Proceedings of the 2007 Winter Simulation Conference
S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds.

REQUIREMENTS AND DESIGN PRINCIPLES FOR MULTISIMULATION WITH MULTIRESOLUTION, MULTISTAGE MULTIMODELS

Levent Yilmaz
Alvin Lim
Simon Bowen

Computer Science and Software Engineering
107 Dunstan Hall, Auburn University
Auburn, A.L. 36849, U.S.A.

Tuncer Ören

M&SNet: Ottawa Center of the MISS
SITE, University of Ottawa
Ottawa, ON K1N 6N5, CANADA

ABSTRACT

The significance of simulation modeling at multiple levels, scales, and perspectives is well recognized. However, existing proposals for developing such models are often application specific. The position advocated in this paper is that generic design principles for specifying and realizing multiresolution, multistage models are still lacking. Requirements for simulation environments that facilitate multiresolution multistage model specification are introduced. A multimodel specification formalism based on graph of models is suggested along with design precepts to enable flexible dynamic model updating. The notion of multisimulation is introduced to enable exploratory simulation using various types of multimodels.

1 INTRODUCTION

Multilevel and multistage modeling is highly interdisciplinary, with progress occurring independently across fields (Davis 2000; Takahashi et al. 2002). Yet, disparate efforts involving the design and development of various types of multimodels result in incoherent solutions that lack a common unifying framework. This proposal is partially based on the rationale depicted with a taxonomy (Yilmaz and Ören 2004) of discrete-event based multimodel methodology that is equally applicable for continuous and discontinuous-change models expressed in ordinary differential equations and hence explores the viability of realization of a unified multimodel formalism. The premise is that, by developing a generalized multimodel formalism that can be configured with alternative strategies one can instantiate various types of multimodels within the same model development framework. Such a strategy is in sharp contrast with the current trend in developing formalisms for different modeling needs, where distinct solutions for multiple representations of scale, resolution, and staging are defined. More importantly, the availability of ensemble of models encapsulated within a multimodel will facilitate exploratory experimentation with computational models, the lack of which is a significant limitation in supporting discovery-based science. The following characteristics of simulations exacerbate the design challenges for modeling and simulating entities at multiple levels of resolution.

- For most realistic problems, the nature of the problem changes as the simulation unfolds. Initial parameters, as well as models can be irrelevant under emergent conditions. Relevant models need to be identified and instantiated to continue exploration. Manual exploration is not cost effective and realistic within a large problem state space (Yilmaz and Tolk 2006).
- Our knowledge about the problem being studied may not be captured by any single model or experiment (Bankes 1999). Instead, the available knowledge is viewed as being contained in the collection of all possible modeling experiments that are plausible given what is known and what is learned.
- Dealing with uncertainty is paramount to analyzing complex evolving phenomena (Davis 1988). Adaptivity in simulations and scenarios is necessary to deal with emergent conditions for evolving systems in a flexible manner.
- As simulations of complex phenomena are more and more used to aid intuition, dynamic run-time simulation composition with exploratory simulation will help identify (re)solution strategies that are flexible, adaptive, and robust (Yilmaz and Tolk 2006).
- In symbiotic simulation (Fujimoto et al. 2002) as well the problems advocated by the NSF DDDAS program, information about the actual performance and accuracy characteristics of the system are acquired during actual simulation rather than before. Run-time model and simulation updating

can be used to advance the symbiotic online simulation field, where models, as well as parameter spaces may evolve.

- In a recent IEEE Spectrum article (Guizzo 2004) on UrbanSim, urban planners look decades ahead, experts conclude as follows: "…. ideally, stakeholders should be able to see computer-generated streetscape animations of **different** scenarios—what it would be like to walk in the neighborhood or commute to work, for example. It would be easier then for people to visualize the effects of **alternative** development policies and projects…."

As a result, multifaceted, multiparadigm (Zeigler and Ören 1986; Fishwick and Zeigler 1992), multiresolution modeling (Davis 2000) and cross resolution consistency management in multiresolution simulations (Reynolds et al. 1997) are explored to layout observations and the conceptual foundations to better understand the challenges in developing advanced simulation modeling infrastructures that can deal with uncertainty and multilevel phenomena. However, there are a number of limitations of existing proposals:

1. The lack of fundamental requirements and design principles results in application specific realization of such models;
2. The lack of a unified and coherent framework that enables development of various types of multimodels results in inconsistent and ad hoc development;
3. The lack of dynamic model and simulation updating and branching limits the potential of run-time exploration using simulation.

The purpose of this paper is to (1) reiterate the rationale for using novel simulation modeling such as multimodels, and multiresolution modeling, (2) present a comprehensive taxonomy of multimodels, and (3) layout fundamental requirements in developing simulation modeling infrastructures that enable multimodel design. A graph-based multimodel specification formalism is presented along with its basic syntax and semantics to facilitate developing simulators for such multimodels.

The rest of the paper is organized as follows. In section 2 we overview existing work on multimodels, with special emphasis on limitations in dealing with the type of problems advocated in this section. Section 3 presents a comprehensive taxonomy of multimodels. Note however, this paper focuses on designing **M**ulti**R**esolution **M**ulti-**S**tage **M**ultimodels (MRMSM) and the use multisimulation to generate their behavior. In section 4 we elaborate on requirements for MRMSM. A model specification formalism is advocated in section to improve dynamic model and simulation updating, which is a critical and essential issue in MRMSM design. Section 6 elaborates on a selected subset of MRMSMs to suggest how models need to be designed to satisfy the requirements set forth in section 4. Section 7 introduces the multisimulation concept. Finally, section 8 concludes by summarizing the contribution of the paper.

2 BACKGROUND

Several projects have explored the issues involved in the construction of multiresolution models. Multiresolution modeling (MRM), originally proposed by Davis and Bigelow (2002) was concerned with resolving the representational and conceptual differences among models that are joined to satisfy a common objective. Davis (2000) builds his exploratory analysis methodology on MRM, which involves building a single model, a family of models, or both to describe the same phenomena at different levels of resolution (Davis and Bigelow 2002). One of the limitations of this approach as it pertains to type of problems depicted above is that models are statically defined at the design time. The level of granularity at which resolution levels are defined in the methodology is in terms of functions and their parameters. Updating the functional decomposition model at run-time by adding new branches or extending the depth of the tree is not part of the methodology. Furthermore, if there are multiple models that can be coupled to inputs of an aggregate model to obtain a high-resolution model, different models can be defined only through design time coupling. The analyst then selects a particular high-resolution model to experiment. While it is possible to instantiate each one of the models to perform multiple simulation experiments, one has to foresee in advance what high-resolution models are plausible during the design time. Extensibility and adaptivity is not engineered into the simulation. Hence, simultaneous exploration with multiple models based on run-time decisions is not considered.

Multi-resolution simulation entities need to be capable of simultaneously operating at different levels, while maintaining consistency at each level of abstraction. Challenges involved in assuring consistency across levels of resolutions are explored in (Reynolds et al. 1997). It has been argued that designing cross-resolution models requires principled design strategies that take potential discrepancies into account (Reynolds et al. 1997; Davis 2000). While the observations presented in (Reynolds et al. 1997) are generally applicable, the presented design strategy is overly simplistic. The design involves a Multi-resolution Entity (MRE) that encapsulates a specific number of entities, each with its own state information. Dynamically updating entities in a given resolution level is not addressed in the MRE design scheme. However, exploring Course of Actions (COAs) with different models require seamless update of individual components at selected resolution levels. Extension of the levels of resolution of an MRE is also not addressed. The tight coupling between the resolution levels makes extensibility difficult within the MRE design scheme. The notion of using a consistency enforcer for

Table 1: Taxonomy of Multimodels (Yilmaz and Oren 2004)

Based on	Additional Criteria				Type of multimodel (MM) (Synonyms are represented within parentheses)
Structure of submodels	**Number** of submodels active at a given time			Only one	Single aspect MM (Sequential MM)
				2 or more	Multiaspect MM
	Variability of structure (variability of number of submodels)	**Static**			Static-structure MM
		Dynamic (Dynamic-structure MM) (Variable-structure MM)	Number of submodels	Extensible	Extensible MM
				Depends on model's stage	Multistage MM
			Alterations of submodels	No	Non-mutational MM
				Yes	Mutational MM
					Evolutionary MM
Behavior (activation) of submodels	**Nature of knowledge** to activate submodels	Constraint-driven			Constraint-driven MM (Adaptive MM)
		Pattern-directed (Pattern-directed MM) (Metamorphic MM)	Sub-model selection is cyclic	No	Acyclic MM
				Yes	Cyclic MM
		Goal-directed			Goal-directed MM (Exploratory MM)
	Location of knowledge to activate submodels	Within the MM (Internal activation of submodels)			Active MM (Internally activated MM)
		Outside the MM (External activation of submodels)			Passive MM (Externally activated MM)

consistency management across resolution levels is a useful attribute of the MRE design strategy.

3 MULTIMODELS

Many real-world phenomena can not be modeled by one single model; rather, they require the use of a set of complementary models that together are able to describe the whole process (Ören 1987, 1991; Zeigler et al. 2000; Yilmaz and Oren 2004). Basic definitions and brief explanations of the envisioned multimodel types shown in Table 1 follow:

A multimodel is a modular model that subsumes multiple submodels that together constitute the behavior of a complex multi-phased process. A multimodel encapsulates several aspects of reality (i.e., submodels) in one model. In single aspect models only one aspect of reality can exist at a given time (to be represented by an appropriate submodel) and transitions can occur from one submodel to another one under monitored conditions. As shown in Table 1, there are various design decisions in multimodel design: *Based on the completeness of submodels*, there are two cases: One can know all (2 or more of) the submodels at the beginning i.e., at modeling stage. Yet, there can be emergent conditions where the need for additional submodel(s). *Based on the number of active submodels*, one needs to consider two cases: (1) Only one submodel is active at a given time or (2) two or more submodels are active at a given time. Simultaneous existence of two or more model components would facilitate simulation of multiple aspects of the phenomena under study. Multiresolution models are considered in this paper as a specific case of multiaspect multimodels. Multistage multimodels are viewed as dynamic structure models, in which the number of submodels depends on the stage of the problem.

Based on the location of information necessary for the activation of submodels there are two cases: the necessary information can be (1) within the submodels or (2) it can be external to submodels. The transitions between submodels can be goal-directed (goal directing the submodel transition rule and goal-directed submodel transition mechanism should be specified) or pattern-directed. Nature of information necessary for the activation of submodel(s) entails the selection conditions of a submodel, where there are also two cases: (1) we know the laws of affecting model transition. *Pattern-directed activation* entails a meta-pattern to guide (1) selection of known submodels and (2) request of new submodels corresponding to an interruption of the simulation gaming using the recommender.

4 REQUIREMENTS FOR MRMSM DESIGN

MRMSM encapsulates a set of components/submodels that represent a system or phenomena at different levels of abstraction possibly at multiple phases. Since such an entity can simultaneously operate at multiple levels of resolution, a number of submodels can be active at a time. Shifts in the phase of a problem trigger model updating by changing the active set of entities. For the sake of simplicity we assume that the number of entities needed in a MRMS model is known in advance.

Requirement 1- An MRMSM should be decoupled from the knowledge regarding how its submodels are composed, created, and represented.

This requirement is critical to facilitate that the MRMSM can be configured with one of multiple families of submodels representing an entity at different levels of resolution for a specific stage of the problem. Furthermore, this requirement imposes the constraint that a family of submodels is designed to work together, and that the MRMSM should be independent of how these submodel families are created and composed. This requirement enables isolating implementation of submodels from the MRMSM.

Requirement 2 - The effects of concurrent interactions at multiple levels of resolution must be combined consistently (Reynolds et al. 1997).

MRMSMs constitute multiple submodels that co-exist during simulation. That is, for instance, a low-resolution and high-resolution entity within the model may simultaneously interact with other external entities at corresponding levels of resolution. Consistency problems between different resolution levels might emerge if shared resources are not used mutually exclusively. Concurrent access to such resources needs to be controlled and coordinated.

Requirement 3 – The state of entities at different levels of resolution within a MRMSM should be consistent.

In an MRMSM the state of the aggregated low-resolution entities should be updated as the state of the corresponding high-resolution entities change. This requires definition of multiple one-to-many dependencies between entities so that when one object changes state, its dependents are notified and updated automatically.

Requirement 4 – In an MRMSM the entities should be allowed to alter their behavior when their internal states are changed.

MRMSM behavior depends on the state and/or stage of the problem, and it must change its behavior at run-time depending on that state/problem phase. Therefore, an MRMSM should enable localization of state-specific behavior and partition behavior for different stages. MRMSM should enable the definition of family of related protocols, encapsulate each one, and make them interchangeable to facilitate varying the protocols independently of the entities that are configured with them.

Requirement 5: The constraints regarding when and under what conditions (1) the consistency of the elements of families of submodels in a multiresolution model be enforced and (2) a shift in the stage of the problem be triggered must be independent of the MRMSM.

Corresponding to the time path of the change of a problem should be a time path of the appropriate submodel families. But, the question is what should be the sequence of this shift pattern of models of family of? Or should there be trigger mechanisms indicating when a shift should occur? This requirement suggests that the constraints be separated from the MRMS model and not be intertwined with the entities to facilitate reuse of the submodel families in a different context with different constraints

Requirement 6 - At the time of the model update, the new set of low-resolution and high-resolution entities and their dependent components must be dynamically loaded and linked into the run-time environment of the simulation.

This requires new model and simulator decoupling strategies that avoid persistent connections to facilitate extensibility. Also, the intricate details of complex model instantiation process should be as independent of the MRMSM as possible to enable flexible update.

Requirement 7 - The consistency of entities undergoing (re)placement needs to be preserved.

The event scheduling and simulation protocol need to be restricted or regulated to facilitate interleaving of entity replacement activities with the simulation events.

Requirement 8 - The state of an MRMSM must be constructed, or at least continue from a specific state after an update operation.

This requires externalization through abstraction, state saving, transmission, and reconstruction after the update operation.

Requirement 9 – An MRMSM should have a mechanism for changing the structure and behavior of the model dynamically. In other words, an MRMS model should support its modification a priori.

This requirement suggests that MRMSMs need to provide facilities that establish a self-representation of the model, offer means by which this representation can be updated, and assure that the manipulations to the self-representation influence the behavior of the model.

Requirement 10 – An MRMSM should provide flexibility in terms of defining behavioral resolution to facilitate analysis that are independent of an implementation and programming language specific constructs.

Component-based (Zeigler et al. 2000 – System Entity Structure), function-based (Davis 2000), and attribute-based (Reynolds et al. 1997) approaches to resolution refinement are course-grain approaches. However, flexible what-if analysis requires analysts to select the behavioral resolution at arbitrary levels of granularity without relying on implementation specific constructs. That is, a high-level *behavioral formalism* that enables *fine-grain stepwise refinement* is needed to specify the behavior of an entity at multiple levels of resolution. This facilitates fine-tuned control of a simulation at run-time at different levels of resolution. By having capabilities to control the scope of update, analysts can perform what-if analyses at arbitrary levels.

5 MULTIMODEL SPECIFICATION

While there exist efforts in conceptualizing multiresolution modeling to characterize system behavior at different levels of aggregation and abstraction, a unified and uniform model specification formalism that expedites the realization of the above requirements is missing. The question is what particular model design specification approach is conducive to handling the above requirements. For instance, most of the requirements above need facilities to explicitly specify the (1) transition conditions between model families, (2) cross resolution consistency management, and (3) actions needed for dynamic model replacement and seamless configuration update.

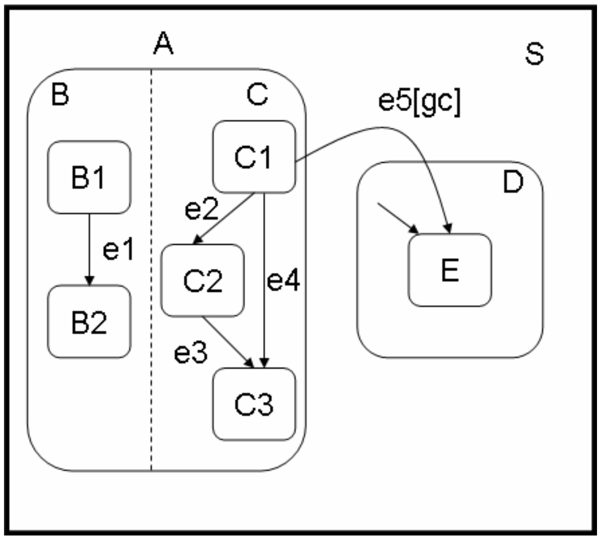

Figure 1: Multimodel Structure

Figure 1 depicts the structure of a multimodel *S*, which has submodels *A* and *D*. At first glance, the diagram can be interpreted as a statechart. However, in our context each node designates a submodel or model component, whereas arcs denote transitions between models. Each transition indicates the event that trigger the activation of a target node, the conditions under which it is activated, and the actions taken to complete the change in the configuration. Submodel A, in Figure 1, can be a multiresolution model with two concurrent levels, each one of which is represented by submodels *B* and *C*. Note that the level of decomposition can be arbitrary by extending the hierarchy. Similarly, submodels *B* and *C* can represent two different aspects of the same subsystem represented by model A. The submodel *D* and alternative configurations of models within *A* facilitate representing distinct stages of the problem. The simulation of a multimodel is based on the interpretation of the model reachability structure depicted in Figure 2. The structure is an *and-or graph* that has two types of nodes. The *and-nodes* require every descendant node to be active. On the other hand, only one descendant node of an or-node is active, while the rest are dormant. In Figure 2, the model S is represented as an *or-node*, as either *A* or *D* is active at any point in time during the simulation. The submodel *A*, however, is an *and-node*, as both *C* and *D* can be active. The submodels *B* and *C* are *or-nodes*. This requires either *B1* or *B2* (basic nodes of *B*) to be latent within submodel B. Similarly, at most one of the submodels of C can be active at any point of time.

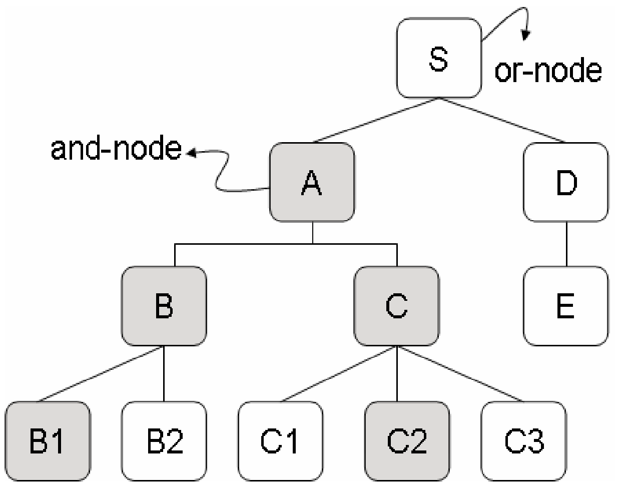

Figure 2: Multimodel And-Or Graph

Definition: For every pair of nodes, their **scope** is the lowest common ancestor, e.g., *scope (B1, E)= S.*

Definition: A pair of nodes is **orthogonal** if their scope is an or-node.

Definition: The **configuration** of a multimodel S is a minimal subtree *CFG(S)* that satisfies

- $S \in CFG(S)$
- If $A \in CFG(S)$ and A is an or-node then *CFG(S)* contains exactly one of the submodels of A.
- If $A \in CFG(S)$ and A is an and-node then *CFG(S)* contains all of the submodels of A.

The **basic configuration** is the set of basic nodes in a configuration, which uniquely identifies a multi-model configuration. Simulating a multimodel requires the definition of its state and semantics of the transitions that facilitate the generation of the its behavior. The state of a multimodel consists of the following elements.

- CFG - A configuration with respect to the multimodel under consideration.
- TR - History information – last visited submodel.
- CND- Assignment of all atomic conditions.
- TM – Set of all scheduled timeout events at the current state.
- E – The subset of atomic events.

A transition takes place to change the configuration of a multimodel to switch to a new stage, insert new set of submodels, update the level of resolution and fidelity. A transition, (*M1, <eventExpr, condExpr, actions>, M2*), from model M1 to M2 is triggered or enabled when

- $M1 \in X.CFG$
- $eventExpr \in (X.E \cup X.TM)$
- $X.CND \prec condExpr$

Once a transition is enabled, its execution results in the change of the configuration of the multimodel. To specify the change, we need to introduce the semantics of a transition. Given a transition $t = (R, < E, C, A >, P)$ at a status *X,* let $S = scope(R, P)$. The set of nodes that are exited as a result of the transition is defined as $EXIT(t) = X.CFG(S) - \{S\}$. The set of nodes that depict the submodels that are activated is defined as $ENTRANCE(t) = CFG(S, P)$. This set of nodes refers to the configuration emerging from S (excluding S) and then including *P* and then expanding by initial node labels. When an enabled transition transforms the configuration of a multimodel the following steps take place.

- A new construction is created by replacing the source submodel M1 with the target submodel, M2.
- The entrance conditions associated with M1 (e.g., *in(M1)*) become false, whereas the entrance conditions associated with M2 (e.g., *in(M2)*) become true.
- The events *exit(M1)* and *enter(M2)* are added to the new status.
- The actions specified in the transition are executed. These actions need to address the requirements listed in section 4.

Designing multimodels based on the above formalism in terms of communicating and interacting objects and classes that are customized to deal with the requirements listed in section 4 is a significant challenge. The next section shows how the principled use of a number of design patterns may

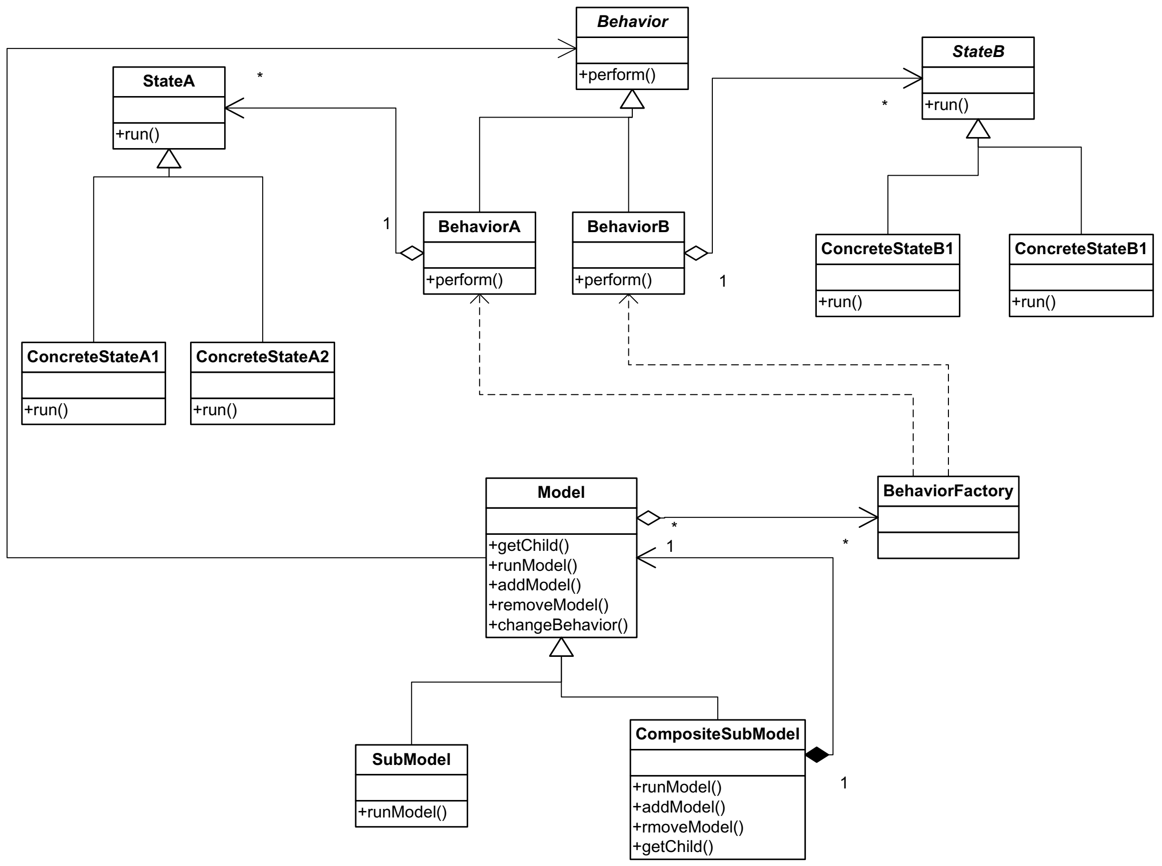

Figure 3: A Partial and Preliminary Design Strategy for Multimodels

facilitate the realization of a selected requirement with the above formalization in mind

6 DESIGN PRINCIPLES FOR MRMSM

The specification formalism discussed in the previous section requires composing objects into tree structures to represent part-whole hierarchies. Furthermore, each submodel needs to be treated uniformly regardless of whether it is a composition (e.g., multiresolution) or an individual submodel. Multiresolution or multistage models can be elements of a coarse granular models, which operate at multiple levels of resolutions and stages themselves. Hence, it is critical to make sure that a multimodel simulator that uses the strategy discussed in section 5 ignores the differences between compositions of submodels and submodels or individual components. The design structure between *Model*, *CompositeSubModel*, and *Submodel* aims to achieve this objective. Specifically, the *Model* defines the interface for objects in the composition, which realize the default behavior. The *Submodel* represent those nodes in the composition that have no children nodes. The *CompositeSubmodel* defines behavior for submodels that have multiple levels of resolution or aspects.

A significant requirement as discussed in the previous section is to *decouple MRMS from the knowledge regarding how its submodels are composed, created, and represented*. The *BehaviorFactory* component facilitates the creation of families of related or dependent submodels or model components without explicitly specifying the concrete realizations. For instance, referring to Figure 1, the structure and behavioral representation of submodel *B* can be encapsulated within the *BehaviorB* component of Figure 3. As a result, the MRMS model (i.e., *Model*) is independent of how submodel *B* is instantiated. This information is encapsulated within *BehaviorB*. That makes it possible configure the MRMS model with one of several submodel families. Furthermore, since the related set of submodels and their components are designed to be used together, the decoupling of MRMS from *BehaviorB* enables enforcing this constraint. Finally, this configuration makes it easy to exchange submodel families easier.

The existence of *State* components associated with *Behavior* entities aims to allow a submodel to alter its behavior when its internal configuration changes (i.e., *sub-*

model C starts using *C2* as opposed to *C1* after the configuration of the multimodel is updated). As a result, the submodel will appear to change its subcomponents. That is, the submodels behavior depends on the configuration of the multi-model, and the submodel needs to change its behavior at run-time depending on the new configuration. The consequence of this design strategy is that configuration-specific behavior and partitions behavior for different configurations. This makes the model transition concept presented in section 5 explicit.

7 MULTISIMULATION WITH MRMSM

Multisimulation (or multisim, for short) is simulation of several aspects of reality in a study (Ören 2001). It includes simulation with multimodels, simulation with multiaspect models, and simulation with multistage models. *Simulation with multimodels* allows computational experimentation with several aspects of reality; however, each aspect and the transition from one aspect to another one are considered separately. (In special cases, multimodels can be metamorphic models or evolutionary models). *Simulation with multiaspect models* (or multiaspect simulation) allows computational experimentation with more than one aspect of reality simultaneously. This type of multisimulation is a novel way to perceive and experiment with several aspects of reality as well as exploring conditions affecting transitions. While exploring the transitions, one can also analyze the effects of encouraging and hindering transition conditions. *Simulation with multistage models* allows branching of a simulation study into several simulation studies; each branch allowing to experiment with a new model under similar or novel scenarios.

Requirement 11 – Multisimulation with multimodels, multiaspect models or multistage models needs mechanisms to decide when and under what conditions to replace existing models with a successor or alternative.

Staging considers branching to other simulation studies in response to scenario or phase change during experimentation. Graphs of model families may facilitate derivation of feasible sequence of models that can be invoked or staged. More specifically, a graph of model families can be used to specify alternative staging decisions. Each node in the graph depicts a model, whereas edges denote transition or switching from one model to another. Figure 4 depicts the components of the abstract architecture of a possible multisimulation engine.

A meta-simulator is a scheduler that generates staged composition of models by traversing the model stage graph and coordinates their simulation and staging within distinct simulation frames. Each frame simulates a distinct subset of models derived from the model stage graph. Note however, not all staged compositions are feasible or useful. Hence, the meta-simulator needs to consult with the model recommender before model staging to determine if emergent trigger or transition condition in the simulation is consistent with the precondition of the model to be staged. More than one model in a family can qualify for staging; in such cases separate simulation frames need to be instantiated to accommodate and explore plausible scenarios. Given a collection of models (or more generally, family of models), a stage graph can be generated automatically by an optimistic approach that connects every available node (model) to every other node within the domain of problem. The edges in a model stage graph denote plausible transitions between models as the problem shifts from one stage to another.

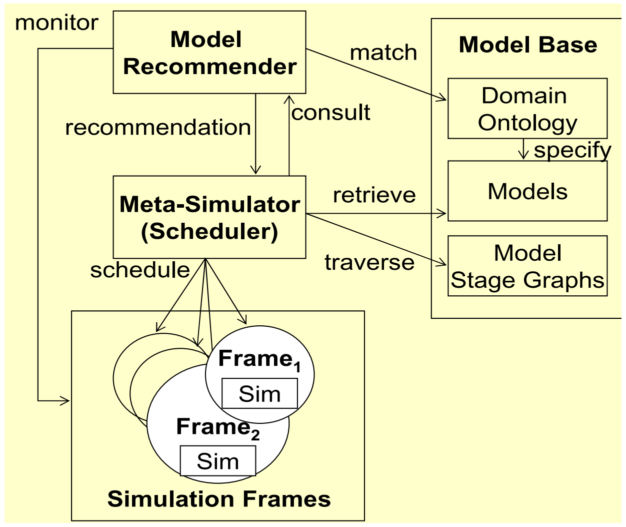

Figure 4: Components of the Multisimulation Engine

Model recommendation in multisimulation can simply be considered as the exploration of the model staging space that can be computed by reachability analysis of the graph. There are two modes for the usage: (1) Offline enumeration of paths using the graph and performing a staged simulation of each model in sequence one after the other, unless a model staging operation becomes infeasible due to conflict between the transition condition and the precondition of the successor model and (2) run-time generation of potential feasible paths as the simulation unfolds. In both cases, an online model recommender plays a key role to qualify a successor model. The first case requires derivation of sequence of models using a traversal algorithm. The edges relate families of models. Therefore, the actual concrete models, the preconditions of which satisfy the transition condition need to be qualified, since transition to some of these model components may be infeasible due to conflict between a candidate model and inferred situation. Identifying such infeasible sequences is computationally intractable; otherwise, it would have been possible to determine if the conjunction of two predicates is a tautology by using a polynomial time algorithm.

Experience in component-based simulation paradigm, however, indicates that for most model components preconditions are simple. Hence, it is possible to eliminate some models that violate the transition condition. For the remaining possible transitions it is possible to select one of the three strategies: (1) omit all difficult qualification conditions, (2) decide on an edge by edge basis which specific models of a model family to include, and (3) include all difficult edges. Omitting all difficult associations between transitions and model preconditions is conservative. This strategy excludes all infeasible models. The cost is the exclusion of some feasible edges. Hand-selecting those associations between transition conditions and models facilitate inclusion of feasible models. Yet, the costs involved with this level of accuracy are the potential human-error and effort needed to filter out infeasible models. Choosing to include all difficult associations is liberal, in that it ensures inclusion of all feasible models. The cost is the inclusion of some infeasible models, hence the inclusion of some undesirable staged compositions that enforce models to be simulated even when their qualification conditions are violated. Nevertheless, it is possible to screen out such models using an online model recommender.

Candidate models and associated simulations can be maintained by *focus points*. A focus point manages branch points in the simulation frame stack. Suppose that a goal instance (i.e., stage transition condition) is at the top of the stack. If only a single model qualifies for exploration, then it is pushed onto the stack. Yet, if more than one model matches the condition, a simulation focus point is generated to manage newly created simulation branching (discontinuity) points, each one of which have their own contexts. When a path is exhausted, the closest focus point selects the next available model to instantiate the simulation frame or return to the context that generated the focus point. As simulation games are explored, a network of focus points is generated. Determining which focus point should be active at any given time is the responsibility of the meta-scheduler. When more than one model is qualified, then scheduler needs to decide which one to instantiate. Control rules can inform its decision. Three steps involve in deploying a new simulation frame in such cases: matching, activation, and preference. The matching step should both syntactically and semantically satisfy the request. The activation step involves running a dynamic set of rules that further test the applicability of models with respect to contextual constraints. Finally, the preference steps involve running a different set of rules to impose an activation ordering among the active frames.

8 CONCLUSIONS

The purpose of this paper is to (1) reiterate the rationale for using novel simulation modeling such as multimodels, and multiresolution modeling, (2) present a comprehensive taxonomy of multimodels, and (3) layout fundamental requirements in developing simulation modeling infrastructures that enable multi-model design. The paper presented a set of such requirements along with preliminary design precepts that have potential to realize emergent challenges. Furthermore, a graph-based multimodel specification formalism is presented along with its basic syntax and semantics to facilitate developing simulators for such multimodels. The notion of multisimulation is introduced to enable exploratory simulation using various types of multimodels.

REFERENCES

Bankes, S. 1999. Tools and techniques for developing policies for complex and uncertain systems. In *Proc. of the National Academy of Sciences, Colloquium*, Vol. 99, suppl. 3, pp7263-7266.

Davis, K. P. 2000. Exploratory analysis enabled by multiresolution, multiperspective modeling. In *Proceedings of the 2000 Winter Simulation Conference*. ed. Jeffrey A. Joines, Russell R. Barton, K. Kang, and Paul A. Fishwick., 2000. Available from RAND as RP-925.

Davis, K. P and Bigelow, H. J. 2002. *Motivated metamodels: Synthesis of cause-effect reasoning and statistical modeling*, The RAND Corporation, Santa Monica, CA.

Davis, K. P. 1988. *The role of uncertainty in assessing the NATO-PACT central region balance, RAND N-2839,* The RAND Corporation, Santa Monica, CA, 1988. (also available in General Accounting Office, NATO-Warsaw Pact Conventional Force Balance, GAO/NSIAD-89-23B, 1988.

Fishwick, P. and Zeigler P. B. 1992. A multimodel methodology for qualitative model engineering. *ACM Transactions on Modeling and Simulation*, 2(1) pp. 52-81.

Fujimoto, R., D. Lunceford, E. Page, A. and Uhrmacher L. 2002. Technical report of the Dagstuhl-seminar on grand challenges for modelling and simulation.Available via <http://www.informatik.uni-rostock.de/~lin/GC/index.html> [last accessed March 14, 2007].

Guizzo E. (2004). “Software and the city,” *IEEE Spectrum*, Available via <http://www.spectrum.ieee.org/WEBONLY/wonews/dec03/1203urb.html> [last accessed March 14, 2007].

Reynolds, F. P., Natrajan, A., Srinivasan S. 1997. Consistency maintenance in multiresolution simulation. *ACM Transactions on Modeling and Simulation*, 7(3) pp. 368-392.

Ören, T.I. 1987. Model update: A model specification formalism with a generalized view of discontinuity," In *Proceedings of the Summer Computer Simulation Conference*, Montreal, Quebec, Canada, 1987 July 27-30, pp. 689-694

Ören, T.I. 1991. Dynamic templates and semantic rules for simulation advisors and certifiers," In *Knowledge-Based Simulation: Methodology and Application*, P.A. Fishwick and R.B. Modjeski (Eds). Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 53-76.

Ören, T.I. (2001 – Invited contribution). Towards a Modelling Formalism for Conflict Management. In: Discrete Event Modeling and Simulation: A Tapestry of Systems and AI-based Theories and Methodologies. H.S. Sarjoughian and F.E. Cellier (eds.), Springer-Verlag, New York, pp. 93-106.

Takahashi, K., Yugi, K., Hahimoto, K., Yamada, Y., Pickett, C., and Tomita, M. 2002. Computational challenges in cell simulation. *IEEE Intelligent Systems* 17:64-71.

Yilmaz, L. and Ören, T.I. 2004. Dynamic model updating in simulation with multimodels: A taxonomy and a generic agent-based architecture. In *Proceedings of SCSC 2004 - Summer Computer Simulation Conference*, July 25-29, 2004, San Jose, CA., pp. 3-8.

Yilmaz, L. and Tolk A. 2006. Engineering ab initio dynamic interoperability and composability via agent-mediated introspective meta-simulation. In *Proceedings of the 2006 Winter Simulation Conference*, pp. 1075-1082. Monterey, CA. December 3-6 2006.

Zeigler, P. B. and Ören, T.I. 1986. Multifaceted. multiparadigm modeling perspectives: Tools for the 90's. In *Proceedings of the 1986 Winter Simulation Conference*. pp. 708-712.

Zeigler, B. P., Praehofer, H. Kim, G. T. 2000. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*, Academic Press.

AUTHOR BIOGRAPHIES

LEVENT YILMAZ is assistant professor of Computer Science and Software Engineering in the College of Engineering at Auburn University. Dr. Yilmaz earned his Ph.D. and M.S. degrees from Virginia Tech. His research interests are on advancing the theory and methodology of simulation modeling, agent-directed simulation to explore dynamics of socio-technical systems, organizations, and human/team behavior, and education in simulation modeling. Dr. Yilmaz is a member of ACM, IEEE Computer Society, Society for Computer Simulation International, and Upsilon Pi Epsilon.
URL: <http://www.eng.auburn.edu/~yilmaz>

ALVIN LIM is associate professor of Computer Science and Software Engineering in the College of Engineering at Auburn University. Dr. Lim received his Ph.D. from University of Wisconsin-Madison. His research interest are self-organizing networks, wireless mobile networks, high performance networks, mobile computing and databases, reliable and dynamically reconfigurable distributed systems, complex software development, parallel processing, and performance measurements and analysis. In self-organizing networks, he is developing new techniques and protocols that allow network nodes to form spontaneous networks, assemble the network themselves, dynamically adapt to device failure and degradation, manage movement of network nodes, and react to changes in task and network requirements. In wireless mobile networks, he is investigating adaptive communication protocols for large-scale mobile information systems and the performance implications of these protocols.
URL: <http://www.eng.auburn.edu/~lim>

SIMON BOWEN is a graduate student at Auburn University. He earned his B.S at Auburn and will be getting his M.S. in Computer Science in 2007. He is currently doing research with Dr. Yilmaz and Dr. Lim in multi-resolution multi-stage modeling and design. His research interests are in software design and architecture and simulation modeling.

TUNCER ÖREN is a professor emeritus of Computer Science at the University of Ottawa. His current **research** activities include (1) advanced M&S methodologies such as: multimodels, multisimulation, and emergence; (2) agent directed simulation; (3) cognitive simulation (including simulation of human behavior by fuzzy agents, agents with dynamic personality and emotions, agents with perception, anticipation, and understanding abilities); and (4) reliability and quality assurance in M&S and user/system interfaces. He has **contributed** in Ethics in simulation (http://www.scs.org/ethics/), M&S Body of Knowledge, and multilingual M&S dictionaries http://www.site.uottawa.ca/~oren/SCS_MSNet/simDic.htm . Currently he is a member of the Board of Directors, AVP for Ethics, and Founding Director of the McLeod Modeling and Simulation Network of the Society for Modeling and Simulation International (SCS). He has over 350 **publications,** mostly on M&S (some translated in Chinese and German) and has been active in over 350 **conferences** and seminars held in 29 countries. He is **recognized** by IBM Canada as a Pioneer of Computing in Canada (2005).
URL: <http://www.site.uottawa.ca/~oren/>