Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

USING ONTOLOGIES FOR SIMULATION MODELING

Perakath Benjamin
Mukul Patki
Richard Mayer

Knowledge Based Systems, Inc.
1408 University Drive East
College Station, TX 77840

ABSTRACT

Ontological analysis has been shown to be an effective first step in the construction of robust knowledge based systems. However, the modeling and simulation community has not taken advantage of the benefits of ontology management methods and tools. Moreover, the popularity of semantic technologies and the semantic web has provided several beneficial opportunities for the modeling and simulation communities of interest. This paper describes the role of ontologies in facilitating simulation modeling. It outlines the technical challenges in distributed simulation modeling and describes how ontology-based methods may be applied to address these challenges. The paper concludes by describing an ontology-based solution framework for simulation modeling and analysis and outlining the benefits of this solution approach.

1 INTRODUCTION & BACKGROUND

This paper describes the role of ontologies in simulation modeling. We will first introduce ontologies and outline the motivations for ontology development and the technical issues in simulation that would benefit from the application of ontology-based approaches. Next we describe the role of ontologies in the simulation model development process. We will then describe the role of ontology-driven methods in addressing key problems associated with distributed simulation modeling, and describe an ontology-driven framework for simulation modeling. We conclude with a summary of the significance and benefits of this research.

1.1 Nature of Ontology

An *ontology* is an inventory of the kinds of entities that exist in a domain, their salient properties, and the salient relationships that can hold between them (Benjamin et. al 1995). Every *domain*—typically, in this context, some piece of the actual world such as a manufacturing system, a university, a business—has its own ontology, which we refer to simply as a *domain ontology*. In a domain ontology, we define various *kinds* of objects (e.g., **tools** and **employees**), *properties* (e.g., **being made of metal** or **having three children**), and *relations* between kinds and their instances (e.g., **part of** or **married to**).

Ontology development focuses on extracting the essential nature of the concepts in any domain and representing this knowledge in a structured manner. The construction of an ontology differs from traditional information capture activities in the *depth* and *breadth* of the information captured. Thus, an ontology development exercise will expand beyond asserting the mere *existence* of relations in a domain; the relations are "axiomatized" within an ontology (i.e., the *behavior* of the relation is explicitly documented). Ontology development is motivated not so much by the search for knowledge for its own sake (as, ideally, in the natural and abstract sciences), but by the need to understand, design, engineer, and manage such systems effectively.

Important research issues in ontology management include determining the appropriate scope and granularity of ontologies, standardization and automation of the process of relating different ontologies, use of natural language processing for ontology comparisons, and the use of ontologies as a basis for defining model repositories (Fishwick et. al 2004).

1.2 Benefits of Ontology Development

One of the most important aspects of the general development and use of an ontology acquisition method is the accumulation of a wide range of domain ontologies. Generally, inefficiency is among the greatest problems in knowledge management. Redundant effort is expended capturing or recreating information that has already been recorded elsewhere. Rather than having to encode information multiple times in different application settings, the idea is to develop *ontology libraries*; i.e., large

revisable knowledge bases of structured, domain specific, ontological information in which can be put several uses for multiple application situations.

Ontologies are important for intelligent systems development for several reasons, including the following:

1. Ontological analysis has been shown to be an effective first step in the construction of robust *knowledge based systems* (Hobbs 1987). Current and future intelligent systems (including modeling and simulation applications) will take advantage of knowledge based and expert systems technology.
2. Ontologies will be required to develop standard, re-usable application and domain *reference models*. Standard ontology reference models such as the Suggested Upper Merged Ontology (SUMO) are available on the Internet.
3. Ontologies are at the heart of software systems that facilitate *knowledge sharing*. The importance of knowledge sharing is evidenced by the large body of research directed toward the development of tools and methods to support a knowledge sharing approach to integration (Neches 1991; Gruber 1992; Benjamin et. al 2006a).

Some benefits of having and ontology-driven approach to simulation, as discussed by Millet et al. (2005), are that this approach allows one to discover concepts and relationships by either browsing through small ontologies or querying large ontologies using customized query languages visualize the structure classes, objects, concepts and relationships using commercially available visualization tools take advantage of ontological descriptions in all phases of simulation modeling improve the efficiency of multi-modeling or component-based modeling, if ontologies of all components are available. The latter two have been discussed in more detail in this paper.

1.3 Motivations for Using Ontologies in Simulation Modeling

Knowledge-intensive approaches to simulation require the acquisition, storage, maintenance, and application of highly structured knowledge, including ontologies. Ontologies are useful across the simulation modeling and analysis lifecycle, particularly in the problem analysis and conceptual model design phases. With increased use of distributed intelligence approaches to simulation modeling (distributed simulation, federated simulation, agent based simulation, etc.), ontologies play a critical role in simulation integration and simulation composability. In particular, ontologies are essential in facilitating simulation model interoperability, composition, and information exchange at the semantic level.

One of the key motivations for distributed, federated simulation modeling is to allow for the decomposition of the overall system model into smaller, more manageable components, and to distribute the model development effort among different organizations or functional groups (Benjamin et. al 2006a). Once the component simulation models have been developed, there is a need for mechanisms to assemble a simulation model of the entire target system in a manner that the "whole (system) = sum of its components." The DoD's past and ongoing investments in distributed simulation and the High Level Architecture (HLA) have been motivated significantly by this idea. A key technical challenge is modeling and simulation *composability* (from a set of independently developed components). "Composability is the capability to select and assemble simulation components in various combinations into simulation systems to satisfy specific user requirements" (Petty and Weisel 2003). The components to be composed are often drawn from a repository. The components may vary from "legacy code and data" all the way to well-defined software components or models with "standards-based" protocols (such as the HLA protocols) and Application Programming Interfaces (API's). Composability enables users to combine, recombine, and configure or reconfigure components in numerous ways to satisfy their diverse needs and requirements. There are two forms of composability: *syntactic* and *semantic*. Syntactic composability deals with the compatibility of implementation details such as parameter passing mechanisms, external data accesses, and timing mechanisms. Semantic composability, on the other hand, deals with the validity and usefulness of composed simulation models (Petty and Weisel 2003).

Ontologies play an important role in addressing three important challenges associated with modeling and simulation composability and interoperability at the semantic level (Benjamin et. al 2006a): (1) semantic inaccessibility, (2) logical disconnectedness, (3) consistency maintenance, and (4) modeling at multiple levels of abstraction.

We now describe the role of Ontologies in Simulation Modeling in terms of (1) the simulation modeling process (Section 2) and (2) distributed simulation modeling (Section 3).

2 ROLE OF ONTOLOGIES IN THE SIMULATION MODELING PROCESS

This section describes the role of ontologies in facilitating the simulation modeling process. The simulation modeling process, including the simulation model development process and the role of ontological analysis in facilitating this process, is illustrated in Fig. 1.

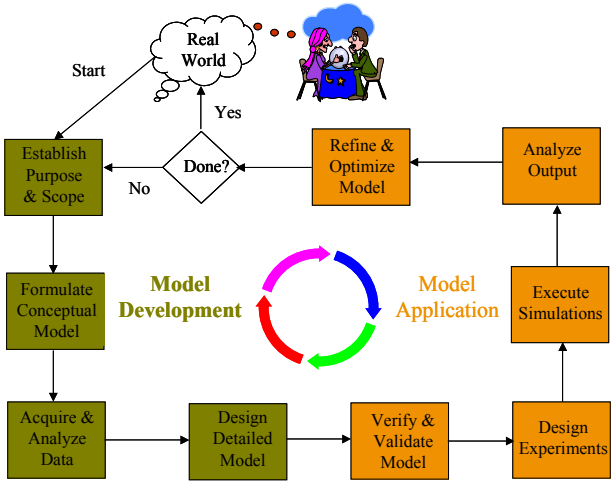

Fig. 1: The Simulation Modeling Process

2.1 Establish Purpose and Scope

Simulation models are often designed to address a set of modeling objectives or to answer a set of questions. An important first step in simulation modeling is to define the purpose of the model. This activity involves several related activities:

- *List of problem symptom.* The subject matter expert often describes a problem in terms of a list of observed symptoms or areas of concern. The desire is to identify the cause of these symptoms and to suggest remedies.
- *Statement of system design or redesign goals.* Often the subject matter expert specifies the objectives of the study in terms of a specific question that needs to be answered, or, alternatively, specifies explicit goals to be met. For instance, a manufacturing system manager might ask the question "How can I streamline my production?" or state a goal: e.g., "I need to achieve a utilization level of at least 80% on all my machines."
- *Constraints on possible solutions to the problem.* The subject matter expert, based on past experience with similar situations, often suggests a variety of possible alternative solutions that must be explored. For example, a manufacturing manager who would like to increase production rate may, because of a budgetary constraint, be unwilling to invest in new machines, but may instead be able to hire additional labor.

Ontologies will help facilitate the above tasks as described below.

- Providing a mechanism to interpret and understand the problem descriptions. Subject matter experts often use specialized terminology to describe symptoms and problems. Domain ontologies help with the unambiguous interpretation of the problem statements and in precisely conveying information about the problem to the simulation modeler.
- Harmonizing statements of objects that are described from multiple perspectives (often, this is a non-trivial task because of terminological differences and the lack of explicit descriptions of the semantics of different terms and concepts).
- Unambiguously interpreting limiting constraints that need to be addressed relative to accomplishing project goals.

2.2 Formulate Conceptual Model

The conceptual modeling process comprises the following inter-related activities: Acquire and Analyze System Descriptions; Determine and Classify Modeling Objectives, Determine Object Roles and Boundary and Level of Detail, and Determine Model Structure and Logic. The following paragraphs describe these activities and the role of ontologies in enabling these activities.

2.2.1 Acquire and Analyze System Descriptions

The conceptual modeling process usually starts with the acquisition of system descriptions from a subject matter expert, traditionally acquired either orally or from written descriptions. However, these types of descriptions are usually unstructured and ambiguous, resulting in lengthy consultations between the analyst and domain expert. There are different systems engineering methods and tools (IDEF0, IDEF3, IDEF5, etc.) that facilitate the structuring and organization of information while it is being acquired from the subject matter expert [www.idef.com]. For example, IDEF3 provides a language and procedure for process knowledge capture that is natural to subject matter experts in a given environment. This knowledge engineering process includes the capture of facts about the objects that participate in a process, about object state transitions, and the precedence and causality relations between processes and events. It is important (during this initial conceptual modeling activity) to distinguish between *descriptions* that record an agent's *perceptions* of how a system works and *models* that provide an *idealization* of how a system works (e.g., simulation models).

A process of abstraction is employed to map domain system descriptions to conceptual models as depicted in Fig. 2.

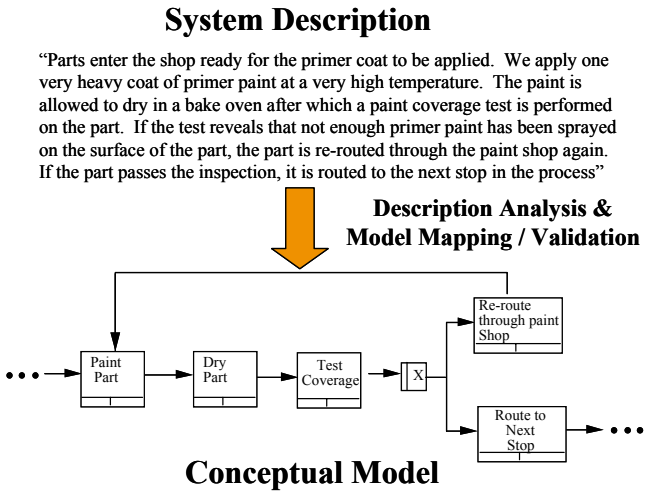

Fig. 1: Mapping System Descriptions to Conceptual Models

The nature of descriptions is that they are *partial*. Therefore, an important aspect of the conceptual modeling process is determining the adequacy or completeness of an existing description relative to queries or requests for decision data. Consistency checking is the activity of incrementally verifying that assertions of facts made at every stage of the description capture do not contradict the facts established at earlier stages. Completeness checking ensures that the incremental information acquired during description capture is sufficient to enable answering the questions that the model is being designed to answer. Ontologies play an important role in identifying inconsistencies and incompleteness in a domain system description. For example, ontologies may be used to interpret the descriptive information about objects and the constraints gathered from subject matter experts and descriptive data collections (Fig. 3).

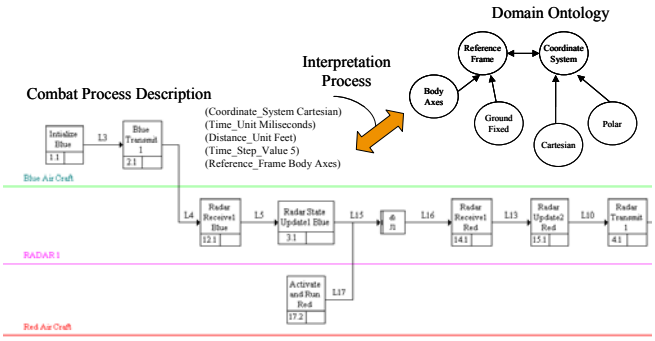

Fig. 3: Using Ontologies to Analyze and Validate System Descriptions

Fig. 3 illustrates the idea of using an ontology of Air-to-Air Combat (AAC) to interpret and disambiguate information about the Coordinate System (Body Axes vs. Ground Fixed) and the Reference Frame (Cartesian vs. Polar) provided in the system descriptions and the evolving conceptual models of the AAC combat behaviors.

2.2.2 Determine and Classify Modeling Objectives

An important step in the development of the conceptual model is to determine the specific goals of the simulation study based on the "demand for decision data" given by the domain expert. Often the question statements are captured in an unstructured form. Consequently, there is a need to refine it further in order to extract the specific goals of the study. The process of refinement, which is performed by the analyst, is based primarily on his interpretation of the query statement and a reasoning mechanism to map this interpretation into a specific goal(s). This reasoning process uses knowledge about the relationships and constraints within the system description (these are interpreted using a domain ontology). During the course of such reasoning, the analyst often needs additional information or clarifications from the domain expert in order to clearly identify the user requirements. The modeling objective plays a key role in determining the structure of the model to be developed, as well as in establishing the boundaries of the system to be analyzed, the level of detail to be included in the model, and the performance measure(s) to be estimated from running the simulation model.

2.2.3 Determine Object Roles, Boundaries, and Level of Detail

The following tasks are carried out once the specific goals of the analysis have been established:

- *Establishment of model boundaries.* One of the early activities in developing the conceptual model is the selection of the part of the system to be studied. The choice of boundaries is very closely linked to the specific goals of the analysis. This decision about boundaries is an important step since it gives perspective to the entire simulation study. As it turns out, a description is partial, including only those portions of the system which are of special interest to the domain expert. While this might provide clues as to the boundaries chosen for the model, it might also become necessary to either ask for additional information about the system or to exclude parts of the description from the boundaries. The reasoning process in mapping the analysis goals to the boundaries is based on knowledge that is encoded within common-sense and domain ontologies.
- *Establishment of level of abstraction.* Once the boundaries of the model have been chosen, the

analyst proceeds to select the level of abstraction to be used in modeling the system elements that are included within the boundaries. This activity is significantly impacted by the goals of the analysis. A simple rule for determining the appropriate abstract level is "Include only those elements of a system that are relevant to the objective, and do so at the highest level of abstraction." Domain ontologies help in mapping the information contained within the description to the modeling objectives.

- *Identification of model objects and roles.* This step refers to the selection of objects from the description to be included in the simulation model, and the specific role that these objects will play in the model. Common simulation object roles include entities ("flow objects"), resources ("agents"), and queues ("waiting spaces"). Determining the correct simulation roles for objects requires in-depth knowledge of the domain (accessible through a domain ontology) and simulation modeling expertise.

2.2.4 Determine Model Structure and Logic

Model structure and logic refer to the characterization of the relations between activities in the model. An activity represents the dynamic behavior that comes about when objects interact with each other. The model structure refers to the characterization of this dynamic behavior. For instance, if an activity is a manufacturing process, then its characterization will relate to specifying its processing time, which qualifies the behavior that occurs when a part is processed on a machine. There are two types of model logic: *flow logic* and *decision logic*. Flow logic is the specification of the flow path of all the objects through the system. Decision logic refers to the set of methods used to choose between alternative state transitions which characterize the dynamic behavior of the system. For example, the specific scheduling rule used to load a machine with parts in a manufacturing system will be part of the decision logic for that system.

Ontologies play an essential role in unambiguously interpreting the information contained within the system descriptions in order to correctly understand the flow logic and decision logic within the real world processes being modeled. Structured representations (such as IDEF0 and IDEF3) help with reducing the ambiguity inherent in natural language system descriptions. For example, the temporal information captured using the IDEF3 precedence links is used to determine the flow logic of the simulation model. In a similar manner, the IDEF3 junction constructs are used to clearly describe the decision logic between multiple process steps.

The IDEF3 modeling constructs associated with "unit of behavior" boxes are related to the dynamic behavior of the objects which are contained within them. The decision logic that is associated with IDEF3 junctions can be of three kinds: probabilistic, conditional, or deterministic. Other general system information such as the part routings, schedules, distance between stations, and starting conditions, needs to be incorporated into the model structure and logic wherever possible. If such information is not included in the description, it may have to be gathered with the help of the domain expert or may be found in the query statement itself.

2.3 Collect and Analyze Data

Data collection and analysis involves several inter-related activities, including Data Source Identification, Data and Text Mining, Focused Statistical Data Analysis (Estimating Input Distributions, Covariance Analysis, etc.), and Validation of Data Analysis Results.

Ontologies play an important role in assisting with the data and text mining and interpreting the results of the data and text mining. Our experience indicates that ontologies provide a deep level of understanding of the data being analyzed [meanings of the names of the fields, process history logs (natural language descriptions), etc.]. Ontology-assisted text and data mining provides valuable insights about the data, leading to higher initial quality simulation model specifications.

2.4 Design Detailed Model

The detailed model design comprises three major types of activities: refining and detailing simulation model objects (detailed object design), detailing and refining simulation model structure and logic (flow logic and decision logic design), and preliminary model validation.

The primary role of ontologies in detailed model design is in the detailed analysis of information about objects and constraints. This involves mapping the simulation model constraints to specifications of real world constraints that are found within the domain system descriptions.

Table 1 summarizes the role of ontologies our the simulation model development process.

3 THE ROLE OF ONTOLOGIES IN DISTRIBUTED SIMULATION MODELING

Simulation interoperability is a major challenge with respect to information representation in modeling and simulation (Lacy et. al 2004). This section describes the role of ontologies in facilitating distributed simulation modeling. We organize the description of the research in this section in terms of four different aspects of distributed simulation: simulation application integration, composable simulation modeling, and simulation modeling at multiple levels of abstraction.

Table 1: How Ontologies Enable the Simulation Model Development Process

Simulation Modeling Activity	Activity Description	Role of Ontologies
Establish Purpose & Scope	Capture needs, questions, objectives. Integrate across multiple perspectives. Map organization / mission goals to simulation goals.	Terminology harmonization to enable shared and clear understanding.
Formulate Conceptual Model	Validate system descriptions. Identify model boundaries. Identify level of modeling abstraction. Identify model objects and roles. Determine model structure and logic.	Ontology knowledge is used to determine the unambiguously differentiated abstraction levels. Ontological analysis helps to map system objects to model objects and to identify appropriate object roles. Ontological analysis helps reason with system constraints to facilitate determination of model logic.
Acquire and Analyze Data	Identify data sources and data dictionaries. Perform data and text mining. Perform statistical analyses, data reduction (distribution fitting, etc.).	Ontologies play an important role in detailed data analysis, especially in disambiguating terminology and interpreting text data descriptions.
Design Detailed Model	Refine, detail, and validate model objects. Refine, detail, and validate model structure and model logic.	Ontologies will facilitate detailed analysis of objects and constraints including mapping the simulation model constraints to evidence / specifications of real world constraints in the domain descriptions.

3.1 The Role of Ontologies in Simulation Application Integration

A complex representation (e.g., a simulation model or a finite capacity scheduling model) carries the information by virtue of some established, systematic connection between the components of the representation and the real world. It is this connection that determines the semantic content of the data being represented. Typically, however, the semantic rules of a representation system for a given application and the semantic intentions of the application designers are not advertised or accessible to other agents in the organization. This makes it difficult for such agents to determine the semantic content of a database.

This problem manifests itself superficially in the forms of unresolved ambiguity (as when the same term is used in different contexts with different meanings) and unidentified redundancy (as when different terms are used in different contexts with the same meanings). But these are just symptoms; the real problem is how to determine the presence of ambiguity and redundancy in the first place. That is, more generally, how is it possible to access the semantics of process-oriented data across different contexts? How is it possible to fix their semantics objectively in a way that permits accurate interpretation by agents outside the immediate context of this data? Without this ability, the kind of coordination between multiple applications and sub-systems necessary for effective enterprise integration is not possible.

(Benjamin, et. al 2005) describes an approach to address these problems that determined inter-application information flow requirements via analysis of application method ontologies and application software tool ontologies. This approach uses the concept of a vendor-neutral process language as the basis for building translators between process-oriented applications (scheduling, simulation, and optimization). It has similar motivations as the Process Specification Language (PSL) standard under development by the National Institutes of Standards Technology (NIST). In our recent adaptation of this research, we used the IDEF3 Process Modeling Language as the vendor-neutral process language [www.idef.com]. Fig. 4 shows the conceptual architecture of our ontology-driven framework.

Translators enable the information flow between the different process-oriented applications (such as scheduling, simulation, and optimization) to a "neutral" process language, IDEF3. The advantage of using a neutral language to facilitate information flow between multiple languages is that it provides translation efficiency. Determining the information flow requirements for these inter-tool translators occurs through an analysis of the application (method and tool) ontologies. The IDEF3 "neutral" process language depicted at the center of Fig. 4 facilitates the *efficient* transfer of information between multiple (process-oriented) tools. The absence of a neutral (i.e., vendor-independent) process language would require the design and development of pair-wise translators between the tools, leading to additional translator development and maintenance effort. Once the neutral language has been designed, the translator design involves the development and analysis of the ontologies of the different methods and the different tools. A key step is to

determine mappings between the different method and tool ontologies (a formal specification of their concepts).

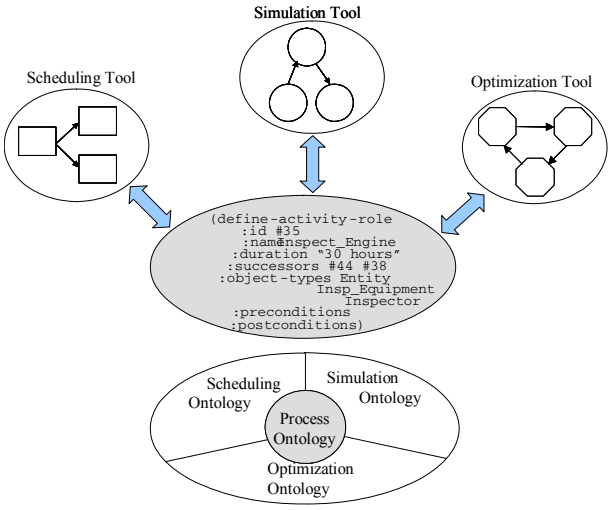

Fig. 4: Ontology-driven Framework for Process-oriented Applications

3.2 The Role of Ontologies in Component Based Simulation

Today's highly complex systems require that simulation models developed by different teams in different domains interact with one another to serve a higher goal. The component simulation models that are to be integrated are often called federates, and the application that results from successful integration is often called a federation. There are multiple challenges involved in component-based simulation. The component models may or may not have been developed with the federation in mind. The lack of "architectures that can be used to implement modeling tools and frameworks that can assemble simulations from model components while preserving syntactical and semantic correctness" has been well documented. The principal technical challenge that must be addressed in dealing with this comprehensive and critical void include modeling and simulation composability, semantic interoperability and information sharing, and model composition at multiple levels of abstraction.

An ontology-driven approach to component-based simulation includes the following steps.

- *Create component ontology*. Define an ontology for all components that explicitly captures the definition of goals of the component, detailed descriptions of net inputs required for execution, detailed descriptions of total outputs generated, system requirements, constraints on successful execution, and preconditions and post conditions, etc.
- *Create a repository*. Create a virtual repository of components and component ontologies.
- *Allow repository access to the M&S community*. An M&S expert attempting to compose a federation must be able to easily identify, based on ontology descriptions, the components required for the composition. The repository must allow for the downloading of component copies for such uses.

Our approach of using ontologies to facilitate component-based simulation is graphically depicted in Fig. 5. The M&S expert composing the simulation queries the *Component Repository* for ontologies based on a simulation requirement. This translates into the choice of appropriate components, which are downloaded from the repository and used to compose a simulation. In the absence of such repositories, it would be impossible to choose the appropriate components, even if such components were available.

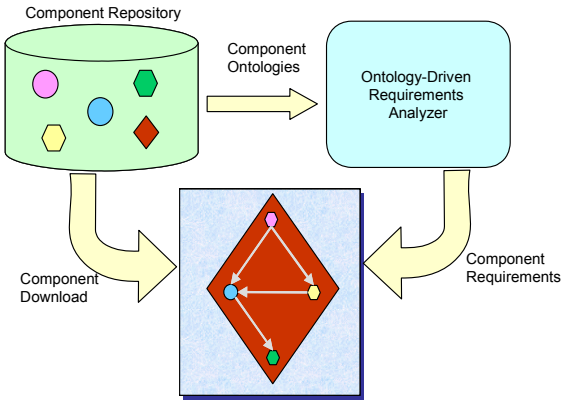

Fig. 5: Ontology-driven Component Based Simulation

3.3 The Role of Ontologies in Simulation Modeling at Multiple Levels of Abstraction

Simulation model components for component-based simulation are often developed at different levels of abstraction. Hence, multiple models may represent the same phenomenon but contain different levels of detail depending on the objective of the study for which they were originally developed. Low resolution models are typically used to see the bigger picture and obtain initial cut system analyses, whereas high resolution models help one to understand the underlying phenomenon, to represent knowledge, and to simulate reality.

In multi-resolution modeling, models that make-up the system might be developed at different resolution levels. Based on the goals of the exercise, the user might pick up

the models at the appropriate resolution levels and execute the system. This creates some problems in modeling and execution that are fairly unique to MRM. High resolution models tend to describe objects at a much higher granularity, whereas in low resolution models, the modeler may choose to aggregate some of these objects into a single object. For example, a theater-level model may define a tank as one object, whereas an engineering level model will most likely model each individual part of the tank. Note that even though the engineering level model provides greater accuracy with respect to the performance of the tank, for tactical decision-making, the higher-level model will be more useful.

Some problems with successful implementation of MRM identified by (Benjamin, et al. 2006b) are consistency of information across different levels, including the right level of detail in the models, lack of techniques that allow integration of objects not only at the data level but also at the behavioral level, or lack of techniques for information flow between different resolution levels. Some of these problems can be addressed by devising appropriate algorithms to detect any potential mismatches among federates before an integration is attempted, which is the subject of discussion in (Benjamin, et. al. 2006b). However, an *ontology-driven approach* can mitigate the effect of some of these problems. For example, if component model ontology defines the goal, the resolution level (using a standard-based, application independent language), granularity, intended user (e.g., army general vs. engineer in ballistics), fidelity, etc., models at appropriate levels can be chosen. This will result in fewer potential mismatches discovered, which in turn will result in the reduction of non-value added time in the process of simulation composition.

4 AN ONTOLOGY-BASED FRAMEWORK FOR SIMULATION MODELING AND ANALYSIS

In this section we outline the architecture of an Ontology-driven Simulation Modeling Framework (OSMF) solution that provides a "visual programming environment" to rapidly compose, build, and maintain distributed, federated simulations.

Central to the OSMF solution concept are model libraries and reference libraries. The "OSMF Model Libraries" contain template models that encapsulate structural and behavioral information that will allow for the rapid composition of simulation from re-usable component parts. The modeling procedure will involve selecting appropriate templates, editing the templates, and composing complex models by connecting together multiple components. The model libraries will include ontology templates, process (behavior) templates, information meta-models, design patterns, and example simulation models. The "Reference Libraries" refer to re-usable and extendible "domain models," including reference ontologies, reference process models, and reference information meta-models. The reference libraries will provide a mechanism for composing and integrating (at the semantic level) external simulation models and tools from other environments or vendors. For example, a reference "Threat Ontology" will help integrate a threat model with the combat simulation Blue Force models by determining the mappings in the threat ontology, reference ontology, and OSMF ontology.

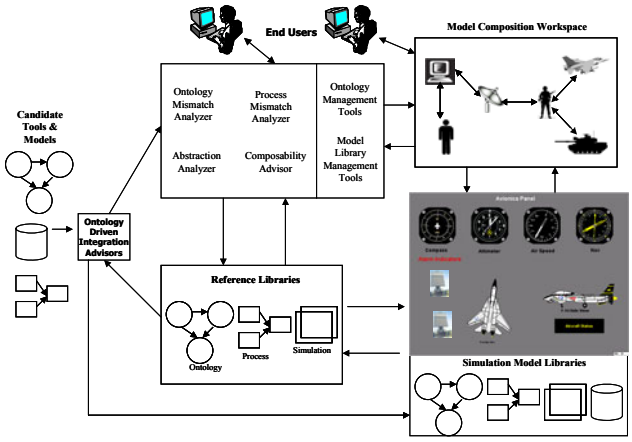

Fig. 6: Ontology-driven Simulation Modeling Framework

The model Composition Workspace is used to browse and edit the model libraries and to provide a graphical user interface for M&S application composition. A set of Composability Assessment Tools (CAT) are invoked to diagnose mismatches between candidate models that are to be integrated and generate model reconfiguration advice for revising the model library components or candidate new models that need to be integrated into the OSMF. Two types of composability assessment are supported using the simulation specification models ("pre-" analysis) and using simulation output ("post" analysis). The Ontology Driven Translator Generator is used to generate translator code between external models and the OSMF.

The OSMF may be run in two modes: Simulation Application Development Mode and Configuration / Set-Up Mode. In the *Simulation Application Development Mode*, the information necessary to develop simulation-based applications resides within the OSMF. In the *Configuration / Set-Up Mode*, models that are external to the OSMF [and that have been independently developed in other environments (e.g., a "legacy" model)] will be invoked or integrated for the purpose of the intended simulated experiment.

5 BENEFITS

The principal benefits of the research described in this paper include significant reductions in time and effort

needed for developing and deploying distributed simulation applications, significant reductions in the time and effort needed to facilitate knowledge sharing, communication, and semantic integration for modeling and simulation applications. The immediate application of the OSMF solution architecture will be to facilitate the development and execution of distributed simulation applications. Longer term, OSMF will provide a sustainable mechanism for building simulations that interoperate and share information at the semantic level. We are currently implementing the OSMF solution architecture on distributed simulation applications with the U.S. Air Force, the U.S. Army, and NASA.

REFERENCES

Benjamin, P., C. Menzel, and R. J. Mayer (1995). Towards a method for acquiring CIM ontologies. *International Journal of Computer Integrated Manufacturing*, 8 (3), 225–234.

Benjamin, P., K. Akella, K. Malek, and R. Fernandes (2005). An ontology-driven framework for process-oriented applications. *Proceedings of the 2005 Winter Simulation Conference.* Piscataway, New Jersey: Institute for Electrical and Electronics Engineers.

Benjamin, P. and M. Graul (2006a). A framework for adaptive modeling and ontology-driven simulation. *Proceedings of SPIE, Enabling Technologies for Simulation Science X, Vol.6227*

Benjamin, P., M. Graul, M. Patki, and M. Erraguntla (2006b). A framework for modeling and simulation at multiple levels of abstraction. *Proceedings of SPIE, Enabling Technologies for Simulation Science X, Vol.6227*

Fishwick, P. and J. Miller. (2004) Ontologies for modeling and simulation: Issues and approaches. *Proceedings of 2004 Winter Simulation Conference*. Piscataway, New Jersey: Institute for Electrical and Electronics Engineers.

Gruber, T. R. (1992). Ontolingua: a mechanism to support portable ontologies. *Knowledge Systems Laboratory Technical Report KSL 91-66, Final Version*, Stanford University.

Guarino, N. (1998). Formal ontology and information systems. In *Proceedings of Formal Ontologies in Information System.*

Hobbs, J., W. Croft, T. Davies, D. Edwards, and K. Laws, (1987). The TACITUS Commonsense Knowledge Base, Artificial Intelligence Research Center, SRI International.

Lacy, L. and W. Gerber. (2004) Potential modeling and simulation applications of the web ontology language – OWL. *Proceedings of the 2004 Winter Simulation Conference*. Piscataway, New Jersey: Institute for Electrical and Electronics Engineers.

Miller, J. and G. Baramidze. (2005). Simulation and the semantic web. *Proceedings of the 2005 Winter Simulation Conference*. Piscataway, New Jersey: Institute for Electrical and Electronics Engineers.

Neches, R., R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. R. Swartout (1991). Enabling technology for knowledge sharing. *AI Magazine* 12(3):36-56.

Petty, M. D. and E. W. Weisel (2003). A composability lexicon. *Proceedings of the Spring 2003 Simulation Interoperability Workshop*, Orlando FL, March 30-April 4 2003, pp. 181-187.

AUTHOR BIOGRAPHIES

PERAKATH BENJAMIN is Vice President (R&D) at Knowledge Based Systems, Inc. (KBSI), and manages and directs the R&D activities at KBSI. He has over 18 years of professional experience in systems analysis, design, development, testing, documentation, deployment, and training. Dr. Benjamin has been responsible for the development of advanced technology that is being applied extensively throughout industry and government.

MUKUL PATKI is a research scientist at KBSI and has expertise in the area of quality and applied statistics. Since joining KBSI, he has worked on various projects involving simulation, statistical analysis, data mining, process modeling, and ontology.

RICHARD MAYER is the co-founder of KBSI and has served as KBSI's President and Senior Research Scientist. Under his direction, KBSI has received funding for applications in engineering design and manufacturing, large-scale systems analysis, next generation CAD/CAM systems, simulation modeling, forecasting, data mining and analytic tool development, and concurrent engineering and information engineering methods and tools.