2010

Developing a semantic web-based distributed model management system: Experiences and lessons learned

Amit Deokar
Dakota State University

Omar F. El-Gayar
Dakota State University

Ruba Aljafari
Dakota State University

Recommended Citation

Deokar, A. V., El-Gayar, O. F., & Aljafari, R. (2010, January). Developing a semantic web-based distributed model management system: Experiences and lessons learned. In 2010 43rd Hawaii International Conference on System Sciences (pp. 1-10). IEEE.

Developing a Semantic Web-based Distributed Model Management System: Experiences and Lessons Learned

Amit V. Deokar[1]
Dakota State University
Madison, South Dakota
amit.deokar@dsu.edu

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

Ruba Aljafari
Dakota State University
Madison, South Dakota
rhaljafari@pluto.dsu.edu

[1] All authors contributed equally to this work.

Abstract

Distributed model management systems (DMMSs) are decision support systems with a focus on managing decision models throughout the modeling lifecycle and across the extended enterprise. The advent and proliferation of web services and semantic web technologies offers the possibilities of sharing and reusing models in a distributed setting.

This paper presents the design and implementation of a semantic web-based DMMS. Key lessons learned, technical and organizational issues encountered are summarized and directions for future research have been outlined. From a technical perspective, future research will need to explore the viability of tools specifically designed to facilitate the semantic annotation of models, specify and validate SA-SMML, and extend the white-box approach presented in this paper to other model types not amenable to structured modeling. From an organizational perspective, further research is needed in the areas of adoption issues and business models that would ensure the sustainable support for of such systems in the service enterprise.

1. Introduction

Distributed model management systems (DMMSs) are a particular class of decision support systems with the focus on managing decision models throughout the modeling lifecycle [1]. They focus on supporting model management (MM) functionalities so that decision making models can be shared and re-used in distributed work environments, both on the inter-organizational and intra-organizational levels. However, to facilitate sharing and reusing models, it is paramount to be able to capture the underlying semantics for model representation and reasoning.

In that regard, semantic web technologies offer attractive solutions and mechanisms for researchers and practitioners who aim at developing semantically rich applications. Such applications enable semantic machine readability, which can overcome many challenges associated with information exchanges and thus, support large-distributed exchanges [4]. Such capabilities are further enhanced with the standardization offered by the World Wide Web consortium (W3C).

A number of applications that can benefit from such a distributed model management infrastructure can be envisioned. For example, decision models have a key role to play in electronic markets (e.g., online auctioning, e-commerce and targeted marketing, and self service travel sites), and modeling them as services facilitates their active role in supporting agile business processes underlying these services. Another example is the implementation of environmental decision support systems (DSS), where there is need to assemble various decision support components and models (e.g., land zoning model, hydrological model) for meeting the requirements of the problems at hand [2]. Yet another example application of DMMS is for supporting knowledge management in scientific community through sharing mathematical models that are developed independently and in a distributed manner, which often have unique semantic and syntactic requirements [3].

Given this motivation, the objective of this paper is to present the design and implementation of a semantic web-based DMMS, as well as capture the lessons learned during its development using existing semantic web technologies. The paper also presents evaluation of some of these tools in the context of MM and identifies opportunities for further research. The main approach for implementing the DMMS is based on service oriented design principles [4] and leverages web services technologies and distributed computing technologies in model selection, composition, and integration. The rest of the paper is organized as follows: Section 2 presents a discussion of related work in two complementary fields to our project: model management and the semantic web

978-0-7695-3869-3/10 $26.00

with its underlying state of the art standards and technologies. Section 3 presents an overview of the DMMS project in terms of the requirements, design, and implementation, followed by lessons learned in Section 4. Section 5 presents directions for future research. Finally, Section 6 concludes the paper.

2. Related Work

In this section, we provide an overview of relevant work in distributed model management and semantic web technologies. The application of these technologies for the development of DMMS is described in Section 3.

2.1. Model management

The domain of MM centers on the study of computer-based methods for representing models and automating modeling processes [5]. Work on MM emerged around the mid-seventies [6], since then, research has been going on to find new mechanisms by which decision models can be stored, formulated, selected, solved, composed, and integrated. MM systems are designed to provide access and control to various modeling resources. Such resources include model schemas and instances, model solvers, modeling platforms, and modeling languages such as the General Algebraic Modeling Language (GAMS).

Some of the functionalities of MM resemble those of database management systems (DBMS) such as model description, manipulation, and control [7]. One important function of MM is model selection. Model selection [8] focuses on identifying a model type or schema for a specific problem instance under consideration. For example, in case of a demand forecasting problem for a specific type of food in a grocery store, a time series model type may be more appropriate. When the historical values of demand for that particular food type are fed into the model, what results is termed as the model instance for that particular problem. After the model is implemented and solutions are presented, interpretation of the results comes to deal with the explanation of these results. Some techniques that are used include sensitivity analysis of the model to any structural changes in the model.

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. On one hand, model integration deals with orchestrating more complex models from two or more existing models at the structural or definitional level [9-10]. According to Tsai [11], interactions among the sub-models, which are simpler and easier to test, can provide an improved understanding of the new model, which in turn may be applied in more complex areas such as strategic analysis. On the other hand, model composition deals with sequencing models from the models library at the functional level [12].

Implementing most of the aforementioned modeling functionalities relies on the way models are represented. Model representation [13] research deals with the development of modeling language(s) that can accommodate graphic, tabular, forms of specification, and semantics.

Structured modeling (SM), originally developed by Geoffrion [14] was the harbinger for a new generation of model representation languages with more expressive power. It presents a popular computer-based framework for model representation that offers the capability of representing model structure as well as model semantics through deploying meta-modeling techniques. Accordingly, many researchers utilized SM formalism in developing large model bases and integrated modeling environments. For example, Dolk [15] applied SM and unified modeling language (UML) as a means for model representation in model warehouses. However, model consumers and providers dealing with such systems should have intimate knowledge about the models in order to register and deploy them in the integrated modeling environment. The object-oriented structured markup language [16] and the Structured Modeling Markup Language (SMML) [17] also deployed SM in their proposed model representation languages. These XML-based languages emphasize representing models in an interoperable format like XML. SMML is chosen as the primary model representation format during the development of DMMS in our work.

2.2. Semantic web technologies

At the core, the semantic web developments focus on providing semantic interoperability, over and above syntactic interoperability. Semantic web can be defined as “web of data described and linked in ways to establish context or semantics that adhere to defined grammar and language constructs” [18].

2.2.1 Ontologies, related standards and tools. Ontologies (e.g., Cyc for common knowledge such as time and space) provide a conceptualization mechanism or a vocabulary to represent knowledge in a particular domain [19]. Hendler [20] 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 [21].

Analogous to web resources, decision models in distributed context are resources that need to be exchanged over the semantic web. In the context of MM, reasoning about model inputs, outputs, as well as its schematic structure is essential for providing more accurate search results, whether these results will be simply presented to the knowledge worker or will be used in more complex services such as model composition and model integration. Thus, we start by exploring mechanisms by which the semantics of the web content is represented.

The Resource Description Framework (RDF) is a W3C recommendation that provides a semantic data model for describing resources on the Web in terms of named properties and values [22]. 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). Different formats such as RDF/XML, Turtle, and N-triples may be used to represent a RDF model. Tools (e.g., Altova SemanticWorks, project SIMILE at MIT) are available for RDF syntax validation and conversion. Also, frameworks like Jena allow programmatic manipulation of RDF data models.

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 [22]. 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) [22] further adds more vocabulary for describing properties and classes. Some examples include property type restrictions, equality, property characteristics, class intersection, and restricted cardinality. There are three sublanguages of OWL with different levels of expressiveness: OWL-Lite, OWL-DL and OWL-Full.

Recently, Semantic Web Rule Language (SWRL) [22] has been proposed by W3C as a rule language that can be used to write rules in terms of OWL concepts and can reason about OWL individuals. Examples of SWRL applications range from simply mapping identical concepts, to performing mathematical computations [18].

Domain ontologies such as those related to decision models (e.g., supply chain) can be developed based on the aforementioned standards. Ontology development tools, either standalone or plug-in components, can reduce the mechanical overhead associated with creating ontologies in different formats. Protégé, an ontology editor and a knowledge acquisition system, is a popular system developed by researchers at Stanford University [23]. The Protégé OWL API is also available to be deployed in semantic applications. Examples of other tools for developing and visualizing ontologies include: NeOn Toolkit, OntoEdit, and the WebODE Engineering platform. Along with these tools, different ontology reasoning engines are used such as FaCT++, Pellet, and RacerPro.

In order to query and discover semantic information, languages and tools for handling storage and manipulation of ontologies and semantic web content are essential. The SPARQL Protocol and RDF Query Language proposed by the W3C serves this purpose. SWRL also has a built-in library called the Semantic Query-Enhanced Web Rule Language (SQWRL) in order to support knowledge extraction from OWL ontologies. In Protégé, the SQWRL Query API provides a JDBC-like Java interface to retrieve the results of SQWRL statements.

From an application perspective, semantic Web programming frameworks (e.g., Jena, IODT by IBM Alpha works, SWeDE by BBN technologies, Visual Knowledge, and Semantic Studio by Semantic Soft) that provide an integrated support for building semantic Web-based systems have been proposed.

2.2.2. Semantic Web services. Web services are self describing, self contained software applications that are accessible over the Internet [24]. Distributed resources such as decision model schemas and instances may be shared in the form of model proxy services. Additionally, executable models may be interfaced as model web services themselves. Similarly, higher level model management activities such as model selection, model execution, model composition, and model integration can be provided as services, acting on model proxy services or model schemas in XML format (such as SMML) themselves as inputs. Related design issues are further discussed in Section 3.2. Relevant standards and current state-of-the-art in using web services are discussed here.

Web services are described in a procedural manner using the Web Services Description Language (WSDL), which captures their functional characteristics. Several approaches have been proposed in order to move towards the goal of adding semantics to web service descriptions. Submissions [25] to the W3C consortium exemplify these approaches: OWL Web Ontology Language for Services (OWL-S), Web Services Modeling Ontology (WSMO), Semantic Web Services Framework (SWSF), and Web Service Semantics (WSDL-S). Recently, W3C put forth a modified version of WSDL-S, Semantically Annotated WSDL (SAWSDL) as their recommendation [26].

SAWSDL provides a lightweight mechanism for extending WSDL service descriptions with additional semantics. It does so by making provisions to add extension attributes to any WSDL or XML schema elements - `modelReference`, `liftingSchema-Mapping`, and `loweringSchemaMapping`. The `modelReference` attribute points to semantic concepts in semantic data models such as domain ontologies, while the schema mapping providing data transformations between XML data model and the semantic data model. Currently, very few frameworks have been developed that support SAWSDL based semantic discovery of web services. Tools such as Radiant, Lumina, (Eclipse plugins), developed by researchers at the University of Georgia [27] were one of the first ones for annotating, publishing and discovering semantic web services based on SAWSDL. Kourtesis and Paraskakis [28] propose an approach - FUSION, based on classifying service descriptions at publication time, and mapping to UDDI for rapid lookups. Klusch and Kapahnke [29] propose a semantic matchmaking framework, SAWSDL-MX, that is based on logic-based matching as well as text retrieval strategies. Current research is moving in the direction of grounding other formalisms such as OWL-S in SAWSDL [30].

3. DMMS

The development of DMMS, based on the aforementioned technologies, is discussed next.

3.1. Requirements

The overall goal of the DMMS is to provide a service-oriented infrastructure for managing and sharing mathematical or decision models. Thus, the ability to share and reuse decision models is a core requirement for this system. From a distributed model management standpoint, several other issues and design requirements have been considered as driving forces: (1) a single model representation format [14], (2) representational independence of model structure and the detailed data [14, 31], (3) representational independence of model structure and the model solution [14, 31], (4) meta-modeling capability to support reasoning about models [31], (5) extensible for different modeling paradigms [14], and (6) accessibility of decision support resources [32].

The above requirements also emphasize the need to reason about syntactic as well as semantic knowledge embedded in models. Given the distributed nature of models, models present similar challenges like semantic web data in many ways. Moreover, models are not standalone entities, but are tied to other resources such as problem specific solvers, and are often expressed in different representational formats. In fact, the rich and complex nature of mathematical models makes development of DMMS-like systems a hard problem.

3.1.1 Use cases. Some typical use cases that reflect on the requirements from a user standpoint are mentioned below. While these use cases are by no means exhaustive, they intend to give a good idea of different kinds of anticipated uses for the DMMS.

- *Model Publication*: Knowledge workers create and annotate new models as well as compose and integrate existing models, save them as model schemas to be reused for particular problem types.
- *Model Discovery*: Knowledge workers discover models by using model discovery services, which retrieve relevant models based on user queries. The search results are not merely keyword based, but offer logic-based semantically rich results.
- *Model Selection*: Knowledge workers examine the search results and select most relevant models that best fit their problem. Decision makers then choose whether to invoke model executive services to solve the model, or solve them using their local software solvers.
- *Model Execution*: Knowledge workers provide model instances specific to their problem after selecting a particular model schema/type. The model execution services solve the model by invoking appropriate model solving algorithms.
- *Model Composition*: When a single model type does not provide a direct solution to the problem, the knowledge worker composes a query to a model composition service, which retrieves possible combinations of models that may be sequenced together to solve the problem at hand. The knowledge worker then picks the best combination of models and solves them.

3.2. Design

The overall architecture of the DMMS has been presented in Figure 1. In this section, the conceptual design of DMMS based on the architecture is presented. The discussion centers around the notion of semantically annotating decision models for reasoning and conducting various model management operations on them. Two key design issues are discussed, one relating to model representation, and the second related to model execution.

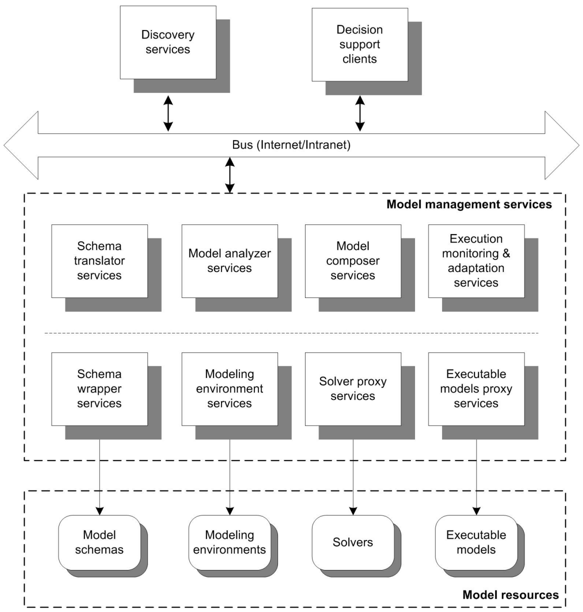

Figure 1: DMMS architecture

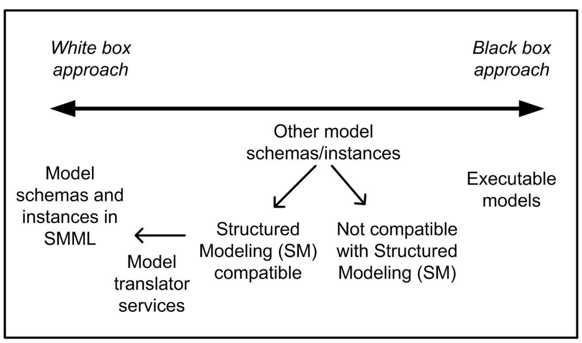

Figure 2: Compatibility of DMMS with different modeling approaches

The first design issue deals with model representation and semantic annotation. Decision models exist in various shapes and formats, and the DMMS system is designed to accommodate these differences across the spectrum. Figure 2 and Table 1 together illustrate these ideas. On one hand, models in a binary executable format do not provide access to the model structure, and are amenable to a so-called "black box approach". Such models are wrapped as model web services and any problem domain semantics associated resides in their SAWSDL description.

On the other hand, model schemas and instances represented using Structured Modeling Markup Language provide explicit access to the model schema and instance structures (and so the term "white box approach"). In addition, SMML has been extended to include the ability to link problem domain concepts to semantic models (e.g., domain ontologies) through semantic annotations, in a manner similar to SAWSDL. This extended model representation format is referred to as SA-SMML. For the purpose of model sharing, these models are encapsulated as model proxy web services. The model proxy web services are essentially "dummy" web services representing the model, and providing access to various parameters of the model through getter and setter operations. Additionally, operations for solving the model are also provided.

Models represented using higher level model representation languages other than SMML (e.g., LINGO, GAMS) lie midway along the spectrum shown in Figure 2. Some of these models may be amenable to be described using Structured Modeling, while other may not, depending on the decision problem they represent.

Table 1. Model representations, semantic annotations and model delivery

Model representation	**Model delivery**	
	Models as 'Services'	Models as 'Data'
	Model web services or proxy web services	Model 'source' files
Binary executables	SAWSDL	N/A
Higher level model representation		
SMML	SA-SMML	SA-SMML
Other model representation		
SMML compatible	SA-SMML[1] or SAWSDL[2]	SA-SMML[1] or SAWSDL[3]
Non-SMML compatible	SAWSDL[2]	SAWSDL[3]

[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 executing the model.
[3] SAWSDL is used to annotate a 'dummy' service representing the model. In addition to operations capturing the parameters of the model, there are operations for accessing and downloading the model.

Models that are amenable for Structure Modeling representation may be translated to SMML using model translator services, and the model semantics can be captured in SA-SMML. While the white box approach provides more information in the form of the internal model structure that can support better discovery and more importantly model integration, such an approach is infeasible in situations where structured representation of a model is not possible. For the class of models that may not be described using Structured Modeling, model proxy web services are created using their respective model representation format. The operations provide access to various parameters of the model, and the semantics resides in SAWSDL descriptions of such services. In the above description, models are essentially represented as “services”. In accordance with the principles underlying service orientation [4], models as “services” have key characteristics such as reuse, abstraction, autonomy, loose coupling, statelessness, composability, and discoverability.

The next design issue is that of considerations related to model execution. While binary executable models wrapped as model web services can readily provide model solutions, models dealing with more complex algorithms such as linear programming require access to model solvers (e.g., LINDO). As described in the overall DMMS architecture (refer Figure 1), various model management resources such as solvers, modeling environments are exposed as web services to operate on models. The design of DMMS supports this interaction in a couple of different ways. For models exposed as services, i.e. model proxy services, model execution services use model proxy services to feed model specific information to model solver services to derive a solution. Alternatively, model execution services are also designed to operate on models in manner similar to data. They are similar in notion to the optimization services proposed by Fourer, Ma, and Martin [33] in the COIN-OR project. The model “source” files, expressing the schema and instance information are provided as inputs to operations that invoke model solvers retrieve model solution(s).

This design approach accommodates models with different types and representation structure (see Figure 2), which attests to the completeness of the proposed design approach. Further, given that the Structured Modeling paradigm is founded on a sound theoretical basis, SMML models thereby inherits these features. Also, models delivered as semantic Web services leverage the semantic Web standards recommended by the W3C, which have gone through a peer-review process and are widely accepted in the industry. This attests to the validity of the approach.

3.3. Implementation

This section summarizes our efforts in the prototype implementation of DMMS, based on the architecture and design described earlier. Experiences with tools used during development are highlighted.

3.3.1 Building the model and the ontology repository. To experiment with different design and implementation issues, it was important to create a test bed at the onset. The test bed developed includes a repository of decision and semantic models. Decision models in different formats have been included that cover the spectrum shown in Figure 2. Following the white box approach, models described using XML representation of Structured Modeling [14], namely Structured Modeling Markup Language (SMML) [17] form a major portion of the repository. The models cover different levels of complexity, including mathematical programming, spreadsheet models, and predicate calculus models [34-35]. Mathematical operations represented as functions in structured models were represented using MathML using the MathType equation editor [36]. The process of populating the model repository was facilitated using XML editors (XMLSpy was used in this case), which significantly helped in lowering the manual model creation overhead for SMML models, particularly for complex model types such as optimization. Other than SMML, models described in other higher level modeling representation formats including LINGO, and MPS.

The repository also contains problem domain ontologies describing relationships between concepts used in the decision models. These OWL ontologies were developed using Protégé 3.4 [23].

3.3.2 Semantically annotating SMML models. In order to incorporate pointers to semantic models, SMML has been extended to Semantically Annotated SMML (SA-SMML), in a manner analogous to SAWSDL extension for WSDL [26]. Three key attributes have been added to `GenusType` and `ModuleType` type definitions in SA-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 are shown in Figure 3.

```
<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>
```

Figure 3. SA-SMML snippet illustrating semantic annotation of models

SA-SMML models in the repository reflect these annotations, thus linking to problem domain semantic models represented as OWL ontologies.

3.3.3 Developing web services for models and model management functions. Representative web services were next developed for encapsulating models using the different design approaches discussed in Section 3.2. A top-down approach for followed in creating the web services by first developing the corresponding WSDL descriptions. J2EE platform was chosen for this prototype development, given the availability of numerous open source technologies developed using Java that could be leveraged in the development process. Eclipse Integrated Development Environment (IDE) was used for the overall development. In particular, the Eclipse Web Tools Platform (WTP) project provided streamlined support for prototype building. Apache Axis2/Java implementation of the open source Apache Axis2 Web services engine was used for testing purposes, in conjunction with the Tomcat web server.

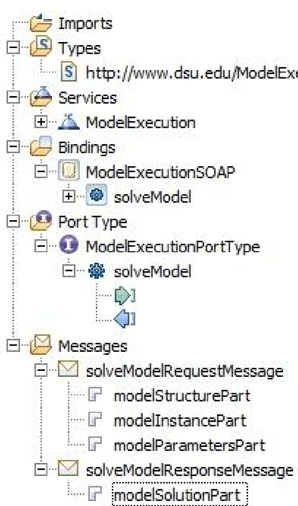

Figure 4. SAWSDL for model execution service

Next, WSDL descriptions were annotated to develop SAWSDL descriptions with semantic references to domain concepts (in cases where models are not represented in SMML). In the case of SMML models, SAWSDL descriptions provide semantic references to Structured Modeling concepts, whereas the SMML model schema itself captures semantic references to domain concepts. Tools such as Radiant and Lumina Eclipse plug-ins, discussed in Section 2.2.2 were explored. However, issues rooted in version incompatibilities precluded from effective use of these tools for the current project.

Other model management web services such as model execution web service were developed in a similar manner. Figure 4 shows an example of a SAWSDL for a model execution web service that consumes a SMML transportation model for providing solution through a solver.

3.3.4 Integration with Semantic Web Services Frameworks. Semantic discovery of published web services based on SAWSDL is key in effectively leveraging the encoded semantics for intelligent querying and retrieval of models, as well as supporting other model management functionalities such as model composition and integration. Currently, SWASDL-MX [29] and FUSION [28] frameworks discussed in Section 2.2.2 have been utilized in the context of DMMS. Experimental evaluation of these approaches is currently underway.

Model composition problem deals with generating an appropriate sequence of models by searching available model resources. It is analogous to the web service composition problem that has been studied in the web services domain. Particularly, OWL-S semantic web services framework has been shown to be particularly suited to the application of AI planning algorithms for model composition [37]. Implementation efforts are underway to create model composition services based on grounding OWL-S in semantically annotated service descriptions, i.e. SAWSDL, as discussed in [30].

4. Lessons Learned

During the design and the implementation of the DMMS, we have identified key issues that need to be taken into consideration in the development of such type of projects. These issues represent some guidelines for 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 business needs. Lessons learned from this project fall into two main categories: Technical and organizational.

4.1. Technical issues

Technical issues pertain to the technical aspects of the tools and technologies that we have tried to leverage so far in the DMMS, the evolving standards, disparate model representation formats, and modeling paradigms.

In evolving research areas such as the semantic Web, standards are yet maturing and constantly undergoing revisions. The Web Ontology Language (OWL) is a good example of this phenomenon. Ontology Inference Layer (OIL), based on the Description Logics (DL) knowledge representation paradigm [38], was earlier proposed as an extension of RDF/RDFS adding more frame-based representation primitives and eluding the RDF reification mechanism [39]. DAML+OIL, also based on DL knowledge representation paradigm [38], evolved from the earlier DARPA Agent Markup Language (DAML) attempting to combine the expressiveness of DAML and OIL by providing DL extensions of RDF/RDFS directly [40]. Web Ontology Language (OWL), is the result of the evolution of the DAML+OIL language, and has now become a W3C recommendation since 2004, and currently serves as the de facto standard for ontologies in the semantic Web [22]. The W3C OWL Working Group is currently receiving feedback on the candidate recommendation for OWL2, which is the next evolution step in that direction. Another example of such standards evolution is the Semantic Web Services set of standards, which was described earlier in Section 2.2.2.

The changes in the standards obviously reflect the steps taken to overcome shortcomings encountered in implementing these standards and applying them in real world contexts. It should then come as no surprise that the tools and technologies in this environment are constantly in flux to maintain compatibility with the current standards. While efforts by vendors and various research groups in developing semantic web technologies and tools are noteworthy, application developers relying on such tools to their own build applications, such as DMMS, nevertheless have to worry about their obsolescence and resultant incompatibility with other tools.

Another issue for model management systems such as DMMS relates to the lack of standardization for representing models by different vendors. While SMML offers a common ground approach, there exists a large overhead in creating translator services that can generate models in formats amenable to be consumed by different kinds of solvers, suited for particular decision problem types. During the development of the prototype, we focused on illustrating the feasibility of the concepts by creating a small set of translator services. Nevertheless, creating exhaustive support services is necessary for such a service infrastructure to gain momentum.

Last, but not least, support for conceptual modeling paradigms other than structured modeling is essential. While structured modeling has its advantages [10, 14], the white box approach discussed earlier has to be extended to represent models that are not amenable to structured modeling, such as continuous simulation models.

4.2. Organizational issues

Organization issues that arose in the context of current implementation effort can be classified into short-term issues directly pertaining to the development of DMMS and long term issues that will have to be addressed for wider scale implementation effort. Development issues pertain to processes and people. Specifically, with respect to development processes, we soon realized that prototyping and iterative development is a necessity. As noted in the technical issues section, despite the existence of web services standards, current technologies and supporting technologies are in a state of flux. It was often necessary to explore various design alternative using existing tools and modify the design and/or the technology, e.g., the development of SA-SMML, as necessary.

Moreover, the diversity of the technology imposed significant requirements on the skills and characteristics of the development team. In effect, programming skills in a particular language were insufficient. Additional knowledge and skills in developing XML applications, developing, semantically annotating, and deploying web services, developing ontologies, and developing mathematical programming models in various languages such as GAMS or LINGO are needed. Equally valuable is the ability to learn and assimilate new technologies and tools.

Current implementation efforts also pointed to issues that will have to be addressed for wider scale implementation effort. Most notably, are adoption and sustainability of such infrastructure. The significance of these issues will vary depending on the context, e.g., intra- versus inter- organization and the particular problem domain. Generally speaking, in an intra-organizational setting where problem domain ontologies already exist and models are recognized and 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, a critical assumption underlying the wide scale deployment of this infrastructure is the willingness of individuals or organizations to share their models as services. A number of issues may arise that can impede such effort:

- *Cost/benefit*: What is the cost to model providers for sharing models and what is the cost to users for using such models, relative to other options?
- *Ownership*: Model providers may not be inclined to share their models for fear of loss of ownership.
- *Confidentiality of models or data (model instances)*

Another issue potentially affecting the adoption and sustainability of such infrastructure is vendor support. Such support is needed to be able to provide language compilers and solvers as services. A business model will need to be developed that ensures that vendor interests are accounted for, e.g., for licensing.

5. Directions for Future Research

In the preceding sections we presented a number of lessons/issues that arose in the development of a distributed model management system for the semantic web. From a technical perspective, venues for future research and development include:

- Given the limitations of existing tools, it is paramount to develop tools specifically designed for facilitating the semantic annotation of models, e.g., via SAWSDL or SA-SMML and for deploying such models as services. Given the complexity of the underlying problem, it would be helpful to initially limit the scope to a certain class of models, e.g., mathematical programming models. This would allow for facilitating the process of capturing model semantics by capturing specific characteristics of this type of models
- Additional work is needed to specify and evaluate SA-SMML as a mechanism for semantically annotating SMML files.

From an organizational perspective, venues for future research and development include:

- Further exploring specific factors that affect the adoption of such system as well as the significance and ways for mitigating such factors.
- Exploring business models that would ensure the sustainable deployment of such system.

6. Conclusions

In this paper, we presented the design and implementation of a distributed model management system. 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 that will need to be addressed. While there are many arguments about the feasibility of the semantic web, both from theoretical and practical perspective [41], the proposed system and supporting technologies is an initial step in leveraging these technologies in the context of mathematical models as an organizational and national resource.

7. References

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

[2] 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.

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

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

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

[6] R. H. Sprague and H. J. Watson, "Model Management in MIS," in *Seventeenth National AIDS*, Cincinati, OH, 1975.

[7] D. R. Dolk, "Data as models: An approach to implementing model management," *Decision Support Systems,* vol. 2, pp. 73-80, 1986.

[8] S. Banerjee and A. Basu, "Model type selection in an integrated DSS environment," *Decision Support Systems,* vol. 9, pp. 75-89, Jan 1993.

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

[10] 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.
[11] Y.-C. Tsai, "Comparative analysis of model management and relational database management," *Omega,* vol. 29, pp. 157-170, 2001.
[12] K. Chari, "Model composition using filter spaces," *Information Systems Research,* vol. 13, pp. 15-35, Mar 2002.
[13] B. Konsynski and R. H. Sprague, "Future research directions in model management," *Decision Support Systems,* vol. 2, pp. 103-109, 1986.
[14] A. M. Geoffrion, "An introduction to structural modeling," *Management Science,* vol. 33, pp. 547-588, May 1987.
[15] D. R. Dolk, "Integrated model management in the data warehouse era," *European Journal of Operational Research,* vol. 122, pp. 199-218, 2000.
[16] H. Kim, "An XML-based modeling language for the open interchange of decision models," *Decision Support Systems,* vol. 31, pp. 429-441, 2001.
[17] O. F. El-Gayar and K. Tandekar, "An XML-based schema definition for model sharing and reuse in a distributed environment," *Decision Support Systems,* vol. 43, pp. 791-808, 2007.
[18] J. Hebeler, M. FIsher, R. Blace, A. Perez-Lopez, and M. Dean, *Semantic Web Programming*. Indianapolis, IN: Wiley, 2009.
[19] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins, "What are ontologies and why do we need them?," *IEEE Intelligent Systems,* vol. 14, pp. 20-26, January-February 1999.
[20] J. Hendler, "Agents and the semantic Web," *IEEE Intelligent Systems,* vol. 16, pp. 30-37, March 2001.
[21] D. Bell, S. De Cesare, N. Iacovelli, M. Lycett, and A. Merico, "A framework for deriving semantic web services," *Information Systems Frontiers,* vol. 9, p. 16, 2007.
[22] W3C, "W3C Recommendations: (1) Resource Description Framework (RDF): Concepts and Abstract Synatax, (2) RDF Vocabulary Description Language 1.0: RDF Schema, (3) OWL Web Ontology Language Semantics and Abstract Syntax, (4) A Semantic Web Rule Language (SWRL) Combining OWL and RuleML," http://www.w3.org/2001/sw/, 2004.
[23] H. Knublauch, R. Fergerson, N. Noy, and M. Musen, "The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications," in *Third International Semantic Web Conference - ISWC 2004*, 2004.
[24] D. F. Ferguson and M. L. Stockton, "Service-oriented architecture: Programming model and product architecture," *IBM Systems Journal,* vol. 44, pp. 753-780, 2005.
[25] W3C, "W3C Members Submissions: (1) OWL Web Ontology Language for Services (OWL-S), (2) Web Service Modeling Ontology (WSMO), (3) Semantic Web Services Framework (SWSF), (4) Web Service Semantics: WSDL-S," http://www.w3.org/2002/ws/swsig/, 2005.
[26] J. Kopecký, T. Vitvar, C. Bournez, and J. Farrell, "SAWSDL: Semantic Annotations for WSDL and XML Schema," *IEEE Internet Computing,* vol. 11, pp. 60-67, 2007.
[27] K. Verma and A. Sheth, "Semantically annotating a Web service," *IEEE Internet Computing,* vol. March-April, pp. 83-85, 2007.
[28] D. Kourtesis and I. Paraskakis, "Combining SAWSDL, OWL-DL and UDDI for Semantically Enhanced Web Service Discovery," in *The Semantic Web: Research and Applications*, 2008, pp. 614-628.
[29] M. Klusch and P. Kapahnke, "Semantic web service selection with SAWSDL-MX," in *CEUR-WS*. vol. 416, 2009.
[30] M. Paolucci, M. Wagner, and D. Martin, "Grounding OWL-S in SAWSDL," in *Service-Oriented Computing – ICSOC 2007*, 2008, pp. 416-421.
[31] 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.
[32] O. C. Ezechukwu and I. Maros, "OOF: Open Optimization Framework," Department of Computing, Imperial College London, Departmental Technical Report 2003/7, April 2003.
[33] R. Fourer, J. Ma, and K. Martin, "Optimization services: A framework for distributed optimization," 2007.
[34] A. M. Geoffrion, "The SML language for structured modeling: Level-3 and Level-4," *Operations Research,* vol. 40, pp. 58-75, Jan-Feb 1992.
[35] A. M. Geoffrion, "The SML language for structured modeling: Level-1 and Level-2," *Operations Research,* vol. 40, pp. 38-57, Jan-Feb 1992.
[36] R. Miner, "The importance of MathML to mathematics communication," *Notices of the American Mathematical Society,* vol. 52, pp. 532-538, 2005.
[37] E. Sirin, B. Parsia, D. Wu, J. Hendler, and D. Nau, "HTN planning for web service composition using SHOP2," *Journal of Web Semantics,* vol. 1, pp. 377-396, 2004.
[38] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. Patel-Schneider, "The Description Logic Handbook: Theory, Implementation and Applications," Cambridge, MA: Cambridge University Press, 2003.
[39] D. Fensel, I. Horrocks, F. van Harmelen, S. Decker, M. Erdmann, and M. Klein, "OIL in a nutshell," in *Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling, and Management (EKAW '00)*, R. Dieng, Ed., 2000, pp. 1-16.
[40] F. van Harmelen, P. Patel-Schneider, and I. Horrocks, "Reference description of the DAML+OIL ontology markup language," http://www.daml.org/2001/03/referenceMarch 2001.
[41] J. Hendler, "The dark side of the semantic web (Editorial)," *IEEE Intelligent Systems,* pp. 2-4, January/Febraury 2007.