4-2013

A semantic service-oriented architecture for distributed model management systems

Omar F. El-Gayar
Dakota State University

Amit Deokar
Dakota State University

Recommended Citation

El-Gayar, O., & Deokar, A. (2013). A semantic service-oriented architecture for distributed model management systems. Decision support systems, 55(1), 374-384.

Decision Support Systems 55 (2013) 374–384

Contents lists available at SciVerse ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

A semantic service-oriented architecture for distributed model management systems

Omar El-Gayar *, Amit Deokar

Dakota State University, 820 N. Washington Avenue, Madison, SD 57042, United States

ARTICLE INFO

Available online 29 May 2012

Keywords:
Decision support systems
Distributed model management
Service-oriented architecture
Semantic web services

ABSTRACT

Decision models are organizational resources that need to be managed to facilitate sharing and reuse. In today's networked economy, the ubiquity of the Internet and distributed computing environments further amplifies the need and the potential for distributed model management system (DMMS) that manages decision models throughout the modeling lifecycle and throughout the extended enterprise.
This paper presents a service-oriented architecture for DMMS. The proposed architecture leverages service-oriented design principles and recent developments in semantic web services to enable model sharing and reuse in a distributed setting. The paper describes a prototype implementation, case study scenarios, and a discussion highlighting lessons learned and implications for research and practice.

1. Introduction

Organizations are increasingly dependent on decision support systems (DSS) and associated models for data analysis and decision support rendering such applications ubiquitous. Supporting the proliferation of such applications is a plethora of modeling languages and platforms, database management technologies, and software engineering approaches for designing and implementing decision support systems. Moreover, with the advent of the Web and distributed computing environments, there is an increasing demand to leverage the existing investment in decision models and data within and across organizations. In this paper, we follow Krishnan and Chari's notion of a 'model' [40]: "*A model (or a model schema) is a formal abstract representation of reality and constitutes an important component of decision support systems. Models can be instantiated with data to create model instances that represent specific problem situations. Model instances are solved by computer executable programs known as solvers to obtain model solutions.*"

Analogous to database management systems, model management systems aim at recognizing models as a corporate (and national) resource that needs to be managed. The objective is to provide the necessary functionalities for manipulating decision models and supporting the modeling process [14,40]. The advent of the Internet created unique opportunities for extending model management functionality such as sharing and reuse in a distributed setting within and across organizations. However, in contrast to data, effort for sharing and reusing models has been hampered by the use of a variety of modeling languages and environments, model formats, difficulty in publishing and discovering models in a distributed setting, and difficulty of executing models in a variety of formats and in a distributed setting.

Service-oriented architecture (SOA) coupled with semantic web technologies can potentially address many of the aforementioned issues. Demirkan et al. [16], emphasize the value that service-oriented technologies and management can bring for businesses. In general, services are independent building blocks that collaborate to deliver application functionality. Services differ from 'traditional' components with respect to the underlying design principles, most notably are autonomy from other services, and compliance to a standard communications framework. The focus is on exposing application logic as loosely coupled services. Design principles underlying SOA emphasize reuse, abstraction, loose coupling, statelessness, composability, and discoverability [20,46]. Semantic web technologies address semantic inter-operability issues thereby facilitating information exchange functions such as model publication, discovery and use.

The objective of this paper is to address problems encountered in sharing and reusing models in a distributed setting. This is accomplished through the design and implementation of a semantic SOA for distributed model management. The rest of the paper follows Peffers et al. [47] and is organized as follows: Section 2 defines the problem and the importance of the solution. Building on the prior section, Section 3 defines the objectives and requirements for the proposed approach. Section 4 describes the design and development of a distributed model management system (DMMS), followed by a demonstration of the functionality of the proposed system and an evaluation of the proposed system in Sections 5 and 6, respectively. Section 7 concludes the paper.

2. Problem identification and motivation

The problem under consideration pertains to issues and challenges encountered when sharing and reusing models in a distributed

* Corresponding author.
E-mail addresses: omar.el-gayar@dsu.edu (O. El-Gayar), amit.deokar@dsu.edu (A. Deokar).

0167-9236/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.dss.2012.05.046

setting. The grand challenge is developing next generation model management systems that are particularly suited to today's distributed environment. The underlying issues and challenges stem from the heterogeneity of model representation formats and the heterogeneity of modeling environments resulting in accessibility and compatibility issues, the lack of awareness of the existence of relevant models, and the lack of universally accepted semantics.

Addressing these aforementioned issues will allow for the seamless exchange of models within and across organizations, for the reuse of models, i.e., using the same model schema with different data sets, different solvers, and different modeling environments, and for the creation of model repositories that can serve as a lasting archive of models. These archives will capture the state of knowledge and will also ensure the availability of these models regardless of the availability of the modeling environment. The utility of a distributed model management system that can address these challenges extends beyond business organizations and is equally important at a national level where significant expenditure is directed to the creation of cyberinfrastructures for supporting research [2]. In this context, DMMS provides the information management capabilities of sharing, reusing, and archiving scientific models for the advancement of scientific research. The following subsections describe a set of motivational scenarios highlighting the challenges encountered in a distributed setting, discuss the current state of distributed model management systems, and describe the contribution of the proposed solution relevant to the current state of the art.

2.1. Motivational scenarios

2.1.1. Scenario 1: intra-organizational model sharing

Consider an organization where there is a need to share decision models among units or branches within the organization. For example, the headquarters uses a decision model for developing the best marketing mix including advertising expenditure, product quality index, and product distribution. A regional branch operating in markets characterized by high variability in demand, different competition conditions, and distribution channels would need to reuse the former model taking into account new parameters and competitive dynamics for its market. Alternatively, various branches may have developed their own models or have adapted existing models to meet their specific requirements. It would be advantageous if each branch is able to share its models with other branches in a seamless manner.

Nevertheless, such goal is often hampered by lack of awareness of the existence of such models in the first place, heterogeneity of modeling environments resulting in accessibility and compatibility issues, and inadequate (or lack of) documentation that often employs inconsistent semantics complicating the problem of assessing the applicability of a particular model as well as the possibility of customizing such models to the situation at hand. The aforementioned issues are even more prevalent in an inter-organizational setting.

2.1.2. Scenario 2: models as knowledge objects

Knowledge intensive business services (KIBS) are private organizations that rely heavily on professional knowledge for supplying intermediate products or services that are knowledge-based [43]. Examples of KIBS include IT support services, management consultancy, and engineering consultancy. According to Hertog [34], KIBS capture scientific and technological information that is often dispersed across the economy, and tailor such information to meet the needs of its client. Interaction between KIBS and their clients involve extensive knowledge flows that take a variety of forms. While tacit knowledge is a significant component of knowledge flow, explicit forms of knowledge such as written reports, project plans, software, and decision models are also prevalent.

For example, consider a consultancy firm that provides services to help its client firms address issues pertaining to their projected energy demand, cost, and optimal mix of their energy portfolio. The firm relies on a number of decision models for forecasting energy demand and supply, prices for various forms of energy, transportation and distribution costs, etc. A client is interested in utilizing energy price forecasting models developed by the consultancy firm that meets its particular needs and is compatible with its own production and distribution models. Given the likelihood that the client may be using different modeling environments and assumptions, such utilization may be severely hampered. The situation is further complicated if the client wishes to select and test a variety of such models for their suitability to their particular needs. Such a situation may be encountered with other clients as well.

2.1.3. Scenario 3: domain specific model sharing — environmental management

While the concept environmental management is not necessarily new, the emphasis on sustainable development has been gaining significant momentum since the Brundland Report [53]. Nevertheless, environmental management is a complex endeavor and provides a rich application domain for decision support systems. Specifically, environmental DSS is often used to handle ill-structured problems, where the structure of the problem and its associated solutions are 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 [1]. With respect 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 coupled with a transport model for depicting transport phenomena. Moreover, in environmental management the situation is further compounded by the variety, heterogeneity and multiplicity of analysis models and tools [30]. Such models are often developed independently with different data requirements (both from semantic and syntactic perspectives).

2.2. Distributed model management

Model management (MM) research has attempted to address some of the aforementioned problems. The focus is on providing the necessary functionalities for manipulating quantitative decision models including model representation, model manipulation, model selection, model composition, solution computation, and result information display and analysis [14]. Some of the functionalities of MM resemble those of database management systems (DBMS) such as model description, manipulation, and control [17]. One important function of MM is model selection. Model selection [4] focuses on identifying a model type or schema for a specific problem instance under consideration. With the quest for supporting more sophisticated modeling tasks, research in MM has looked at complex research problems such as model composition and model integration. While model integration deals with orchestrating more complex models from two or more existing models at the structural or definitional level [5,27], model composition deals with sequencing models from the models library at the functional level [15]. A comprehensive review of these functionalities can be found in [11,14,40].

The advent of the Internet, the World Wide Web and the proliferation of computer networks have energized the decision support research community to explore means for sharing and reusing models and decision support tools in such environments. In that regard, Bhargava et al. [7] proposed a web-based architecture for sharing decision models, prototyped as DecisionNet application. The main idea is

that of sharing models by publishing and retrieving them through a centralized registry mechanism, similar to yellow pages, by model providers and consumers. The approach promoted the notion of conceptualizing decision technologies as objects that can be "used without having to be owned" and at a broader level, the notion of electronic markets for decision technologies [7]. Hue and Kim [35] and Hue et al. [36] proposed a framework for distributed collaborative model management, emphasizing coordination and propagation of changes in a model base in real-time. The focus is on coordinating the changes made to a collection of shared models and propagating the effect of these changes throughout the organization.

With regard to web services, Iyer et al. [37] recently proposed a web services architecture for model sharing and reuse of spreadsheet models, while Ezechukwu et al. [21] proposed an architecture for supporting distributed optimization over the Internet. The architecture is comprised of Algebraic Modeling Language (AML) for representing models, an Optimization Reporting Markup Language (ORML) for representing model solutions, and a collection of Java programs referred to as Optimization Service Connectivity Protocol (OSCP) that are responsible for converting AML models to a target system for execution and converting model results to ORML. Madhusudan [41] presented a framework for distributed model management based on web services. The framework utilizes the integrated Service Planning and Execution (ISP&E) [42] for composing web services. Bose and Sugumaran [12] present a similar framework yet focus on web-based decision support and discuss how semantic web technologies can be leveraged in such environments. Neither [41] nor [12] specifically discuss design or implementation issues pertaining to model sharing in such environments.

Related work in this area envisions the Web as a development and delivery environment for decision support systems [48]. The focus is on developing and deploying decision support systems in a distributed setting (specifically the Web) as well as the delivery of decision support tools such as modeling environments and solvers as a service in a distributed setting. In that regard, Zhang and Goddard [54] recognize the need for leverage heterogeneous and distributed data and decision tools often encountered in web-based DSS and propose a layered software architecture and a component-based framework for addressing these needs. The layered software architecture provides a formal view of a web-based DSS at design stage, while the component-based framework addresses implementation in a distributed environment. The approach focuses on providing decision support functionality for a particular application, i.e., application driven as opposed to model management functionality in a distributed setting. Focusing on optimization, Valente and Mitra [50] discuss application service providers (ASP) and e-Services as approaches for delivering optimization support in a web environment. The emphasis is on providing decision tools such as optimization modeling environments and solvers on demand. The concepts can be thought of as extensions of Bhargava et al.'s [9] notion of electronic markets with a specific focus on optimization. While Valente and Mitra [50] acknowledge the need to standardize model representation formats to facilitate model exchange, the emphasis is on the submission and consumption of these models by optimization services as opposed to the MM functionality such as sharing and reuse of models as an organizational resource.

2.3. Significance of the proposed approach

While past research identified and addressed a number of problems associated with the management of models as an organizational resource, limitations do exist particularly as it relates to the management of such models in a distributed setting. The proposed approach to a distributed model management architecture extends past research as follows:

1. It extends the work by Barghava [8], Iyer [37], and Ezechukwu and Maros [22] to explicitly leverage the concepts of service-oriented architecture to facilitate model sharing in a distributed environment, and to capitalize on the recent advances in distributed computing; in particular, web services as a means for utilizing decision support resources in a standardized and platform independent manner. It extends Madhusudan [41] and [12] by explicitly addressing how semantic web technologies and web services can be leveraged for supporting model management functions from a design as well as implementation perspectives.
2. It is truly distributed in nature, where models as well as different model management functionalities are conceived as services. This characteristic relieves the decision maker of performing computationally expensive operations by merely invoking them through a thin client interface.
3. It complements existing developments such as OSP and WEBOPT [50] that focus on delivering decision tools as services. In conceptualizing models as services, models can be consumed by decision tools in other environments with minor accommodations. Alternatively, mediator or proxy services can serve as a mechanism for consuming models from such environments.
4. It recognizes that to date there is no agreed upon standard format for representing quantitative decision models. Moreover, models can exist in various forms, e.g., as executables or at higher level of representation such as algebraic modeling language. This is accommodated through translator services as well as alternative provisions for black versus white box representations of models.
5. It enhances model and service descriptions with semantic web technologies. This characteristic allows reasoning about models in an intelligent manner to support the goals of model sharing and reuse. The architecture leverages Structured Modeling Markup Language (SMML) [19] and Semantically Annotated Web Service Description Language (SAWSDL) [39] to capture model semantics. However, the architecture is open to other formats, e.g. ODDM [10]. SMML is chosen for its reliance on Structured Modeling as an underlying formalism and SAWSDL for its acceptance as a W3C standard.

In summary, the proposed approach is an open architecture that leverages service-oriented and semantic web technologies to facilitate model management functionality in a distributed environment. The proposed architecture allows models (similar to data) to be leveraged as an organizational resource that can be shared and reused. The following section presents the requirements for such a solution.

3. Objectives for a DMMS

To identify the objectives of a DMMS we conducted an extensive review of the model management literature (1975–present), coupled with an analysis of representative motivational scenarios (see Section 2.1) and our own experience with a large DSS development project [18]. In this project, the emphasis was on sharing and reusing environmental data and supporting analysis models in the context of renewable energy assessment. The overarching need for the DSS is to facilitate model sharing and reuse in a distributed setting. This translates to a number of requirements as follows:

1. Ideally, a single model representation format as noted in [26] is recommended. However, in practice, this may not be possible [50]. The academic and practitioner decision support and modeling communities continue to propose alternative model representation languages and formats. Accordingly, a DMMS must be able to accommodate a variety of model representation formats in a seamless and scalable manner.
2. Representational independence of model structure and the detailed data [26,45]. This allows a particular model to be reused with different data sets.

3. Representational independence of model structure and the model solution [26,45]. This allows a particular model to be solved using various solvers and solution platforms. This allows leveraging existing initiatives aimed at providing modeling platforms and solvers as a service, (see [50]).
4. Meta-modeling capability to support reasoning about models [45]. In a distributed setting, and to facilitate model discovery and manipulation, such capabilities should extend beyond the traditional "keyword descriptions" and descriptions focusing on the syntactic aspects of the model to descriptions that capture semantics of the underlying parameters and variable. This requirement emphasizes the need to reason about syntactic as well as semantic knowledge embedded in models.
5. Extensible for different modeling paradigms [26]. Recent efforts (e.g., [22,25,37,50]) have focused on a particular modeling paradigm. Ideally, in a distributed setting, a DMMS should be capable to accommodate a variety of such paradigms in a standard and scalable manner.
6. Accessibility of decision support resources [21]. As noted in Zhang and Goddard [54], decision support systems in a distributed environment will inevitably need access to data and tools from multidisciplinary areas. Such resources will need to be integrated in a seamless and scalable manner.
7. Compatibility with web technologies [10] to ensure that existing investments in model management initiatives can be leveraged. Capitalizing on standardized web technologies will contribute to the long-term sustainability, acceptance, and scalability.

3.1. Use cases

From a functional perspective, the focus of a DMMS transcends the support of the modeling lifecycle to emphasize sharing and reuse. The first step in sharing is the creation of a model repository. This is enabled through a model publication function where a model developer can share his/her model with other users. To facilitate model discovery, model publication will include functionality to semantically annotate the model regardless of modeling formats. Once semantic and syntactic meta-model information is captured, the model schema can be saved (with or without a representative model instance, i.e., data).

To leverage models in an existing repository, a DMMS must be capable of providing the necessary functionality for discovering, i.e., searching and retrieving models relevant to a particular problem. The discovery functionality should be able to utilize meta-model information (semantic and syntactic) to identify and retrieve most relevant models. Given the inherent complexity, model discovery will need to support a semi-automated interactive search process where the user interacts with the DMMS search engine in an iterative fashion.

Once the user identifies a particular model, model selection functionality allows the user to select the desired model, further explores the data requirements for the model, and assists the user in choosing whether to solve the model in a distributed setting (using model execution services) or locally using services within the user's environment.

As indicated earlier, a DMMS will provide model execution functionality where a user can contribute model instances (problem specific data) to an execution engine for solving the model. Inherent in this functionality is the ability to leverage syntactic and semantic meta-model information to guide the user to the selection of the most appropriate solvers (model solving algorithms).

In addition to supporting the publication, discovery, selection, and execution of a single model, a DMMS may facilitate the composition of multiple models where two or more models are combined in sequence to solve a particular problem, e.g., a demand forecasting model feeding a transportation model. Composition algorithms will be able to leverage semantic and syntactic available in meta-model information to identify the best combinations of models for a specific problem.

The aforementioned use cases represent some of the typical functionality expected from a DMMS emphasizing sharing and reusing models. However, these use cases are by no means exhaustive. The following section describes the design and development of a DMMS that meets these requirements.

4. Design and development of a SOA for distributed model management

The architecture of the DMMS has been presented in Fig. 1. At the core of the architecture is a service bus providing the underlying communication infrastructure for the various services. The bus supports intra and inter-organization communication among services by implementing web services standards such as Simple Object Access Protocol (SOAP) over Hyper Text Transfer Protocol (HTTP). Connected to the bus is a collection of decision support services such as model management services, modeling support services, models as services, and problem domain ontologies for semantically annotating models. A decision support client can access any of the services connected to the bus irrespective of the physical location of the service.

4.1. Model semantics

Consistent with the notion of the semantic web [31], a particular characteristic of the proposed architecture is the support for the semantic annotation of models for the purpose of reasoning and conducting various model management operations on them. Semantic web developments seek to extend syntactic interoperability by providing semantic interoperability. Hendler [32] describes ontologies as knowledge terms that include semantic interconnections and simple rules of inference. Thus, by describing web resources semantically, ontologies allow web agents to share and comprehend these resources with minimum human intervention [6]. In essence, ontologies provide a conceptualization mechanism or a vocabulary to represent knowledge in a particular domain [13].

In this architecture, we utilize the Web Ontology Language (OWL) [52] to provide the semantic data model for describing model resources. Compared to the Resource Description Framework (RDF) and RDF Schema (RDF-S), OWL provides additional vocabulary for capturing model semantics.

Distributed resources such as decision models may be made available for consumption, sharing, and reuse, through web service technologies. In fact, with standardization efforts in this direction (e.g., the Web Services Description Language (WSDL)), service-oriented architectures and web service technologies are becoming more prevalent in organizations, particularly for conducting business in electronic markets or "emerging" services as termed by [49]. Examples of such services include online auctioning, e-commerce and targeted marketing, and self-service travel sites. Decision models have a key role to play in such electronic markets, and modeling them as services facilitates their active role.

In this architecture, models are conceptualized as web services and are described in a procedural manner using WSDL, which captures their functional characteristics. However, it lacks semantics in the sense that machines cannot reason about these descriptions to perform higher level operations such as searching, composing in an intelligent manner. To capture model semantics, we leverage SAWSDL (a W3C standard) [39] as a lightweight approach for extending WSDL service descriptions with semantic information. This is accomplished by introducing extension attributes to any WSDL or XML schema element. A modelReference attribute points to the relevant semantic concepts in the semantic data model, e.g., the domain ontology, while the schema mapping provides data transformations between XML data model and

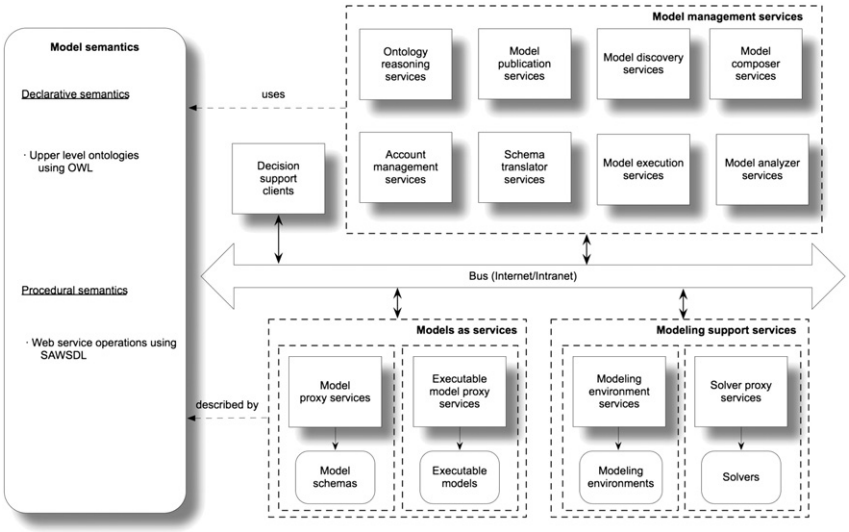


Fig. 1. DMMS architecture.

the semantic data model. With semantically enhanced web services, syntactic as well as semantic interoperability between services can be achieved.

4.2. Models as services

Conceptually, a model as a loosely coupled component delivering a specific functionality can be conceived a service. While there may be no generally accepted definition of "service", in this architecture we adopt Ferguson and Stockton's [24] definition in which a service is defined as having "*a well-defined interface (with a set of messages that the service receives and sends and a set of named operations), an implementation of the service, and if deployed, a binding to a documented network address*". In reference to the aforementioned principles underlying service-orientation, and in the context of distributed model management, the following is noted:

1. Abstraction: models are 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.
2. Autonomy: similar to services, within its boundary (execution environment), models have complete autonomy independent of other models.
3. Discoverability: models should facilitate their discovery for consumption by other models.
4. Reuse: much of the work underlying model selection, composition, and integration focuses on finding ways to leverage existing models through reuse.
5. Loose coupling: related to abstraction and autonomy, and in the context of model selection and composition, models are loosely coupled with other models.
6. Statelessness: models are stateless, thereby supporting loose coupling and autonomy characteristics.
7. Composability: supporting reusability, models may be composed using 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. In instantiating the notion of models as services, one will need to be cognizant to the diversity of model representation formats. As shown in Fig. 2(a), models exist in various forms and shapes. On one side, models can be in the form of binary executable with no access to the underlying model structure, i.e., a "black box". In this case, models are wrapped as web services and associated semantic information is captured in their SAWSDL description, as mentioned in Fig. 2(b).

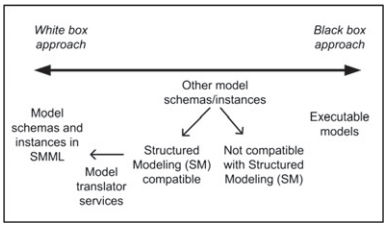
On the other side, model schemas and instances are represented using XML, e.g., SMML [19]. Such representation provides a theoretically founded model representation scheme, i.e., Structured Modeling (SM) thereby realizing desirable model management representation requirements such as model-data independence and model-solver independence. SMML is also amenable to sharing in a distributed setting as an XML application, and most notably provides explicit access to the model schema and instance structure, i.e., a "white box". As an XML application, model semantics is captured using attributes similar in structure to SAWSDL extension to WSDL noted earlier. These attributes relate concepts from the model schema to corresponding constructs from the problem domain ontology. In effect, semantically enriched SMML representation serves as meta-models capturing syntactic and semantic information about a model. Such models are made available for sharing and reuse in the form of model proxy services. Proxy services serve as "dummy" web services representing the model for the purpose of registering, discovering and executing the model. Supporting model discovery, operations within these services leverage semantic and syntactic information in the corresponding SMML representation. For model execution, operations provide access to model parameters as well as information necessary for using a compatible solver.

Models may also be represented using higher-level languages such as GAMS or LINGO as shown in Fig. 2(a). The mechanism for representing such models and for incorporating semantics will depend on whether these models are amenable to representation as structured models. For SM-compatible models, the user has the option to leverage schema translator services to translate the model from its original modeling language to SMML or to maintain the original language. For the former, semantic information will be captured within SMML as noted earlier. To accommodate the latter option as well as models that are not SM-compatible, model proxy services are created. Similar to SMML model proxy services, operations within these services provide the necessary functionality for discovering and executing these models. However, in this case, model semantics is captured in the SAWSDL description of these services, as depicted in Fig. 2(b).

4.3. Modeling support services

Modeling support services provide access and management to a variety of modeling resources, most notably, specialized solvers, and

(a) Range of modeling approaches



(b) Model representations, semantic annotations and model delivery

	Model Delivery
	Models as 'Services'
Model Representation	Model web services or proxy web services
Binary executable	SAWSDL
Higher level model representation SMML	Semantically Annotated SMML
Other model representation SMML compatible	Semantically Annotated SMML[1] or SAWSDL[2]
Non-SMML compatible	SAWSDL[2]

[1] Will need to translate to SMML.
[2] SAWSDL is used to annotate the proxy web service representing the model. In addition to operations capturing the parameters of the model, there are operations for accessing the model.

Fig. 2. Compatibility of DMMS with different modeling approaches.

modeling platforms and languages such as GAMS and AMPL. In this SOA, these resources are represented as web services. For solvers, solver proxy services are used to serve as an application program interface (API) to existing solvers. Operations within these services expose all the needed functionality to access the solver. This includes invoking the solver, passing the relevant parameters (model schema and instance), and retrieving results. For development environments and platforms such as AMPL or MATLAB, proxy services, termed modeling environment services, are used to expose the functionality and to manage the interface with these environments. Both kinds of services are registered with the Universal Description and Discovery Information (UDDI) server to facilitate discovery, followed by model execution. In a typical execution scenario, a model represented in a language such as GAMS will be orchestrated by model execution services involving the model proxy service, the modeling environment service (like GAMS proxy service), and a compatible solver proxy service. In this example, the GAMS proxy service will translate the GAMS model to the appropriate binary format needed for the particular solver.

4.4. Model management services

Model management services include services for supporting model management functions such as model publication, discovery, execution, translation, and composition and for administrative functions such as account management and ontology management. In that regard, model publication services allow users to incorporate semantic and syntactic meta-model information into their models and register their models to the model repository. Adopting XML web services, publication and discovery services utilizes a UDDI server for managing information about all registered services, i.e., serving as a registry. UDDI uses XML to represent its contents, and contains enough information to direct clients to additional resources such as WSDL files which in turn provide information about the functionality of a service and the details necessary to communicate with the service. It should be noted, however, that such repository can physically reside with the user or in a shared location. The physical location is transparent for a model consumer.

Model discovery services leverage semantic and syntactic meta-model information to identify models relevant to a particular problem. Model discovery services support an iterative search process allowing the user to refine their search criteria based on the returned results. Depending on the particular model representation, model discovery services 'consume' the semantic information incorporated in the SAWSDL description (for binary and non SM-compatible model proxy services) or in the semantically annotated SMML model representation.

Model execution services are responsible for solving a model. This involves providing the model with a compatible model instance (data), and identifying and invoking the appropriate solver service. The mechanism will differ depending on the underlying model representation. For example, in the case of binary executable models, a model execution service simply runs the model and returns the results of the execution of the model. In the case where proxy model services are involved, the role of the model execution service extends to orchestrating the translation of a model (in case of SMML models) into an executable model format and invocation of a compatible model solver service.

Account management services can be used to provide software licenses and access to fee-based services. Other model management services include services for model translation, model composition, and model analysis. Schema translator services represent a repository of services for translating model schemas to/from a variety of popular formats and languages such as GAMS and LINGO to/from SMML. Model composer services allow for leveraging a collection of models for a specific decision situation by coordinating the execution of such models. Model analyzer services provide functionality for analysis of model results and for conducting what-if analysis.

5. Demonstration

In this section, we summarize our efforts in the prototype implementation of DMMS, based on the architecture and design described earlier. Experiences with different tools in developing the prototype are highlighted followed by representative usage scenarios.

5.1. Building the model and the ontology repository

At the onset of the development efforts of the prototype, a repository consisting of decision models and semantic models was developed. This repository served as a test bed for experimenting with different design and implementation issues during the development of the different components of the proposed architecture shown in Fig. 1. Different forms of decision models covering the spectrum, shown in Fig. 2, have been included in the repository. Models described using SMML [19], shown on the left end of the spectrum form the majority of the repository. SMML models follow a "white box" approach where the model schema and instance structures are openly accessible through XML parsers. These models cover different levels of complexity, including mathematical programming, spreadsheet models, and predicate calculus models [28,29]. MathML [44] is used for encoding mathematical operations represented as functions within the SMML structured models. Other than SMML, models described in other higher level modeling representation formats include LINGO, and MPS, among others. The repository also contains problem domain ontologies in OWL describing relationships between concepts used in the decision models. In terms of tools and technologies used,

the use of an XML editor (Altova XMLSpy) and an equation editor (MathType for MathML equations) helped in lowering the manual model creation overhead for SMML models, particularly for complex models. This project uses Protégé 3.4, an ontology development tool for creating and editing OWL ontologies [38]. Specific advantages of Protégé include an ability to incorporate ontology reasoning engines, an application program interface (API) that can be deployed in semantic applications, a large and diverse user community, and availability at no charge.

5.2. Semantically annotating SMML models

The next phase involved semantically annotating SMML models in the repository. To incorporate semantics within SMML models, different design choices were explored including, (a) transforming SMML models into OWL ontologies and then providing references to other domain ontologies from this ontology, and (b) incorporating semantic references within existing SMML models. The former design choice entailed completely altering the model representation format, with lack of clarity in terms of what the ontology concepts and instances represent, and was not pursued. The later design choice, which was adopted, leverages the SMML representation and achieves the desired goal of incorporating semantics through a lightweight mechanism of providing semantic links from within SMML models. It does so by introducing additional attributes within SMML. This approach is analogous to the SAWSDL extension for WSDL [39]. Three key attributes have been introduced in the `GenusType` and `ModuleType` type definitions in SMML model structure schema, namely `semanticReference`, `liftingMapping`, and `loweringMapping`. The `semanticReference` attribute points to semantic concepts, while `loweringMapping` and `liftingMapping` attributes specify data transformations between a decision model's XML structure and the associated semantic model. A screenshot showing these additional attributes in the model structure schema is shown in Fig. 3(a). Semantically annotated SMML models in the repository reflect these annotations, thus linking to problem domain semantic models (OWL ontologies).

5.3. Developing web services for models and model management functions

Following the semantic annotation of decision models, the next phase involved encapsulating models as web services, i.e., model proxy services. Different design approaches were used corresponding to the various model types, as discussed in Section 4.2. In general, this involved a top-down approach, first generating corresponding WSDL descriptions, and then generating the web services corresponding to these descriptions. Java 2 Enterprise Edition (J2EE) platform and compatible technologies were used, given the availability of numerous open source technologies developed using Java that could be leveraged in the development process. The Eclipse Web Tools Platform (WTP) within the Eclipse Integrated Development Platform (IDE) was used as the development environment. Apache Axis2/Java implementation of the Apache Axis2 web service engine, along with the Tomcat web server was used for prototyping, and testing.

WSDL descriptions for models proxy services were annotated using SAWSDL annotation scheme. For SMML models, the problem domain concepts were annotated in the model schemas, while the modeling approach concepts (e.g., genus, module, etc. concepts pertaining to Structured Modeling) were annotated in the WSDL descriptions. For models other than SMML models, problem domain concepts were annotated in their WSDL descriptions, resulting in semantically rich model descriptions.

Developing modeling support services involves creating proxy web services to provide programmatic interfaces to solvers. In the prototype, a solver proxy service for LINDO solver was developed that can consume a model schema and instance (data) and return the results. Similarly, core model management services, particularly for model publication, discovery, translation, and execution were developed. Fig. 3(b) shows a model execution web service description snippet. Development of other related services for model composition, and for administrative functions such as account management and ontology management is planned as next steps.

5.4. Illustrative scenarios

To demonstrate interaction of the various services comprising the proposed architecture, we discuss two representative scenarios using Unified Modeling Language (UML) sequence diagrams illustrating discovery and execution use cases discussed in Section 3.1. The first scenario (Fig. 4(a)) demonstrates a typical interaction among services for invoking an executable model, i.e., a model that exists as a standalone executable, and visualizing the result using an existing user interface service. In this scenario, the decision support client uses the discovery service to locate the desired model and the data needed for the model. The discovery service in turn leverages the reasoning ontology service to incorporate model semantics in the search process. Once a model is located, the DSS client utilizes model execution service to execute the desired model (using its web services' endpoint information). The model execution service then invokes the particular model and model results are returned in a result file. The executable model considered here is representative of models on the right end of the spectrum of Fig. 2(a).

In the second scenario (Fig. 4(b)), a client wishes to execute a model represented in the SA-SMML format, a semantically enhanced XML-based representation for mathematical models (shown on left end of the spectrum in Fig. 2). Similar to scenario #1, the client uses the discovery service to locate the desired model. Since the model exists in a non-executable format, the model execution service utilizes a model proxy service during the orchestration of the model execution process. The execution process leverages a number of services as depicted in the diagram to translate the model into a model representation format, e.g., GAMS or LINGO that can then be compiled into a solver-compatible format. This step may be omitted if there are services available for directly manipulating semantically annotated SMML files. During the execution process, modeling environment services are responsible for compiling the model into a solver compatible format, e.g. MPS and invoking the appropriate solver.

Fig. 5 shows the user interfaces for the model publication, discovery, and execution services in the prototype system. Referring to the motivational scenarios discussed in Section 2.1, the ability to publish, discover, select, and execute models is the basis of sharing and reusing modeling resources, whether it is in the context of knowledge intensive business services, various domains such as a environmental management, or across functional units within an organization.

6. Evaluation and discussion

Following the demonstration of the utility of the DMMS architecture, we now evaluate the proposed architecture in terms of how it has addressed each of the requirements discussed in Section 3.

1. It is able to accommodate models represented using variety of modeling languages and formats. Models that can be represented using Structured Modeling, constituting mathematical programming models, and several other commonly used model types, can be translated from extant modeling formats (e.g., LINGO) to SMML and vice versa, using schema translator services. Such models are shared through model proxy services, while executable models with no model structural information are shared using executable model proxy services.

(a) Semantically annotated SMML snippet illustrating semantic annotation of models

```
<GENUS name="PLANT"
semanticReference="http://www.dsu.edu/mmsoa/ontology/supplychain.owl#plant">
        <TYPE>pe</TYPE>
        <INDEX>i</INDEX>
        <INTERPRETATION>There is a list of <KEY_PHRASE>PLANTS</KEY_PHRASE>.
        </INTERPRETATION>
</GENUS>

<GENUS name="SUP"
semanticReference="http://www.dsu.edu/mmsoa/ontology/supplychain.owl#supplier">
        <TYPE>a</TYPE>
```

(b) Web service illustration of model execution service

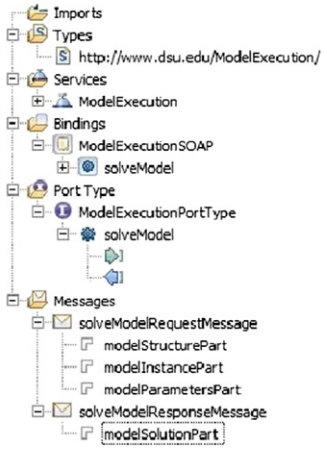
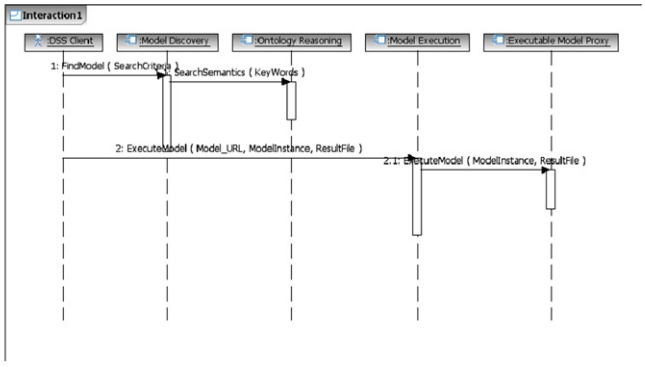


Fig. 3. DMMS demonstration code snippets.

(a) Scenario #1: Executing binary executable models (black box)



(b) Scenario #2: Executing models represented in higher formats, e.g., SMML (white box)

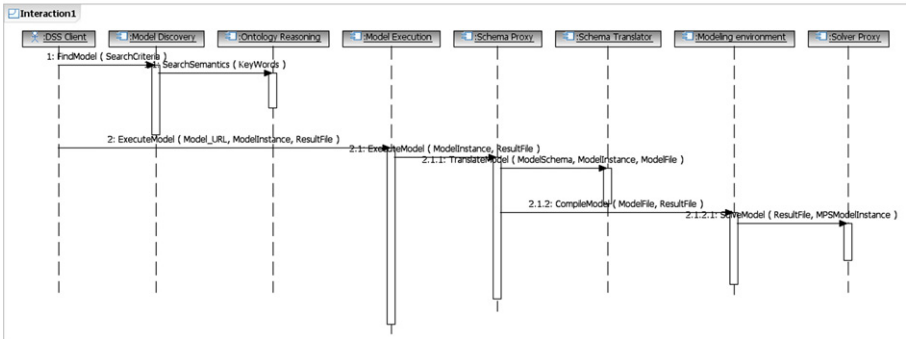


Fig. 4. Illustrative scenarios.

2. Representational independence in terms of separating the concerns of model structure/schema and model instance/data is supported by the proposed DMMS architecture. SMML [19], which is used as an intermediary model representation format, is based on Structured Modeling [26] that represents model schemas and instances independently. This inherently allows models to be used with different compatible data sets (model instances) through the proposed architecture.
3. Representational independence in terms of separating the concerns of model structure/schema and model solution is also supported by the proposed DMMS architecture. Analogous to the previous point, of using SMML [19] as the intermediate modeling format, which in turn is based on Structured Modeling [26], the model solution is inherently represented separately from the model schema. As such, a given model can be executed using different solvers that can address models of that type.
4. The proposed architecture provides support for meta-modeling, which is one of the key requirements. The use of SMML [19] as an XML model representation language allows for syntactic meta-modeling by including tags for capturing the structure of model schemas and instances and facilitating model management at a basic level through tags such as `KEY_PHRASE` associated with model schema and instance elements. Further, the lightweight semantic extension of SMML allows references to be made to domain semantic models (e.g., ontological concepts). Ontology reasoning services within the DMMS architecture use these meta-modeling facets, i.e. semantic references, to draw inferences, while supporting use cases such as model discovery and composition.
5. The proposed architecture is extensible to different modeling paradigms. The Structured Modeling [26] basis for model representation affords accommodating wide range of management science/operations research models. Furthers, models based on modeling paradigms that are not compatible with the Structured Modeling approach can also be shared within the architecture as model proxy services, although restrictive in terms of semantic search capabilities.
6. The proposed architecture is a service-oriented architecture and it provides accessibility to decision support resources by encapsulating and publishing them as services. In particular, services have been developed for resources such as model schemas, modeling environments, and solvers, in addition to model management services.
7. The proposed architecture is agnostic to the specific user interface employed, and is compatible with thick as well as thin clients. While a thin browser-based client has been developed as part of the prototype to demonstrate the essential functionality, a desktop-based client that provides richer user interface functionalities is also feasible.

During the design and the implementation of the proposed DMMS architecture, we have identified key issues that need to be taken into consideration in similar future projects. They may also be developed further into criteria that organizations can use to evaluate their business cases and decide whether such projects may be feasible and serve their organizational needs. The implications from this project fall into two main categories, namely technical and organizational.

6.1. Technical issues and implications

The technical issues are primarily related to semantic web technologies and model representation. The first technical issue stems from the phenomenon of evolving standards. Unlike some of the established areas, areas of semantic web and semantic web services are still maturing and have been the focus of research and development for industry practitioners and academicians in recent years. Consortia such as the W3C provide coordination and oversight of these standardization efforts in trying to overcome shortcomings of prior versions and capture the current understanding of various knowledge representation schemes through revised standards. As an example of this phenomenon, consider the semantic web standard OWL, which is currently the de facto standard for ontology representation. It has its roots in the Ontology Inference Layer (OIL) [23]. Later, DAML + OIL, also based on description logics [3], evolved from the earlier DARPA Agent Markup Language (DAML) [51]. OWL, is thus the result of the evolution of the DAML + OIL language, and has now become a W3C recommendation since 2004 [52]. It has since then undergone one major revision in 2009 to evolve into OWL2.

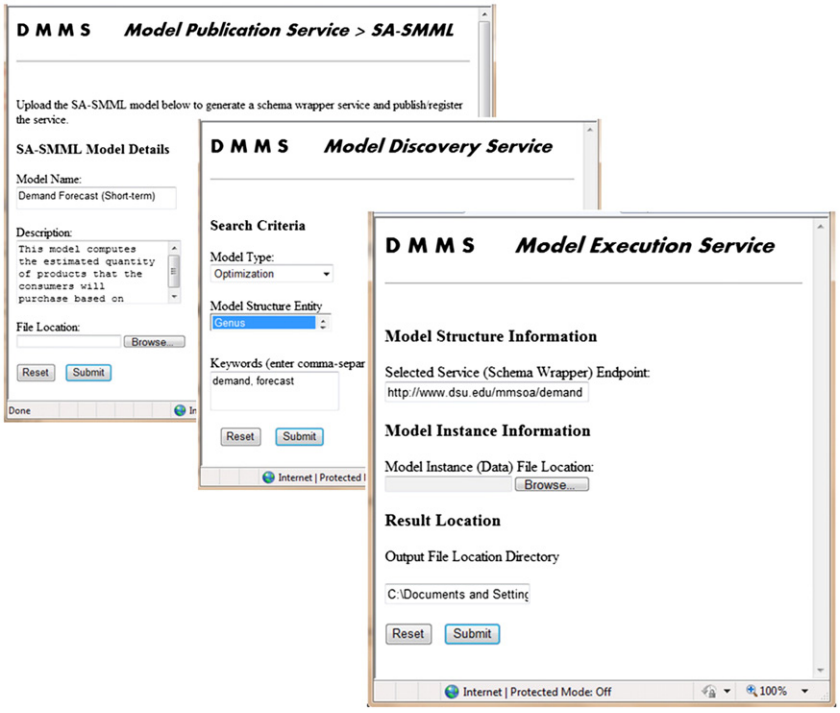


Fig. 5. Prototype screen shots.

Another example of the phenomenon of evolution of standards is the area of Semantic Web Services set of standards, which saw numerous proposals including OWL Web Ontology Language for Services (OWL-S), Web Services Modeling Ontology (WSMO), Semantic Web Services Framework (SWSF), and Web Service Semantics (WSDL-S). In 2007, W3C put forth a modified version of WSDL-S, SAWSDL as their recommendation [39]. The key implication from the perspective of developing decision support systems is that while adopting current standards is desirable, it is often times challenging, especially when the standardization efforts are in flux. Another related issue is that the tools and technologies that employ the current standards or maintain compatibility with them are evolving as well, however at a somewhat different pace depending on vendors. While efforts by vendors and various research groups in developing semantic web technologies and tools are noteworthy, nevertheless application developers relying on these tools and engaged in development of systems such as DMMS, have to be constantly wary about this evolution of standards, resulting tool incompatibility or obsolescence.

Technical issues in building systems such as DMMS also pertain to the developments in the model management area. The first issue is that the lack of standardization for representing models by different vendors creates a significant overhead in developing schema translator services to accommodate diverse modeling formats. These schema translator services need to generate models in formats amenable to be consumed by various solvers, suited for a specific decision problem types. SMML offers an intermediate format that can serve as a hub in the hub-and-spoke analogy where each of the spokes represents a specific modeling format. This reduces the overhead by avoiding creating translators between each pair of model representation schemes and instead using SMML as an intermediary. In this project, we developed a small number of translator services to illustrate the feasibility of the concepts. Nevertheless, exhaustive support of translator services is required for the architecture to gain acceptance and momentum, from an adoption perspective. Moreover, an increased reliance on SMML as an intermediary model representation format will require further work to specify and evaluate the lightweight mechanism for semantically annotating SMML models. A related issue within the realm of model management is that the Structured Modeling approach, which is used as a baseline for SMML can be restrictive in terms of the allowable model types. Additional support for other model representation schemes, such as those for continuous simulation modeling, is essential. From an adoption perspective, the limited number of tools for semantically annotating models highlights the significance of developing tools specifically designed for semantic annotation of models, e.g., via SAWSDL or SMML. A related implication, and given the complexity of the underlying problem, is the importance of initially limiting the scope to a certain class of models, e.g., mathematical programming models. This will also allow for the possibility of capturing semantics associated with a particular class of models.

6.2. Organizational issues and implications

Organizational issues encountered during the current implementation efforts of DMMS can be discussed in terms of issues related to the design and development of DMMS, i.e. near-term issues, as well as issues related to the wider scale implementation efforts, i.e. long-term issues. In terms of development efforts, particularly the processes involved, it was imperative to use an iterative and agile methodology for development, rather than a waterfall model approach. While requirements from a system perspective were quite well defined and stable, the tools and technologies supporting the relevant semantic web and semantic web services standards were evolving and thus were in a state of flux. This resulted in often requiring revisiting design decisions and exploring alternative design choices. The approach for semantically annotating SMML-based decision models was one such design choice.

Another issue related to the near-term development efforts was related to skill set requirements. The diversity of the technology imposed significant requirements on the skills and characteristics of the development team. As a result, in projects such as DMMS, programming skills in a particular language are insufficient. Additional knowledge and skills in (a) developing XML applications, (b) developing, semantically annotating, and deploying web services, (c) developing ontologies and programmatically working with them, and (d) understanding and developing management science models in various languages such as GAMS or LINGO are needed. Equally valuable is the ability to learn and assimilate new technologies and tools.

Wider implementation efforts will need to address issues realized during the current implementation efforts. Adoption and sustainability of such an infrastructure emerged as a key issue to be considered. Depending on the context, e.g. science and engineering community, intra-organization, or inter-organization, the significance of these issues will vary. In general, in an intra-organizational setting where problem domain ontologies already exist and models are recognized as significant components of the intellectual assets of the organization, it is reasonable to assume that the adoption and sustainability of such infrastructure is relatively easier to achieve. Regardless of the settings, a critical consideration is the motivation and willingness among participants to share their models. This in turn, may involve addressing variety of issues such as cost/benefit, vendor support, ownership, and confidentiality that may impede wider adoption of such an infrastructure. The incentives and penalties, if any, associated with sharing models, implicit or explicit will be a key determinant from a cost/benefit standpoint. Support from vendors in terms of accessibility of solvers, modeling language compilers, as services as well as license structure to support DMMS architecture will be needed for broader adoption. Given that models encapsulate knowledge nuggets that are organizational assets, the issue of ownership of models and data (model instances) will also need to be addressed in the wider implementation efforts, as model owners may not be willing to share their models for fear of loss of ownership. In a similar vein, security issues such as confidentiality of models and data will also need to be considered and solutions such as role-based security access will need to be used. Related implications of the aforementioned issues include the need to further explore specific factors affecting the adoption and diffusion of such systems and to develop business models that will ensure the sustainability of these systems.

7. Conclusions

In this paper, we presented the design and implementation of a distributed model management system. 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 [26], (2) representational independence of model structure and the detailed data [26,45], (3) representational independence of model structure and the model solution [26,45], (4) meta-modeling capability to support reasoning about models [45], (5) extensible for different modeling paradigms [26], (6) accessibility of decision support resources [21], and compatibility with the web [10]. 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, where although model resources are distributed, the model management functionalities reside in a centralized manner (see, e.g. [41]). 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.

Moreover, a key characteristic of the proposed system is leveraging semantic web technologies to facilitate model discovery, sharing, and reuse. Ongoing development effort also revealed key technical and organizational issues and implications for research and practice that will need to be addressed. While there are many arguments about the feasibility of the semantic web, both from theoretical and practical perspectives [33], the proposed system and supporting technologies are an initial step in leveraging these technologies in the context of mathematical models as an organizational and a national resource. The latter coincides with ongoing efforts at the national level to promote the creation of cyberinfrastructures [2].

References

[1] D. Abel, K. Taylor, G. Walker, G. Williams, Design of decision support systems as federated information systems, in: G. Kersten, Z. Mikolajuk, G. Yeh (Eds.), Decision Support Systems for Sustainable Development: a Resource Book of Methods and Application, Kluwer Academic Publishers, Boston/Dordrecht/London, 2000, pp. 305–328.

[2] D.E. Atkins, K.K. Droegemeier, S.I. Feldman, H. Garcia-Molina, M.L. Klein, D.G. Messerschmitt, P. Messina, J.P. Ostriker, M.H. Wright, Revolutionizing science and engineering through cyberinfrastructure, Report of Blue-Ribbon Advisory Panel on Cyberinfrastructure, National Science Foundation, January 2003.

[3] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, P. Patel-Schneider, The Description Logic Handbook: Theory, Implementation and Applications, Cambridge University Press, Cambridge, MA, 2003.

[4] S. Banerjee, A. Basu, Model type selection in an integrated DSS environment, Decision Support Systems 9 (1) (1993) 75–89.

[5] A. Basu, R.W. Blanning, Model integration using metagraphs, Information Systems Research 5 (3) (1994) 195–218.

[6] D. Bell, S. De Cesare, N. Iacovelli, M. Lycett, A. Merico, A framework for deriving semantic web services, Information Systems Frontiers 9 (1) (2007) 16.

[7] H.K. Bhargava, R. Krishnan, R. Muller, Decision support on demand: emerging electronic markets for decision technologies, Decision Support Systems 19 (1997) 193–214.

[8] H.K. Bhargava, R. Krishnan, S. Roehrig, M. Casey, D. Kaplan, R. Muller, Model management in electronic markets for decision technologies: a software agent approach, Presented at Hawaii International Conference on Systems Sciences, HICSS, Honolulu, Hawaii, 1997.

[9] H.K. Bhargava, R. Krishnan, S. Roehrig, M. Casey, D. Kaplan, R. Muller, Model management in electronic markets for decision technologies: a software agent approach, Presented at Proceedings of the 30th Hawaii International Conference on System Sciences (HICSS-30 '97), Honolulu, Hawaii, 1997.

[10] T. Bhrammanee, V. Wuwongse, ODDM: a framework for modelbases, Decision Support Systems 44 (3) (2008) 689–709.

[11] R.W. Blanning, Model management systems: an overview, Decision Support Systems 9 (1993) 9–18.

[12] R. Bose, V. Sugumaran, Semantic web technologies for enhancing intelligent DSS environments, Decision Support for Global Enterprises, 2007, pp. 221–238.

[13] B. Chandrasekaran, J.R. Josephson, V.R. Benjamins, What are ontologies and why do we need them? IEEE Intelligent Systems 14 (1) (1999) 20–26.

[14] A.-M. Chang, C.W. Holsapple, A.B. Whinston, Model management issues and directions, Decision Support Systems 9 (1993) 19–37.

[15] K. Chari, Model composition using filter spaces, Information Systems Research 13 (1) (2002) 15–35.

[16] H. Demirkan, R. Kauffman, J. Vayghan, H. Fill, D. Karagiannis, P. Maglio, Service-oriented technology and management: perspectives on research and practice for the coming decade, Electronic Commerce Research and Applications 7 (2008) 356–376.

[17] D.R. Dolk, Data as models: an approach to implementing model management, Decision Support Systems 2 (1) (1986) 73–80.

[18] O.F. El-Gayar, L. Michels, G. Fosnight, A. Deokar, The development of an EDSS: lessons learned and implications for DSS research. Proceedings of the 44th Annual Hawaii International Conference on System Sciences HICSS44 11. Kauai, HI: IEEE Computer Society; 2011.

[19] O.F. El-Gayar, K. Tandekar, An XML-based schema definition for model sharing and reuse in a distributed environment, Decision Support Systems 43 (2007) 791–808.

[20] T. Erl, Service-oriented Architecture: a Field Guide to Integrating XML and Web Services, Prentice Hall, 2004.

[21] O.C. Ezechukwu, I. Maros, OOF: open optimization framework, Departmental Technical Report 2003/7, Department of Computing, Imperial College, London, April 2003.

[22] O.C. Ezechukwu, I. Maros, OOF: Open Optimization Framework, March 9 2006 Retrieved from, http://www.doc.ic.ac.uk/old-doc/deptechrep/DTR03-7.pdf.

[23] D. Fensel, I. Horrocks, F. van Harmelen, S. Decker, M. Erdmann, M. Klein, Oil in a nutshell, in: R. Dieng (Ed.), Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling, and Management (EKAW '00), 2000, pp. 1–16.

[24] D.F. Ferguson, M.L. Stockton, Service-oriented architecture: programming model and product architecture, IBM Systems Journal 44 (4) (2005) 753.

[25] R. Fourer, J.P. Goux, Optimization as an Internet resource, Interfaces 31 (2) (2001).

[26] A.M. Geoffrion, An introduction to structural modeling, Management Science 33 (5) (1987) 547–588.

[27] A.M. Geoffrion, "Reusing Structured Models Via Model Integration," Presented at Proceedings of the Twenty-second Hawaii International Conference on System Sciences (HICSS-22 '89), Kailua-Kona, HI, 1989.

[28] A.M. Geoffrion, The SML language for structured modeling: level-1 and level-2, Operations Research 40 (1) (1992) 38–57.

[29] A.M. Geoffrion, The SML language for structured modeling: level-3 and level-4, Operations Research 40 (1) (1992) 58–75.

[30] P. Hall, Decision support systems for sustainable development: experience and potential, in: G. Kersten, Z. Mikolajuk, G. Yeh (Eds.), Decision Support Systems for Sustainable Development: a Resource Book of Methods and Application, Kluwer Academic Publishers, Boston/Dordrecht/London, 2000, pp. 369–390.

[31] J. Hebeler, M. FIsher, R. Blace, A. Perez-Lopez, M. Dean, Semantic Web Programming, Wiley, Indianapolis, IN, 2009.

[32] J. Hendler, Agents and the semantic web, IEEE Intelligent Systems 16 (2) (2001) 30–37.

[33] J. Hendler, The dark side of the semantic web (editorial), IEEE Intelligent Systems (2007) 2–4.

[34] P.D. Hertog, Co-producers of innovation: on the role of knowledge-intensive business services in innovation, in: J. Gadrey, F. Gallouj (Eds.), Productivity, Innovation and Knowledge in Services: New Economic and Socio-economic Approaches, Edward Elgar, Cheltenham, UK, 2002, pp. 223–253.

[35] S.Y. Huh, H.M. Kim, A real-time synchronization mechanism for collaborative model management, Decision Support Systems 37 (3) (2004) 315–330.

[36] S.Y. Huh, Q.B. Chung, H.M. Kim, Collaborative model management in departmental computing, Infor 38 (4) (2000) 373–389.

[37] B. Iyer, G. Shankaranarayanan, M.L. Lenard, Model management decision environment: a web service prototype for spreadsheet models, Decision Support Systems 40 (2) (2005) 283–304.

[38] H. Knublauch, R. Fergerson, N. Noy, M. Musen, The protégé OWL plugin: an open development environment for semantic web applications, Presented at Third International Semantic Web Conference — ISWC 2004, 2004.

[39] J. Kopecký, T. Vitvar, C. Bournez, J. Farrell, SAWSDL: semantic annotations for WSDL and XML schema, IEEE Internet Computing 11 (6) (2007) 60–67.

[40] R. Krishnan, K. Chari, Model management: survey, future research directions and a bibliography, Interactive Transactions of OR/MS, 3, 2000.

[41] T. Madhusudan, A web services framework for distributed model management, Information Systems Frontiers 9 (1) (2007) 9–27.

[42] T. Madhusudan, N. Uttamsingh, A declarative approach to composing web services in dynamic environments, Decision Support Systems 41 (2) (2006) 325–357.

[43] I. Miles, N. Kastrinos, K. Flanagan, R. Bilderbeek, P. den Hertog, W. Huitink, M. Bouman, Knowledge intensive business services: their role as users, carriers, and sources of innovation, Eims Publication No. 15, Innovation Programme, Dgxiii, Luxembourg, 1995.

[44] R. Miner, The importance of MathML to mathematics communication, Notices of the American Mathematical Society 52 (5) (2005) 532–538.

[45] W.A. Muhanna, R.A. Pick, Meta-modeling concepts and tools for model management: a systems approach, Management Science 40 (9) (1994) 1093–1123.

[46] E. Newcomer, G. Lomow, Understanding SOA with Web Services, Addison-Wesley, Upper Saddle River, NJ, 2005.

[47] K. Peffers, T. Tuunanen, M.A. Rothenberger, S. Chatterjee, A design science research methodology for information systems research, Journal of Management Information Systems 24 (2007) 45–77.

[48] D.J. Power, R. Sharda, Model-driven decision support systems: concepts and research directions, Decision Support Systems 43 (3) (2007) 1044–1061.

[49] J.M. Tien, Toward a decision informatics paradigm: a real-time, information-based approach to decision making, IEEE transactions on systems, man, and cybernetics: part C, Applications and Reviews 33 (1) (2003).

[50] P. Valente, G. Mitra, The evolution of web-based optimisation: from ASP to e-Services, Decision Support Systems 43 (4) (2007) 1096–1116.

[51] F. van Harmelen, P. Patel-Schneider, I. Horrocks, Reference description of the DAML+Oil ontology markup language, http://www.daml.org(2001/03/reference).

[52] W3C, W3c recommendations: (1) resource description framework (RDF): concepts and abstract syntax, (2) RDF vocabulary description language 1.0: RDF schema, 2004, http://www.w3.org/standards/techs/rdf.

[53] WCED, Our Common Future, Oxford University Press, Oxford, 1987.

[54] S. Zhang, S. Goddard, A software architecture and framework for web-based distributed decision support systems, Decision Support Systems 43 (2007) 1133–1150.

Omar El-Gayar is a Professor of Information Systems and Dean of Graduate Studies and Research, Dakota State University. His research interests include: decision support systems, multiple criteria decision making, and the application of decision technologies in healthcare, environmental management, and security planning and management. His inter-disciplinary educational background and training is in information technology, computer science, economics, and operations research. Dr. El-Gayar's industry experience includes working as an analyst, modeler, and programmer. His numerous publications appear in various information technology-related fields. He is a member of AIS, ACM, INFORMS, and DSI.

Amit V. Deokar is an Assistant Professor of Information Systems in the College of Business and Information Systems at Dakota State University. His recent research interests are in decision support systems, business process management, collaboration processes and technologies, knowledge management, and healthcare informatics. He has published several conference publications, journal articles, and book chapters in these areas. He holds a BE in Mechanical Engineering from V.J. Technological Institute, Mumbai, a MS in Industrial Engineering from the University of Arizona, and a PhD in Management Information Systems from the University of Arizona. He is a member of AIS, MWAIS, ACM, and AAAI.