# A Semantic Web Services-based Architecture for Model Management Systems

Amit V. Deokar
*Dakota State University*
*Madison, SD 57042*
*amit.deokar@dsu.edu*

Omar F. El-Gayar
*Dakota State University*
*Madison, SD 57042*
*omar.el-gayar@dsu.edu*

## Abstract

*In the context of decision support systems, a model management system (MMS) is analogous to a database management system providing support for the various phases of a modeling life-cycle while insulating user from the physical aspects of model base storage and processing. An underlying premise is the recognition of models as a resource that need to be managed and modeling as an activity that need to be supported within organizations. The recent developments of the web and distributed computing environments is creating ever increasing demands for sharing and reusing heterogeneous models over corporate intranets and the Internet.*

*To this end, this paper presents a semantic web services-based architecture for MMS. The architecture is based on service-oriented principles and web services standards to support model sharing and reuse in a distributed environment. Integral to the architecture is the use of ontologies and artificial intelligence (AI) planning techniques to facilitate model composition. We demonstrate the features and advantages of the proposed architecture using representative examples. We conclude with a discussion of the current state of implementation and plans for future work.*

## 1. Introduction

Modern organizations are faced with numerous information management challenges in an increasingly complex and dynamic environment. Vast amounts of data and myriads of models of reality are routinely used to predict key outcomes. Decision support systems (DSS) play a key role in facilitating decision making through management of data and models. The basic thrust of such applications is to enable decision-makers to focus on making decisions rather than being heavily involved in gathering data, and conceiving and selecting analytical decision models [1].

Internet technology has brought many new opportunities to conduct business electronically, leading to increased globalization. Managers and decision makers are increasingly collaborating in distributed environments in order to make efficient and effective use of organizational resources [1]. Thus, the need for distributed decision support in general and model management in particular is more today than ever before. This has attracted significant attention from researchers in information systems related areas to develop a computing infrastructure to assist such distributed model management [2].

As noted above, models form a key component of typical DSS. Models can be conceived as specific formulations of decision situations amenable to certain problem solving techniques, such as simple linear regression, or an LP product mix formulation. Examples of models include, a demand forecasting model for predicting customer calls in a call center, a production planning model to decide optimal product quantities to be produced. Supporting the modeling life-cycle encompasses variety of functionalities including description, manipulation, integration, composition, execution of models [2]. The plethora of models and the recognition of models as a corporate resource are creating ever increasing demands on model sharing and reuse with particular emphasis on model composition, i.e., leveraging existing models by linking a sequence of models to develop a composite model [3].

A number of model composition approaches have been reported in the literature [2]. Nevertheless, model composition remains a challenging and a resource intensive endeavor in terms of generating an appropriate sequence of models by searching available model resources. Also, in many instances, partial solutions may be available or models may fail during execution. Adaptation and re-composition becomes an important issue in such cases. Especially, providing automated support for model composition and execution in distributed settings is an ongoing area of research.

In this paper, we focus on supporting model sharing and reuse by facilitating automated model composition for distributed model management. Service-oriented paradigm has evolved over the past few years as a prime technology candidate to support distributed applications, and interoperability across platforms and data sources. We propose such a service-oriented architecture for addressing model composition in distributed setting. Moreover, we leverage advances in semantic web and ontologies, as well as AI planning techniques for effective model composition and execution.

1530-1605/08 $25.00 © 2008 IEEE

The remainder of the paper is organized as follows: Section 2 briefly presents a set of motivational examples for distributed decision support while Section 3 provides a brief literature survey of research in the areas of model management, distributed model management, and model composition. Section 4 presents relevant concepts and research pertaining to ontologies and the semantic web. Section 5 discusses the potential synergy between models and services. Next, section 6 proposes a service-oriented architecture for distributed model management emphasizing the mechanics of model composition using relevant standards from the semantic web research as well as application of Hierarchical Task Network (HTN) technique for service planning. Further, section 7 provides a case study illustration of the key concepts followed by a discussion of the current state of implementation in Section 8. Finally, the paper concludes with related discussion and future work in Section 9.

## 2. Motivating examples

Table 1. provides a representative list of examples of distributed model management application domains. According to Goul and Corral [4], enterprise modeling refers to the activities, process representations and conceptualizations of an enterprise. The objective is to improve enterprise integration and support analysis of an enterprise. Such models are more likely to exist as a collection of models rather that one monolithic model [5]. As envisioned by Ba et al. [5], a critical element of an enterprise modeling framework is the ability to automate building and executing task-specific models (from existing model fragments) as needed in response to user generated requests. Recent work by Sen at al. [6] further extends this notion by proposing an architecture for dynamic and inter-organizational decision support. The framework proposed in our paper complements Sen et al. [6] proposed architecture at layer 2 "Unified Enterprise Modeling Language (UEML) representation of models" and layer 3 "Decision Support Environment (DSE) middleware" with a particular focus on decision models supporting enterprise decision making processes.

The proliferation of decision technologies in organizations has been hampered by a number of problems that are not normally encountered by software where a "mass market" exists [7]. From a user's perspective, examples of these problems include: awareness of the existence of relevant tools and models, access to relevant technologies, compatibility with existing infrastructure and tools, and interoperability with diverse data sources, models, and tools.

While the concept environmental management is not necessarily new, the emphasis on sustainable development has been gaining significant momentum since the Brundland Report [8]. Nevertheless, environmental management is a complex endeavor and provide rich application domain for decision support systems. Specifically, environmental DSS are often used to handle ill-structured problems where the structure of the problem and its associated solutions is developed progressively over time using a variety of data sources, analysis models, and visualization techniques. Implementation of such systems must be able to assemble various decision support components to meet the requirements of the problems at hand while catering for the complexity of such tools [9]. With respects to analysis models, a land zoning model may be augmented with a hydrological model to be able to handle water quality issues related to land zoning decisions, a tidal flow model may be connected with a surface flow model as in the HYDRA DSS, simulation models for smog analysis (DYMOS) may be linked to a traffic flow (DYNEMO) to assist with environmental planning, and a geographical information system module may be with a transport model for depicting transport phenomena. Moreover, in environmental management the situation is further compounded by variety, heterogeneity and multiplicity of analysis models and tools [10]. Such models are often developed independently with different data requirements (both from a semantic and syntactic perspectives).

**Table 1. Motivational examples of distributed model management application domains.**

| Application area |
|---|
| Enterprise modeling and dynamic decision support |
| Electronic markets for decision technologies |
| Environmental management and sustainable development |
| Scientific workflows |
| Crisis management |
| Healthcare and distributed decision support |
| Supply chain management |

The scientific community also encounter situations similar to those encountered in environmental management where there is a need to assemble a collection of models to address a particular analysis problem. Such models are often developed independently, are distributed, and have their particular semantic and syntactic requirements. [11]

Another promising application domain demonstrating the need for the composition of distributed models is in crisis management. For example, weather models may be linked to various crop yield models, which in turn may be linked to macro-economic models to analyze the effect of various crises on the food supply sector as well as the entire economy. Such models may be region specific and may be different to accommodate the possible heterogeneity in the available data across regions. Other notable application areas are healthcare and clinical decision support, and supply chain management.

## 3. Model Management

Model management encompasses variety of functionality including model description, model manipulation, scheduling, execution, and information display. Research in the area of model management has been ongoing since the 1980s. The initial thrust of this research was motivated by management science and operations research applications (e.g., Geoffrion's [12] structure modeling approach). While a comprehensive review of the model management (MM) literature can be found elsewhere [2, 13, 14].

Distributed model management has, since the mid-1990s, become critical with increased globalization demands. Advances in distributed computing have been leveraged by researchers to address this problem, some of which are mentioned here. DecisionNet, described by Bhargava et al. [7], is a prototype of a web-based architecture for sharing decision models. It is based on the idea that decision models can be shared by model providers and model consumers through a centralized registry mechanism, where models can be registered and located. A data warehouse based approach for model storage has been proposed by Dolk [15], utilizing [12] structured modeling approach for representing models. Huh and Kim [16, 17] proposed a framework for distributed collaborative model management, emphasizing coordination and propagation of changes in a model base on a real-time basis. Iyer et al. [18] recently proposed a web services architecture for model sharing and reuse of spread sheet models while Ezechukwu et al. [19] proposes an architecture for supporting distributed optimization over the Internet. Recently, Madhusudan [20] presented a framework for distributed model management based on web services. The framework utilizes the integrated Service Planning and Execution (ISP&E) [21] for composing web services.

In this research we extend the current literature along a number of dimensions. Specifically, the architecture is distributed in the true sense, in that, even the model management functionalities are exposed as web services. This is contrary to many distributed model management approaches discussed earlier, where although model resources are distributed, the model management functionalities reside in a centralized manner. This approach has a major advantage that a decision maker can query, compose, or deploy models using only a thin client, without bearing the burden of model management computations.

### 3.1. Model composition

Model composition is the problem of generating a sequence of models from a library of available models in response to a particular decision-making situation. Model composition is an important component of model management in the decision support context, where decision models are desired to be composed together from individual model units. It is often used interchangeably with the term model integration in the literature. However, we try to distinguish between the two terms based on the approach taken for synthesizing models together. Model integration focuses on synthesizing models at the structural or definitional level [15, 22-27]. At this level, different model schemas are integrated in a cohesive manner. Model composition, on the other hand, focuses on assembling models together at a functional level [3, 20, 28-33]. In this paper, we are concerned with the model composition aspect, especially providing automated support for it. It can also be noted that only few research proposals attempt to address model composition in distributed settings [5, 7, 34-36].

This research extends earlier work by emphasizing the use of ontologies and Artificial Intelligence (AI) planning techniques to compose distributed models in an automated fashion. Specifically, ontologies provide the necessary context and semantics for leveraging existing models, while AI planning techniques provide the means for selecting and assembling models in a manner consistent with the problem at hand.

## 4. Ontologies and semantic web services

Ontologies are explicit conceptualizations (i.e., meta-information) that describe the semantics of information resources [37]. Significant advances have recently been made along the lines of using ontologies for reasoning about resources available on the Web [38, 39]. Model resources such as models, solvers, or executable models that are distributed over the Web can lend themselves to these advances to provide better model management capabilities, particularly in distributed settings. In this section, we review some of the relevant advances and standards that are instrumental in realizing the semantic web infrastructure and are applicable for model management as well.

The Resource Description Framework (RDF) is a W3C standard that builds on top of XML to provide a data model for describing resources on the Web in terms of named properties and values, and encoded in a formal, machine-processable format [40]. An RDF description of a resource consists of a set of RDF statements (or triples). Each RDF triple consists of three parts: an object (a resource), an attribute (a property), and a value (another resource or plain literal).

RDF Schema (RDF-S) extends RDF by providing a type system for RDF or an ontological vocabulary for describing properties and classes of RDF resources [41]. RDF-S, thus provides a way to build an object model with a semantics for generalization-hierarchies of such properties and classes.

The Web Ontology Language (OWL) [42] goes a step further by adding more vocabulary for describing

properties and classes. Some examples include property type restrictions, equality, property characteristics, class intersection, and restricted cardinality.

Web services are a class of resources that are distributed, similar to models. In fact, in Section 4, we draw analogy between models and services. Essentially, web services are self-describing, self-contained software applications that are accessible over the Internet [43]. Web services form a corner stone of our proposed architecture for model management systems (refer Section 6) and its relation to semantic web technologies is discussed next.

Currently, web services are described procedurally using the Web Services Description Language (WSDL) [44], which lack semantic descriptions of web services. Several research approaches have been proposed to adding semantics to web service descriptions. Four submissions to the W3C consortium exemplify these approaches: OWL Web Ontology Language for Services (OWL-S) [45], Web Services Modeling Ontology (WSMO) [46], Semantic Web Services Framework (SWSF) [47], and Web Service Semantics (WSDL-S) [48].

We use OWL-S for providing semantics to models, encapsulated as web services in our architecture, and is briefly discussed here. OWL-S is an OWL-based Web Service Ontology language, whose objective is to provide a vocabulary for encoding rich semantic web service descriptions, in a way that builds upon OWL. Service descriptions may be provided using OWL-S that mainly consists of three interrelated sub-ontologies for the top-level concept *Service*, namely *service profile*, *service model*, and *service grounding*. The *service profile* is used to express 'what a service does', which may be used for service advertising, constructing service requests, and matchmaking. The *service model* provides essentially a process model to describe 'how the service works', in the form of inputs, outputs, preconditions, and effects (typically called IOPE), which may be used for service seeking, composing service descriptions, coordinating and monitoring of service executions. However, it can be noted that OWL-S takes the view that a process is not necessary a program to be executed, but a specification of the ways in which a client may interact with the service. There can be three types of processes: *atomic*, *composite*, and *simple*. *Atomic* processes correspond to the actions a service can perform by engaging it in a single interaction; *composite* processes correspond to actions that require multi-step protocols and/or multiple server actions; and *simple* processes provide an abstraction mechanism to provide multiple views of the same process. Finally, the *service grounding* provides information on 'how the service can be accessed' by mapping the constructs of the process model onto detailed specifications of message formats, protocols, and so forth (typically expressed in WSDL).

## 5. Models as services

Conceptually, a model as a loosely coupled component delivering a specific functionality can be conceived a service. Likewise, a service as an entity abstracting underlying logic can be considered as a model. In reference to the aforementioned principles underlying service orientation [43], and in the context of model management, the following is noted:

- Reuse: Much of the work underlying model selection, composition, and integration focuses on finding ways to leverage existing models through reuse.
- Abstraction: Models is an abstraction of reality. To facilitate model selection and composition, models commonly expose only the models' description and interface. Note that model integration with its underlying 'white box' assumption is inconsistent with service oriented principles.
- Autonomy: Similar to services, within its boundary (execution environment), models has complete autonomy independent of other models.
- Loose coupling: related to abstraction and autonomy, and in the context of model selection and composition, models are loosely coupled with other models.
- Statelessness: models are statelessness thereby supporting loose coupling and autonomy characteristics.
- Composability: supporting reusability, models may compose other models.
- Discoverability: models should facilitate their description and discovery for consumption by other models.

In effect, with the exception of model integration and model interpretation, a significant synergy exists between model management, and service-oriented technologies and management. The next section highlights opportunities for synergy between these two research areas.

## 6. An architecture for model management systems

A proposed service-oriented architecture for model management systems is illustrated in Figure 1.

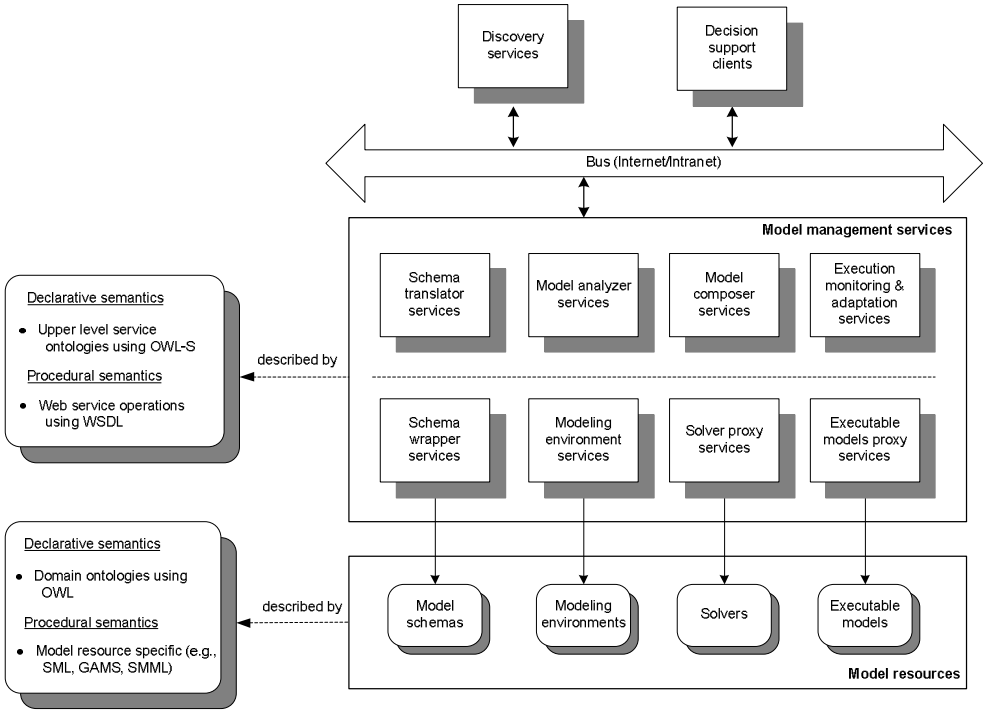


**Figure 1. A semantic web services-based architecture for model management systems**

A number of relevant design characteristics have been considered in this architecture. They are primarily driven by the issues and requirements for supporting model management during decision support in distributed settings: (1) a single model representation format [12], (2) representational independence of model structure and the detailed data [12, 49], (3) representational independence of model structure and the model solution [12, 49], (4) meta-modeling capability to support reasoning about models [49], (5) extensible for different modeling paradigms [12], and (6) accessibility of decision support resources [19].

This architecture builds on earlier work on distributed decision systems, with a particular emphasis on model management. The architecture is distributed in the true sense, in that, even the model management functionalities are exposed as web services. This is contrary to many distributed model management approaches discussed in Section 2.1, where although model resources are distributed, the model management functionalities reside in a centralized manner (see, e.g. [20]). This approach has a major advantage that a decision maker can query, compose, or deploy models using only a thin client, without bearing the burden of model management computations.

Two main aspects of the architecture are evident for model sharing and reuse. First, semantic description of models using OWL, provides a mechanism to reason about their properties. Second, the semantics associated with OWL-based ontologies of models is extensible for describing corresponding web services using OWL-S. These semantic web services can then be used either as atomic model units or composed together into composite model units to derive a solution for a particular decision making problem. Since both atomic as well as composite services are described using OWL-S, they can be discovered more effectively through logic-based search techniques, rather than just keyword based search. These two novel aspects of model management are discussed below.

## 6.1. Semantic descriptions of models

Associating semantic metadata to models and other model resources is essential in order to reason about their capabilities. Recent advances in ontologies and semantic web standards, discussed in Section 3, facilitate providing semantics to model resources through descriptions encoded in the form of their respective *domain ontologies*. Similarly, their corresponding web services as well as other supporting web services, such as model schema translator services, may be described in the form of *higher level ontologies*, particularly designed for web services.

Semantic descriptions of models can facilitate multiple uses. They can provide metadata for intelligent searching, browsing, and composing of models by decision makers. Moreover, implicit assumptions, model uses, constraints, and such can be explicitly captured through creation of domain ontologies of models. In fact, certain models may be elaborated in detail to provide, what can be termed as a

'white box' representation of models. Models (or model schemas) described using paradigms such as structured modeling, e.g. using SMML, may be semantically expressed to the finest level of detail with domain ontologies. Last, but not the least, these OWL-based domain ontologies can be extended to create higher level ontologies with OWL-S, in order to describe models wrapped as web services. The composer and execution monitoring services can make use of these higher level service ontologies to facilitate model composition functionalities.

Shown below is a snippet of domain ontology for a revenue computation model, described with OWL.

```
...
<owl:Class rdf:ID="FinancialModel">
   <rdfs:comment>Used to compute revenues and
                 Income
   </rdfs:comment>
   <rdfs:subClassOf rdf:resource="#Model"/>
</owl:Class>
<owl:DatatypeProperty rdf:ID="hasOutput">
   <rdfs:domain rdf:resource="#Model">
   <rdfs:range
            rdf:resource="&xsd;positiveInteger">
</owl:DatatypeProperty>
...
```

Similarly, shown below is a snippet of the OWL-S service ontology for the revenue computation model.

```
...
<Description rdf:about="#FinancialModel">
    <hasPreCondition>
    <expr:HTN-expression>
         <expr: expressionBody>
        ((computed-price ?product_price)
         (forecasted-demand
                              ?product_demand)
         (production-cost ?pcost)
         (distribution-cost ?dcost))
         </expr: expressionBody>
    </expr:HTN-expression>
    </hasPreCondition>
</Description rdf:about="#FinancialModel">
```

In essence, different models may use different model representation techniques or languages such as SML, DAM SQL, MDL, and LINDO. A model represented using any such technique is described using an OWL-based domain ontology. In this regard, OWL-based domain ontologies utilizing any such model representation technique may be generated computationally. Certain additional elements such as the application domain, model purpose, and so forth, are standardized across every model described with an OWL-based domain ontology.

## 6.2. Model composition approach

The overall proposed approach for model composition is as follows. Model composition is proposed as a web service which encodes the model composition algorithm. The model composition algorithm itself is based on Hierarchical Task Network (HTN) planning, which is a class of AI planning algorithms, and is discussed later. The semantic descriptions of models, provided in the form of higher level service ontologies using OWL-S, serve as a building block in how the model composer service may function.

The user request is formulated to capture the application context as well as to characterize the desired state. First, upon receiving a model composition request, the application context is used in conjunction with the OWL and OWL-S domain ontologies to semantically extract candidate models that may satisfy the user request upon composition. The parameterized state representation provided with the user request serves as an input to the model composition algorithm. Also, the composer service extracts the key elements of the model resources, now packaged as web services, including the input, output, precondition, and effects (IOPEs) from the ontologies. Finally, the model composition algorithm is then deployed to search the state space for potential composition of available model resources to respond favorably to the model composition request.

**6.2.1. AI planning for model composition.** Given ontological descriptions of models as well as auxiliary services, the problem of model composition can be formulated as a service planning problem [20]. AI planning techniques are particularly amenable for such problems, where the decision-making situation can be modeled as a desired goal state that needs to be attained from the current state. Simply stated, a solution to such a planning problem is possibly a sequence of operators (an operator is a parameterized function providing applicable state transformations), also called a *plan*, to achieve the desired transformation to the goal state. Extensive literature exists on a variety of AI planning techniques, comprehensively discussed in [50]. Hierarchical Task network (HTN) planning is observed to be most fitted to the task of model composition based on some of its distinct features, discussed next.

HTN planning is a hand-tailorable planner. In other words, it can encode domain-specific problem solving knowledge, along with domain-independent planning, which makes it significantly more efficient compared to other classical AI planning techniques. Also, HTN planning creates plans using ordered task decomposition. Thus, it plans for tasks in the same order that the tasks (in this case, services) will later be performed. The details of HTN planning and the Simple Hierarchical Ordered Planner (SHOP2) that is used for this research can be found in [51, 52].

In an HTN planner, the objective is not to achieve a set of goals but instead perform some set of *tasks* (symbolic representations of models to be executed in this case), also called an *initial task network*. The input to the planning system includes a set of *operators* (with preconditions and effects), similar to those of classical

planning and also a set of *methods*, each of which is a prescription for how to decompose some task into some set of *subtasks* (smaller tasks). Planning proceeds by using methods to decompose *nonprimitive tasks* recursively into smaller and smaller subtasks, until *primitive tasks* are reached that the plan executor can perform directly using the planning operators. Essentially, the HTN planning technique performs recursive search of the state space using ordered task decomposition and constraint satisfaction as its search-control strategy.

We have demonstrated the HTN planning technique for model composition for the example discussed by Kottemann and Dolk [33]. The problem of model composition is specified to the HTN planner as an initial task network, which forms the objective to be achieved. A sample snippet of a financial revenue model, represented declaratively as operators, is shown below. It can be noted that the IOPE is represented in the semantic description of OWL-S ontologies (snippet shown earlier).

```
(:operator (!compute-financial-model
   ?production_cost ?distribution_cost
   ?product_demand ?product_price)
;; preconditions
((computed-price ?product_price)
 (forecasted-demand ?product_demand)
 (production-cost ?production_cost)
 (distribution-cost ?distribution_cost))
;; delete list
()
;; add list
((computed-financial model ?revenue ?income)))
```

Multiple model compositions may be possibly generated using the HTN planning technique. In the particular example given in [33], two possible plans are generated, differing in their ordering sequence, based on the constraints, such as the preconditions. They are shown below (the parameters are not shown for simplicity).

```
Plan 1:
((((!forecast-demand)
  (!compute-production-cost)
  (!compute-distribution-cost)
  (!compute-price)
  (!check-convergence)
  (!compute-financial-model))
Plan 2:
((((!forecast-demand)
  (!compute-distribution-cost)
  (!compute-production-cost)
  (!compute-price)
  (!check-convergence)
  (!compute-financial-model))
```

During model composition, models may be selected from a plethora of alternative models, some of which may even provide similar functionality as other peer models in the library. Additionally, the planner may generate multiple composite models that satisfactorily meet the model composition request, but consist of different component models. Thus, model selection is important for both, selecting individual models for performing model composition, as well as selecting the appropriate composite model from a number composed alternatives.

We note that this issue is closely related to Quality of Service (QoS) for web services composition. Criteria such as availability, reliability, execution price, execution duration, reputation are discussed in the related literature [53-55]. Multi-criteria decision making techniques may be then employed, as mentioned in [56]. Other model management related criteria such as algorithmic solver performance of models, computing capacity, model representations may also need to be taken into account, as discussed in [57].

**6.2.2. Execution monitoring and adaptation.** Once a composite model is constructed by the model composer service, the client may choose to deploy this composite model to obtain the computational results. This is done by invoking an execution monitoring and adaptation service and fetching the composite model to this service. This service then retrieves the OWL-S ontologies for selected model resources, and uses the service model and service grounding descriptions therein, to invoke different model resource services in the appropriate sequence. The actual execution of these web services takes place on the host machines where these services reside.

Runtime exceptions such as a service failure may occur, which need to be managed by the execution monitoring and adaptation service. This service maintains a state monitor which allows it to compare the current and expected execution states and trigger any adaptation needed. Typical adaptation strategies include re-execution of services or re-planning. Optimization of these strategies is a related area of research.

## 7. Case study scenarios

To demonstrate interaction of the various services comprising the proposed architecture, we have developed a series of Unified Modeling Language (UML) sequence diagrams. Two representative scenarios are discussed below. The first scenario (Figure 2) demonstrates a typical interaction among model management services for composing a model from existing models and executing them in the appropriate order. In this scenario, the decision support client uses the discovery service to first try and locate the desired model. Due to unavailability of such a model, the decision support client then invokes a model composer service with the model request. The model composer service, in turn retrieves the semantic descriptions (OWL-S ontologies) of model resources (bundled as web services). Based on the service profile, and service model descriptions in these ontologies, the composer service extracts the IOPE for each model resource.

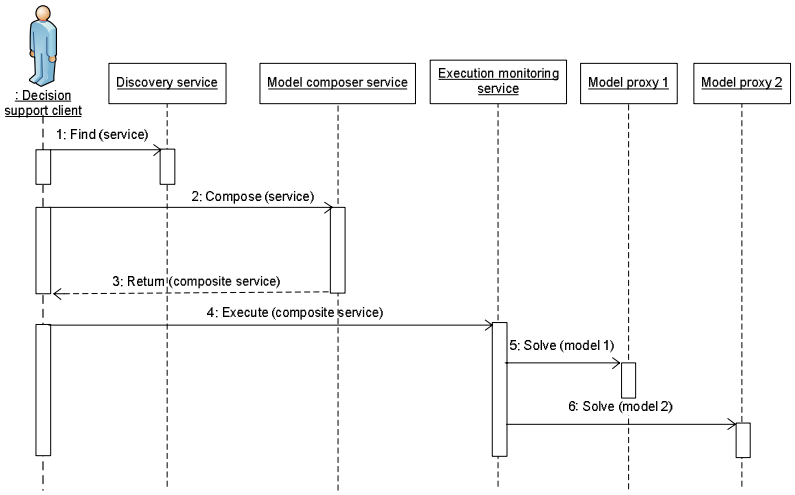


**Figure 2. Scenario#1 - Model composition and execution monitoring services**

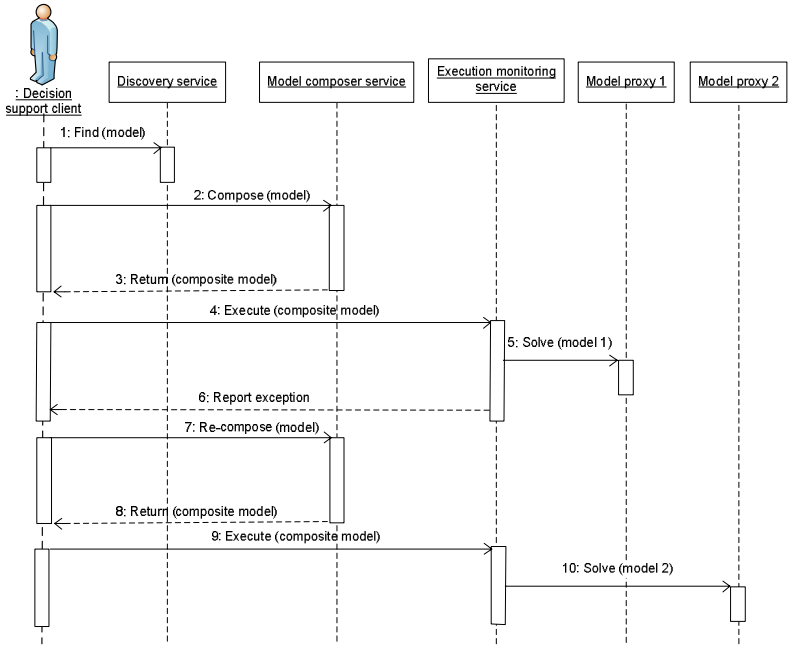


**Figure 3. Scenario#2 - Model composition and execution monitoring services with re-planning**

Next, it uses the embedded HTN planner to search for a feasible composite model. Since such a composite model is viable (in this scenario), the composer returns it to the decision support client. The client then uses the execution monitoring services to monitor the execution of the composite service generated. Again, semantic descriptions of selected model resources are retrieved by the execution monitoring service, since it uses the service model and service grounding descriptions in OWL-S ontologies to coordinate and monitor the ordered deployment of each model resource.

In the second scenario (Figure 3), a run time exception is shown to occur during the execution of the first model proxy service. The monitoring service then relays this message back to the client to seek further decision. Re-execution or re-planning of the composite model may be performed in this situation. In case of a re-execution decision, the monitoring service may be invoked again with the same composite service. However, in this case, the client chooses to re-plan to find an alternative way to compose the desired model, purposefully not considering the failed model resource. After finding the alternative plan, the monitoring service is triggered to coordinate the execution of this new composite model.

## 8. Implementation

A prototype of the proposed architecture is currently under development using J2EE platform and lisp-based HTN planner SHOP2. SHOP2 is open-source planning tool, developed at the University of Maryland [52]. The current emphasis is on developing model composer and execution monitoring services. Other model management services depicted in Figure 1 have been prototyped as part of our prior research effort [53, 54]. We are also populating a model library with associated ontologies in OWL and OWL-S using the Protégé editor. One of the important developments underway is the ability to provide computational translation of models described using different representation techniques into OWL and OWL-S model ontologies.

## 9. Discussion and concluding remarks

The proposed semantic web services-based architecture is based on the confluence of service-oriented principles and semantic web techniques. Design principles supported by SOA emphasize reuse, statelessness, autonomy, abstraction, discoverability, loose coupling, and composability. The proposed architecture for model management systems is novel in the following aspects. First, it is completed distributed through the provision for not only distributed model resources, but also distributed model management services. Second, it proposes a semantic layer for model representation that can facilitate automation and reasoning mechanisms, such as model composition. Third, model composition is illustrated using semantic web services and AI planning techniques.

There are several research directions that are relevant. Model representation may leverage semantic web techniques for providing rich model ontologies. There is also potential for extending the work on variety of model schema wrappers. Overall, the model management research can be extended in a number of ways by benefiting from the synergy between models and services.

## 10. References


[1] E. Turban, J. E. Aronson, T.-P. Liang, and R. Sharda, *Decision Support and Business Intelligence Systems*, Eighth ed.: Pearson Prentice Hall, 2007.

[2] R. Krishnan and K. Chari, "Model management: Survey, future reseach directions and a bibliography," *Interactive Transactions of OR/MS,* vol. 3, 2000.

[3] K. Chari, "Model composition in a distributed environment," *Decision Support Systems,* vol. 35, pp. 399-413, Jun 2003.

[4] M. Goul and C. K., "Enterprise model management and next generation decision support," *Decision Support Systems,* vol. 43, pp. 915-932, 2007.

[5] S. Ba, K. R. Lang, and A. B. Whinston, "Enterprise decision support using Intranet technology," *Decision Support Systems,* vol. 20, pp. 99-134, 1997.

[6] S. Sen, H. Demirkan, and M. Goul, "Dynamic decision support through instantiation of UEML representations," *Communications of the ACM,* vol. 50, 2007.

[7] H. K. Bhargava, R. Krishnan, and R. Muller, "Decision support on demand: Emerging electronic markets for decision technologies," *Decision Support Systems,* vol. 19, pp. 193-214, 1997.

[8] WCED, *Our common future*. Oxford: Oxford University Press, 1987.

[9] D. Abel, K. Taylor, G. Walker, and G. Williams, "Design of decision support systems as federated information systems," in *Decision Support Systems for Sustainable Development: A resource book of methods and application*, G. Kersten, Z. Mikolajuk, and G. Yeh, Eds. Boston/Dordrecht/London: Kluwer Academic Publishers, 2000, pp. 305-328.

[10] P. Hall, "Decision support systems for sustainable development: Experience and Potential," in *Decision Support Systems for Sustainable Development: A resource book of methods and application*, G. Kersten, Z. Mikolajuk, and G. Yeh, Eds. Boston/Dordrecht/London: Kluwer Academic Publishers, 2000, pp. 369-390.

[11] I. J. Taylor, E. Deelman, D. B. Gannon, and M. Shields, *Workflows for e-Science: Scientific Workflows for Grids*: Springer, 2006.

[12] A. M. Geoffrion, "An introduction to structural modeling," *Management Science,* vol. 33, pp. 547-588, May 1987.

[13] R. W. Blanning, "Model management systems : An overview," *Decision Support Systems,* vol. 9, pp. 9-18, 1993.

[14] A.-M. Chang, C. W. Holsapple, and A. B. Whinston, "Model management issues and directions," *Decision Support Systems,* vol. 9, pp. 19-37, 1993.

[15] D. R. Dolk, "Integrated model management in the data warehouse era," *European Journal of Operational Research,* vol. 122, pp. 199-218, 2000.

[16] S. Y. Huh, Q. B. Chung, and H. M. Kim, "Collaborative model management in departmental computing," *Infor,* vol. 38, pp. 373-389, Nov 2000.

[17] S. Y. Huh and H. M. Kim, "A real-time synchronization mechanism for collaborative model management," *Decision Support Systems,* vol. 37, pp. 315-330, Jun 2004.

[18] B. Iyer, G. Shankaranarayanan, and M. L. Lenard, "Model management decision environment: a Web service prototype for spreadsheet models," *Decision Support Systems,* vol. 40, pp. 283-304, 2005.

[19] O. C. Ezechukwu and I. Maros, "OOF: Open Optimization Framework," Department of Computing, Imperial College London, Departmental Technical Report 2003/7, April 2003.

[20] T. Madhusudan, "A web services framework for distributed model management," *Information Systems Frontiers,* vol. 9, pp. 9-27, Mar 2007.

[21] T. Madhusudan and N. Uttamsingh, "A declarative approach to composing web services in dynamic

environments," *Decision Support Systems,* vol. 41, pp. 325-357, January 2006.

[22] A. Basu and R. W. Blanning, "Model integration using metagraphs," *Information Systems Research,* vol. 5, pp. 195-218, Sep 1994.

[23] A. Basu and R. W. Blanning, "The analysis of assumptions in model bases using metagraphs," *Management Science,* vol. 44, pp. 982-995, Jul 1998.

[24] G. H. Bradley and R. D. Clemence Jr., "Model integration with a typed executable modeling language," in *Proceedings of the Twenty-First Hawaii International Conference on System Sciences (HICSS-21 '88)*, Kailua-Kona, HI, 1988, pp. 403-410.

[25] D. R. Dolk and J. E. Kottemann, "Model integration and a theory of models," *Decision Support Systems,* vol. 9, pp. 51-63, 1993.

[26] A. M. Geoffrion, "Reusing structured models via model integration," in *Proceedings of the Twenty-Second Hawaii International Conference on System Sciences (HICSS-22 '89)*, Kailua-Kona, HI, 1989, pp. 601-611.

[27] T. P. Liang and B. R. Konsynski, "Modeling by analogy: Use of analogical reasoning in model management systems," *Decision Support Systems,* vol. 9, pp. 113-125, Jan 1993.

[28] M. I. Bu-Hulaiga and H. K. Jain, "An interactive plan-based procedure for model integration in DSS," in *Proceedings of the Twenty-First Hawaii International Conference on System Sciences (HICSS '88)*, Kailua-Kona, HI, 1988, pp. 428-434.

[29] K. Chari, "Model composition using filter spaces," *Information Systems Research,* vol. 13, pp. 15-35, Mar 2002.

[30] V. Dhar and M. Jarke, "On modeling processes," *Decision Support Systems,* vol. 9, pp. 39-49, 1993.

[31] A. Dutta and A. Basu, "An artificial intelligence approach to model management in decision support systems," *IEEE Computer,* vol. 17, pp. 89-97, September 1984.

[32] J. E. Kottemann and D. R. Dolk, "Process-oriented model integration," in *Proceedings of the Twenty-First Hawaii International Conference on System Sciences (HICSS-21 '92)*, Kailua-Kona, HI, 1988, pp. 396-402.

[33] J. E. Kottemann and D. R. Dolk, "Model integration and modeling languages: A process perspective," *Information Systems Research,* vol. 3, pp. 1-16, 1992.

[34] M. Holocher, R. Michalski, D. Solte, and F. Vicuna, "MIDA: An open systems architecture for model-oriented integration of data and algorithms," *Decision Support Systems,* vol. 20, pp. 135-147, 1997.

[35] M. A. Jeusfeld and T. X. Bui, "Distributed decision support and organizational connectivity," *Decision Support Systems,* vol. 19, pp. 215-225, 1997.

[36] M. K. Mayer, "Future trends in model management systems: Parallel and distributed extensions," *Decision Support Systems,* vol. 22, pp. 325-335, 1998.

[37] D. Fensel, *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*: Springer, 2001.

[38] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific American,* pp. 35-43, May 2001.

[39] D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster, "Spinning the semantic web: Bringing the world wide web to its full potential," Cambridge, Massachusetts: The MIT Press, 2003.

[40] Resource Description Framework (RDF): Concepts and Abstract Synatax (W3C Recommendation), http://www.w3.org/TR/rdf-concepts/, 2004.

[41] RDF Vocabulary Description Language 1.0: RDF Schema (W3C Recommendation), http://www.w3.org/TR/rdf-schema/, 2004.

[42] OWL Web Ontology Language Semantics and Abstract Syntax (W3C Recommendation), http://www.w3.org/TR/owl-absyn/, 2004.

[43] D. F. Ferguson and M. L. Stockton, "Service-oriented architecture: Programming model and product architecture," *IBM Systems Journal,* vol. 44, pp. 753-780, 2005.

[44] Web Services Description Language (WSDL) 1.1 (W3C Recommendation), http://www.w3.org/TR/wsdl, 2001.

[45] OWL Web Ontology Language for Services (OWL-S) (W3C Member Submission), http://www.w3.org/Submission/2004/07/, 2004.

[46] Web Service Modeling Ontology (WSMO) Submission (W3C Member Submission), http://www.w3.org/Submission/2005/06/, 2005.

[47] Semantic Web Services Framework (SWSF) (W3C Member Submission), http://www.w3.org/Submission/2005/07/, 2005.

[48] WSDL-S Submission Request to W3C (W3C Member Submission), http://www.w3.org/Submission/2005/10/, 2005.

[49] W. A. Muhanna and R. A. Pick, "Meta-modeling concepts and tools for model management: A systems approach," *Management Science,* vol. 40, pp. 1093-1123, 1994.

[50] M. Ghallab, D. Nau, and P. Traverso, *Automated Planning: Theory and Practice*: Elsevier, 2004.

[51] D. S. Nau, T.-C. Au, O. Ilghami, U. Kuter, H. Munoz-Avila, J. W. Murdock, D. Wu, and F. Yaman, "Applications of SHOP and SHOP2," *IEEE Intelligent Systems,* vol. 20, pp. 34-41, March-April 2005.

[52] D. S. Nau, T.-C. Au, O. Ilghami, U. Kuter, J. W. Murdock, D. Wu, and F. Yaman, "SHOP2: An HTN planning system," *Journal of Artifical Intelligence Research,* vol. 20, pp. 379-404, December 2003.

[53] O. F. El-Gayar, "A service-oriented architecture for distributed decision support systems," in *37th Annual Meeting of the Decision Sciences Institute*, San Antonio, TX, USA, 2006.

[54] A. V. Deokar, M. Therani, R. O. Briggs, and J. F. Nunamaker Jr., "A structured approach to designing interleaved workflow and groupware tasks," in *Proceedings of the Tenth Americas Conference on Information Systems (AMCIS 04)*, New York, 2004.