynamics model to generate aggregate trends. As in the capital flow mechanism, an HPE will be calculated on the basis of phases of a key economic index for calculating economic/financial values required by other infrastructural models. As necessary, these parameters are updated to capture changes in the overall economic climate.

However, if the resulting economic phases are more diverse and more complex, the process of communicating with the system dynamics model need not be. Because there are market indices that are widely recognized as providing insight into the state of the economy (e.g., Dow Jones Industrial Average, Standard & Poor's, and Wilshire 5000 Index), the aggregates that a system dynamics model needs to track to represent the evolution of the national economy are reasonably clear. Further, this common use of stickiness allows two mechanisms of quite divergent scope and content to interact with the system dynamics model by using a common mixed initiative interface.

The third mechanism, representing payment resumption, is potentially the most local and detailed of the three. It represents firm-level decision making in the face of unanticipated and disturbing circumstances. For this to be effective, the payment resumption mechanism needs to take into account the factors to which decision makers give weight, some of which are summarized in the preceding section.

Adversarial Layer

The fourth layer is inescapably military in nature. It concerns the preparation and conduct of attacks designed to achieve maximum disruption of the financial system of the United States and, more broadly, the global economic system. Such attacks may be coordinated in complex ways and be of long duration. Therefore, it is necessary to construct a scenario generation system that can probe the dynamics of the first three layers and reveal structural deficiencies that cause such attacks to have a more disruptive impact than might be expected.

The Integral Layer

The entire model has its ultimate expression as a system dynamics structure. However, because of the complexity of the domain, it must also have a domain-specific model that is exogenous to the dynamics and that drives them during critical phase changes. This interaction between the domain model and the system dynamics model is one of the principal concerns of the integral layer.

The integral layer defines the architecture through which the two models communicate. As summarized in Table 1, different types of interaction serve different purposes. Construction of the integral layer will involve explorations of the trade-offs between rich domain representation versus potential computational overhead.

The integral layer defines the architecture through which the two models communicate. There are three possible modes: 1) external, 2) mixed initiative, and 3) real-time DSS. The first mode uses the scenario generator to define relevant events and simulates relevant time-series results. The latter can then be incorporated into the system dynamics model as data. The second mode involves interaction between the domain model and the system dynamics model, with either one initiating the process.

The third mode alternative (real-time DSS) is more complicated because it involves the use of data from an actual incident to parameterize the relevant models, including adjustments when subsequent data indicate a variance from simulated results. In this case, the communication between the domain model and the system dynamics model can use whatever available mechanisms or initiatives seem most appropriate.

Table 1. Types of Interaction between the Domain and System Dynamics Models

Type of Interaction	Description
Scenario exploration	Domain model is run first and results are sent (via time-series) to the system dynamics model at the start of the simulation.
Intertwined models	Domain and system dynamics models alternate. Each runs until the model either: a) has completed a specified time block, b) requires information from the other model, or c) simulated values move into a region that requires fresh calculations performed by the other model.
Crisis response	Real-time empirical data are loaded and processed in the domain model, which simulates time-series that are passed to the system dynamics model.

Approaches to Solving the Integration Problem

Mechanism Integration

The integration of multiple mechanisms at differing levels of aggregation means that the software expressing the resultant model will need to support intermechanism communication. At some point, we may have integrated development environments that support the interleaving of fine- and coarse-grained communication flows across diverse domains but, at present, multimechanism modeling will frequently need to create customized intertool information flows. Accordingly, this section describes two approaches that we developed to address the integration problem. First, a system-dynamics-centered approach is presented. In this approach, we use a system dynamics model as the center and controller of operations and communication with agent-based models. Later, we present a controller-centric approach that addresses the interaction and communication outside of both the system dynamics and the agent-based models.

The System-Dynamics-Centric Approach

This section describes a proof-of-concept approach to the integration problem: a system-dynamics-centric approach (SD-centric approach). An SD-centric approach means that the system dynamics model is the primary, controlling model and that the role of the agent-based model is to provide aggregated view of agent details when requested to do so.

In a system dynamics model, values for stocks, flows, and auxiliary variables are computed at each time step. We can conceptually think of the current condition of the model as numbers in a spreadsheet in which each column contains a variable and each row contains the values for the complete set of variables for a single time step. The new values for the next time step are computed from previously computed values. In the SD-centric hybrid model, some cells of the current-condition spreadsheet can be filled or accessed by either the system dynamics or agent-based models. These data values need to be stored in a staging area that can be easily accessed by either model. To this end, we have designed a Data Manager that holds only data that need to

be shared between the agent and system dynamics models providing an easy access communication link. Data specific to each model are stored within the models themselves. To share these data, there must be a data storage and retrieval mechanism. The Data Manager's structure, communication, and data flows are depicted in Figure 1. Because this is an SD-centric model, the agent model operates in a reactive role. The system dynamics model will request data or push a computed value to the Data Manager. If the data requested by the system dynamics model are not available in the Data Manager, the Data Manager will request the data from the agent-based model. If these data have not been computed by the agent-based model, it will execute as many time steps as required to compute the value and return it, and then pause and wait for the next request. If the requested data are available in the Data Manager, then no interaction with the agent model is required.

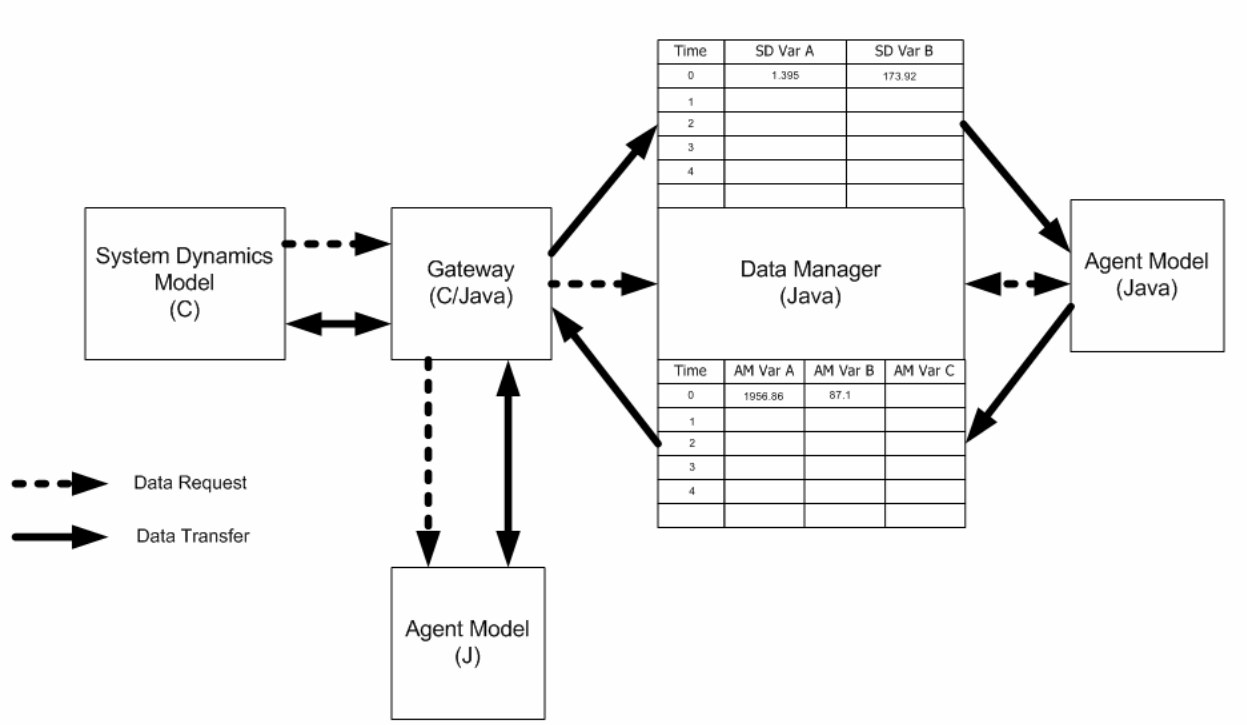

Figure 1. Data Manager structure, data, and communication flows.

In an SD-centric model, the agent-based component will provide data only for flows and variables. It may, however, compute its flow values by using both stocks and flows from the system dynamics model. To accomplish this, we created a *Get* mechanism and a *Set* mechanism that the system dynamics model can use to communicate with the agent-based model. We created a piece of formulation, called the *gateway*, which handles the data transfers between the system dynamics model and the Data Manager. The Data Manager and the agent-based model communicate directly. The *gateway* is an extension to the external function capability that is provided with the Vensim software. On the Vensim side, it is implemented in the C programming language. The system dynamics modeler simply enters calls to the *gateway* external-function: *Get* or *Set* data. The request is reformatted and passed from C to the Java side of the gateway via Java Native Interface (JNI) calls. The *gateway* will then route the request to the appropriate Data Manager/agent-based model for processing.

In our proof-of-concept work, two languages were used to implement the agent-based models: Java and J. (J is an interpreted, APL-like language. See www.jsoftware.com for complete documentation on this language.) Four external functions have been defined within the controller:

- JavaSet('Var_Name', SD_Variable, Time, Time_Step)
- JavaGet('Var_Name', Time, Time_Step)
- JSet('Var_Name', SD_Variable, Time, Time_Step)
- JGet('Var_Name', Time, Time_Step)

Where:

- Time is the system dynamics 'Time' variable,
- Time_Step is the time step size specified in the system dynamics model (integration interval),
- SD_Variable is the system dynamics variable that holds the value that is to be transferred to the agent-based model, and
- 'Var_Name' is the variable name in the agent-based model that is either to be retrieved or updated. (Note that in the current implantation, the J Var_Name is actually a J verb name that will perform the operation while the Java implementation uses this name as a key for a lookup table to identify the appropriate object to be accessed.)

In our proof-of-concept implementation, we specified identical time units and time step sizes to simplify the data transfers. See Conclusions and Future Work for a brief discussion on working with differing time units or time step sizes.

The Java agent model uses the Java Data Manager (JDM) that interacts with the *gateway*. It is the JDM that fields the requests for data and returns the appropriate aggregated data. The agent-based model is responsible for aggregating the data, and it must register with the JDM the objects that can provide the aggregated data. The current J implementation uses a series of verbs that perform the same operations as the JDM. Figure 2 shows the structure of a combined SD-centric model that contains all of the currently available *Get* and *Set* operations with Java and J models. Variable names that appear in either orange or green represent variables that *Get* or *Set* agent-based model variables. A description of the model appears below.

To illustrate the available flexibility in interaction between the system dynamics and agent-based models, the model implements a data refresh variable named *Data Refresh.* This is a Boolean variable that specifies whether or not there should be a *Get* on an agent-based model variable. *Data Refresh* can be set true under one of two conditions: 1) the normal refresh cycle defined in the model has been reached (e.g., get a value every 24 hours) or 2) a *Situation Refresh* has been detected in the agent-based model (e.g., an event has been detected by the agent-based model of which the system dynamics model must immediately be made aware). Each *Set* operation is executed for each clock tick. The agent-based model will determine whether or not to process these data. The *Situation Refresh* variable is set by requesting that the J agent-based model return the current value of the 'situation' variable. It is retrieved via:

JGet('situation', Time, TIME STEP)

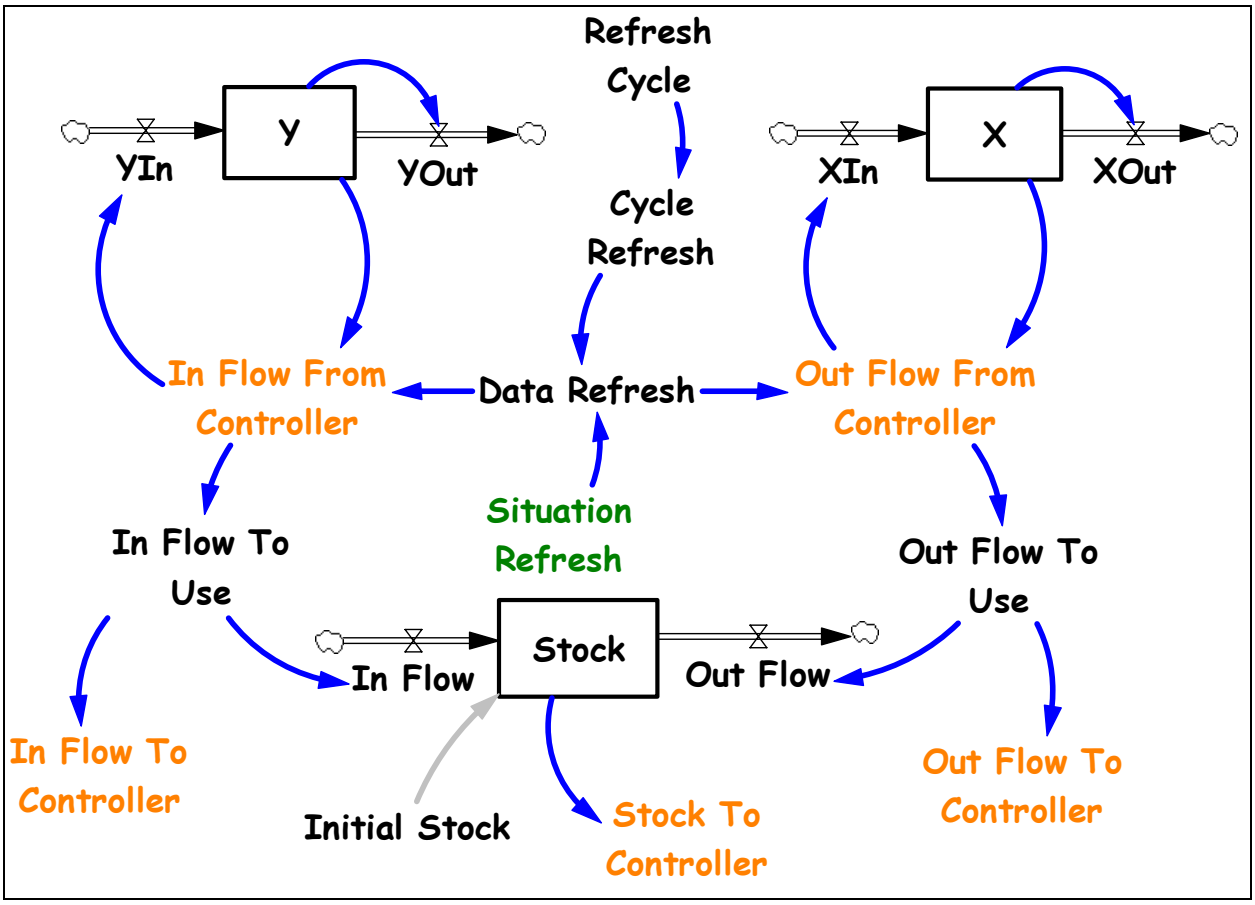

Figure 2. Simple model illustrating constructs for system dynamics and agent-based models interaction.

Because both the *Cycle Refresh* and *Situation Refresh* variables are Boolean, the Vensim formulation to determine if an agent-based model access is required is simply:

MAX(Cycle Refresh, Situation Refresh)

The *In Flow* variable will be set either from a data refresh operation or use the value that is currently stored in the *Y* stock variable. The *Y* and *X* stock variable formulation is simply used to save the values used in the previous clock tick. The *Out Flow* variable is handled in an identical manner. In our proof-of-concept implementation, the *In* variables were processed in Java and the *Out* variables were processed in J. This was done to show that both languages could be accessed together. Normally, the entire agent-based model would be written in a single language. Note that multiple agent-based models can be accessed in a single SD-centric combined model. The Vensim formulation to determine which *In Flow* to use is:

IF THEN ELSE
(Data Refresh = 1, JavaGet('InFlow', Time, TIME STEP),Y / Units Adjust)

Where *Units Adjust* is available to convert agent-based model units when it is needed. The "*To Controller*" flow variables will set the agent-based model variables on each clock tick. The *Stock to Controller* variable is used to illustrate that both stock and flow values can be passed to the controller. The Vensim formulation to pass these data to the agent-based model is:

JavaSet('InFlowUse', In Flow To Use, Time, TIME STEP)

Each agent-based model is initialized as the system dynamics model begins execution. For Java, this includes starting the Virtual Machine (VM) and instantiating each of the agent objects. These objects register their variable access methods with the Java Data Manager. For J, this includes establishing a socket with the J interpreter and loading all J agent-based model verb definitions.

The Controller-Centric Approach

We developed an alternative approach to the system-dynamics-centric approach presented in the previous section. We decided that, to be able to combine different system dynamics models and different agent-based models, all at the same time, it made sense to formulate a controller outside of any of the models. This approach is what we call the controller-centric approach (Figure 3). In this approach, a controller is built outside the system dynamics model and outside any agent-based model. It acts as the entity in charge of controlling operations of the models and controlling communication among models. The controller keeps the pace of the overall advancement of the simulation and keeps track of what information, and when, is shared among models.

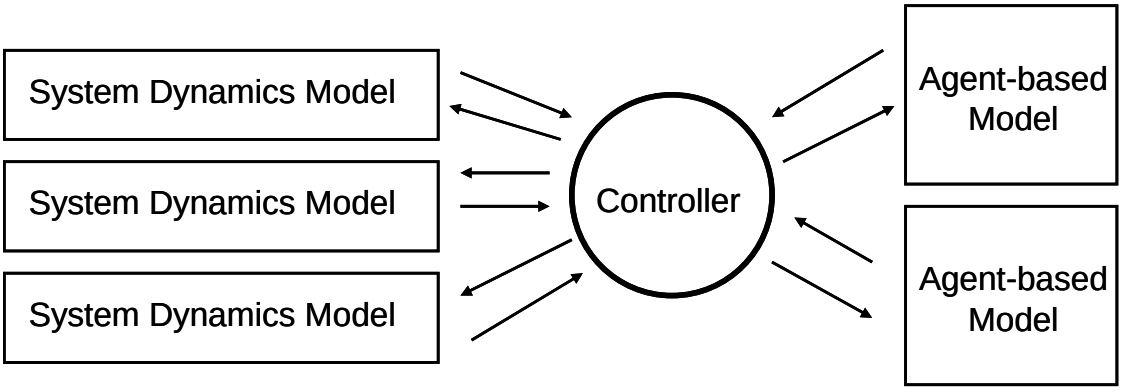

Figure 3. Controller-centric approach.

Vensim provides a DLL, which can be called by other applications. The Vensim DLL can display sketches, tool output, setup, and perform simulations and manipulate data. The Vensim DLL cannot build or modify models, but it does supply some functionality for retrieving information about a model. In the present work, we developed a controller in J Software by using the Vensim DLL. This controller can load models, setup game, simulate, continue game, and retrieve model variables and attributes.

In Figure 4, an example of the interface developed in J to control the operation and communication between system dynamics models and agent-based models is shown. Future work will extend this approach to apply this controller to the CIP/DSS banking and finance model developed in Vensim.

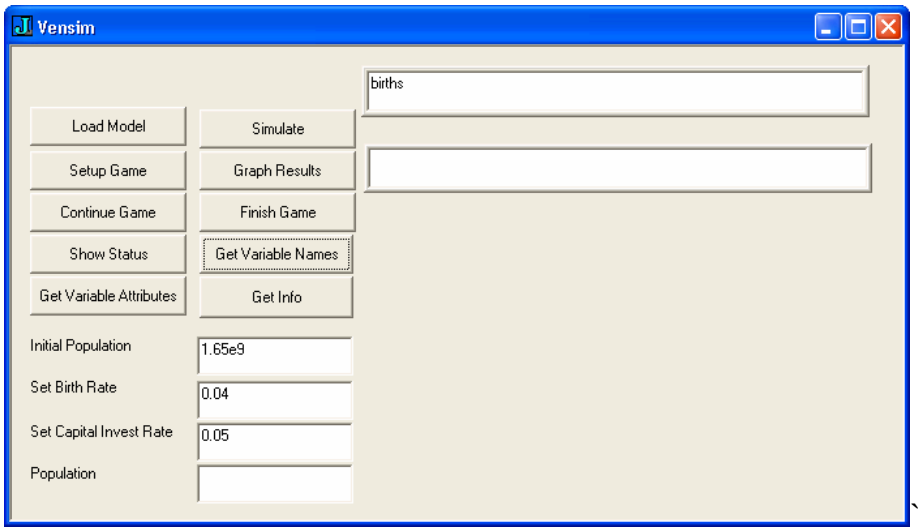

Figure 4. Controller built in J.

Conclusions and Future Work

Our work has allowed us to recognize the main problems associated with combining system dynamics models and agent-based models to capture subtle intricacies of complex systems in certain domains. It became clear that combining such models can be done, but it requires much effort and precision with respect to what information is shared and how often. Additionally, the work so far suggests that the identification of specific pieces of information to be transferred from one type of model to another is a very promising area of study. This clarification is evident only after the effort of finding the interacting elements is exerted.

From a modeling perspective, our future work will center on implementing a new agent-based model of bank communication networks and integrating it into an existing system dynamics model of the banking and finance sector in the CIP/DSS.

From a software perspective, our future work will focus on performance issues in the controller and agent-based model data managers. In an ideal situation, the system dynamics and agent models will use the same time unit and time step size. In this case, unadjusted data can be used by both models. However, if the models use different time units or time step sizes, conversion will need to be applied to the data. For example, if the system dynamics model uses a time unit of minutes and the agent model uses a time unit of hours, the rates passed between the two models must be converted (i.e., widgets/hour to widgets/minute for a rate passed from the agent model to the system dynamics model). While the mathematics involved in converting the data is trivial, we will be analyzing the implications and possible inconsistencies generated by working with models of differing temporal fidelity.

Acknowledgments

This work was funded by the U.S. Department of Homeland Security. The views and conclusions expressed in this paper are those of the authors and do not reflect the views or policies of the U.S. Department of Homeland Security.

References

Akkermans, Henk. 2001. Emergent Supply Networks: System Dynamics Simulation of Adaptive Supply Agents. In *The 34th Hawaii International Conference on System Sciences (HICSS'01)*. Hawaii.

Feder, Jens. 1998. *Fractals*. New York, NY: Plenum Press.

Forrester, Jay Wright. 1958. Industrial Dynamics: A Major Breakthrough for Decision Makers. *Harvard Business Review* 26 (4):37-66.

———. 1961. *Industrial Dynamics*. Cambridge, MA: Productivity Press.

———. 1968. *Principles of Systems*. Cambridge MA: Productivity Press.

Größler, Andreas, Myrjam Stotz, and Nadine Schiertz. 2003. A Software Interface Between System Dynamics and Agent-Based Simulations – Linking Vensim® and RePast®. In *The 21st International Conference of the System Dynamics Society*. New York, NY: System Dynamics Society.

Hammond, Kenneth R. 2007. *Beyond Rationality: The Search for Wisdom in a Troubled Time*. New York, NY: Oxford University Press.

Hines, Jim, and Jody House. 2001. The source of poor policy: Controlling learning drift and premature consensus in human organizations. *System Dynamics Review* 17 (1):3-32.

Kim, Dong-Hwan, and Jae-Ho Juhn. 1997. System Dynamics as a Modeling Platform for Multi-Agent Systems. In *International Conference of the System Dynamics Society*. Istambul, Turkey: System Dynamics Society.

Lane, David C. 2001. Rerum cognoscere causas: Part II: Opportunities generated by the agency/structure debate and suggestions for clarifying the social theoretic position of system dynamics. *System Dynamics Review* 17 (4):293-309.

Parunak, H. Van Dyke, Robert Savit, and Rick L. Riolo. 1998. Agent-Based Modeling vs. Equation-Based Modeling: A Case Study and Users' Guide. In *Multi-agent Systems and Agent-based Simulation (MABS'98)*, edited by L. N. i. A. I. Series. Berlin, Germany: Springer.

Pourdehnad, John, Kambiz Maani, and Habib Sedehi. 2002. System Dynamics and Intelligent Agent-Based Simulation: Where is the Synergy? In *International Conference of the System Dynamics Society*. Palermo, Italy.

Rahmandad, Hazhir, and John D. Sterman. 2004. Heterogeneity and network structure in the dynamics of contagion: Comparing agent-based and differential equation models. In *Engineering Systems Division, Working Paper Series, ESD-WP-2004-05*. Boston, MA: Massachusetts Institute of Technology.

Richardson, George P. 1991. *Feedback Thought in Social Science and Systems Theory*. Edited by P. Communications. Second Edition ed, *System Dynamics Series*. Waltham, MA: Pegasus Communications.

Richardson, George P. 1996. System Dynamics. In *Encyclopedia of Operations Research and Management Science*, edited by S. I. Gass and C. M. Harris. Boston, MA: Kluwer Academic Publishers.

Richardson, George P., and Alexander L. Pugh, III. 1981. *Introduction to System Dynamics Modeling with DYNAMO*. Cambridge, MA: Productivity Press.

Schieritz, Nadine. 2002. Integrating System Dynamics and Agent-Based Modeling. In *International Conference of the System Dynamics Society*. Palermo, Italy: System Dynamics Society.

Schieritz, Nadine, and Andreas Größler. 2003. Emergent Structures in Supply Chains: A Study Integrating Agent-Based and System Dynamics Modeling. In *The 36th Hawaii International Conference on System Sciences (HICSS'03)*. Hawaii.

Schieritz, Nadine, and Peter Milling. 2003. Modeling the Forest or Modeling the Trees: A Comparison of System Dynamics and Agent-Based Simulation. In *The 21st International Conference of the System Dynamics Society*. New York, NY: System Dynamics Society.

Scholl, Hans J. 2001a. Agent-based and System Dynamics Modeling : A Call for Cross Study and Joint Research. In *The 34th Hawaii International Conference on System Sciences (HICSS'01)*. Hawaii.

———. 2001b. Looking Across the Fence: Comparing Findings from SD Modeling Efforts with those of other Modeling Techniques. In *International Conference of the System Dynamics Society*. Atlanta, GA: System Dynamics Society.

Sterman, John D. 2000. *Business Dynamics: Systems Thinking and Modeling for a Complex World*. First ed. Boston MA: Irwin McGraw-Hill.

Tirole, Jean. 2002. *Financial Crisis, Liquidity, and the International System*. Princeton, NJ: Princeton University Press.

Wakeland, Wayne W., Edward J. Gallaher, Louis M. Macovsky, and C. Athena Aktipis. 2004. A Comparison of System Dynamics and Agent-Based Simulation Applied to the Study of Cellular Receptor Dynamics. In *The 37th Hawaii International Conference on System Sciences (HICSS'04)*. Hawaii.