Model Driven Multimedia Development Description

Haeng-Kon Kim

Department of Computer Engineering, Catholic Uniersity of Daegu, Korea
hangkon@cu.ac.kr

Abstract

Modeling has traditionally been an important part of the multimedia software development process as it allows developers to tackle complex problems by using abstraction and hiding technical details. A models goal is to predict problems early on in design phases to avoid problems during implementation. Since defects can be detected earlier on in the software development process, the use of models reduces maintenance costs. Model-based software development is used for more than just visualizing the code or design of the system; it is used to produce source code directly from the model.

Traditional model management and service system which is the key component of DSS only supports the single format model in the special application field. Based on requirements of multi-model-aided decision, this paper designs the general model management and service system which provides the general management and service for different format models on network. The system proposes the solution to manage different format models by providing model frame generator which aims to analyze decision problem top-down and model application frame which aims to integrate model function bottom-up.

Keywords: *Multimedia Information System, Decision Support System, Model-aided Decision, Model Base, Model Management*

1. Introduction

Traditionally modeling was used for showing communication between stakeholders and providing a diagram of a multimedia software system [1, 2]. Today modeling is used for those reasons but also for simulation, generating test cases and generating source code. Modeling has become so widely used that formal approaches were developed such as Entity Relationship Diagrams (ERD), Specification Description Languages (SDL), and the Unified Modeling Language (UML). In late 2000, OMG (Object Management Group) came out with MDA (Model-Driven Architecture) to promote the use of models in software artifacts. Over time the process has evolved to include multiple stakeholders and higher levels of abstraction.

There are several advantages to using model-based software development. The model can be set up for a specific programming language so code can be auto-generated. Models can be used to predict bottlenecks and constraints to save on errors during implementation. The developer can achieve higher levels of abstraction implementing the system in models before coding. Model-based development encourages good object oriented design and provides an easy to read model for both the shareholder and stakeholder alike. One of the more important advantages is the detailed history of the auto-generated code, which is crucial when it comes to code verification

Model management, one of the key elements of DSS (Decision Support System) is studied widely and gained much attention. Researchers have designed several model representations

on the basis of advanced technologies of software and hardware, and implemented many systems which are used on model management, composition and model services [1, 3-8]. Though the series of significant productions have provided the solutions of the model-aided decision at present, there are hardly any provide general management and service for the different format models on network.

As information industry develops, model-aided decision is paid more and more attentions. Lager numbers of the resources of the basic models and profession models are the foundation of model-aided decision, while arise unprecedented challenge, for example, since the models are developed and rebuilt in different technology architectures, which cause the distributed locations and heterogeneous techniques. However, few solutions are effective to manage model resources whose formats are different on network. The model resources as stationary terms in control process, which are developed for special application and used in one way, are not shared and used sufficiently. Therefore it is necessary for the model management and services system to manage models whose formats are different on network and provide several assorted solutions to share models.

For actual production's deficiencies, under model-aided decision's requirements, this paper designs and implements the general model management and service system which manages several models compiled to different formats such as DLL, COM, EXE and Web Services and provides two mechanisms to support these functions. One aims at decision problem in the way of top-down and the other aims at functions integration in the way of bottom-up. The system provides the solution to share and apply the heterogeneous decision models.

2. Related Works

2.1. Model Execution

Modeling for execution is not a new concept, as this approach has been used in traditional software and system development for a while. This technique always starts with a high level platform independent model (PIM) that is often expressed in some visual language. This model might then be translated or transformed into a platform specific model (PSM) in a particular programming language for a particular platform, which then can be compiled or interpreted for execution. In fact, this methodology uses the principles of Object Management Group's (OMG) Model Driven Architecture (MDA). Figure 1 depicts the relationship between the models.

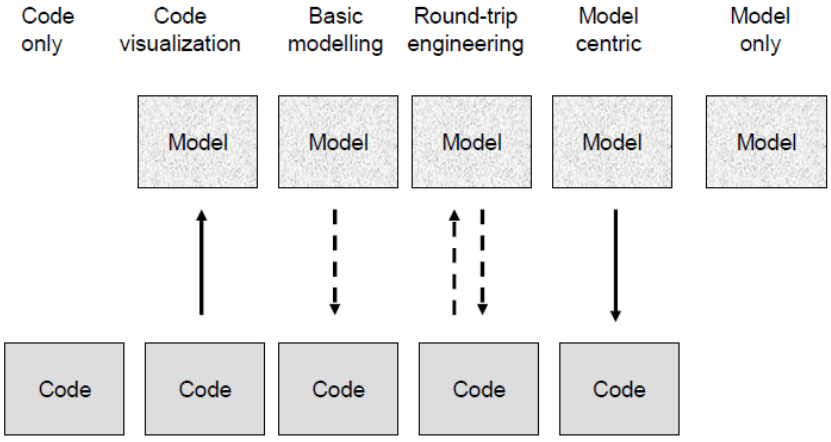

Figure 1: Model-Driven Multimedia Software Development Adoption Spectrum

The model-driven architecture starts with the well-known and long established idea of separating the specification of the system's operation from the details of how that system uses the capabilities of its platform. The three primary goals of MDA are portability, interoperability, and reusability through architectural separation of concerns. BPM solution delivery is a new business-driven and process-centric way of delivering applications. It can and should use the best practices we have learned over decades of software development and system engineering. MDA principles can certainly apply here, but because BPM solution delivery is unique in that it puts more emphasis on iterative business design and requires more involvement of business users and analysts, there are special considerations when adopting modeling for execution techniques. Many companies have been using modeling for execution methodology and techniques for solution delivery based on the WebSphere BPM product suite. Figure 2 illustrates the typical transition from business model to technical business model to implementation model.

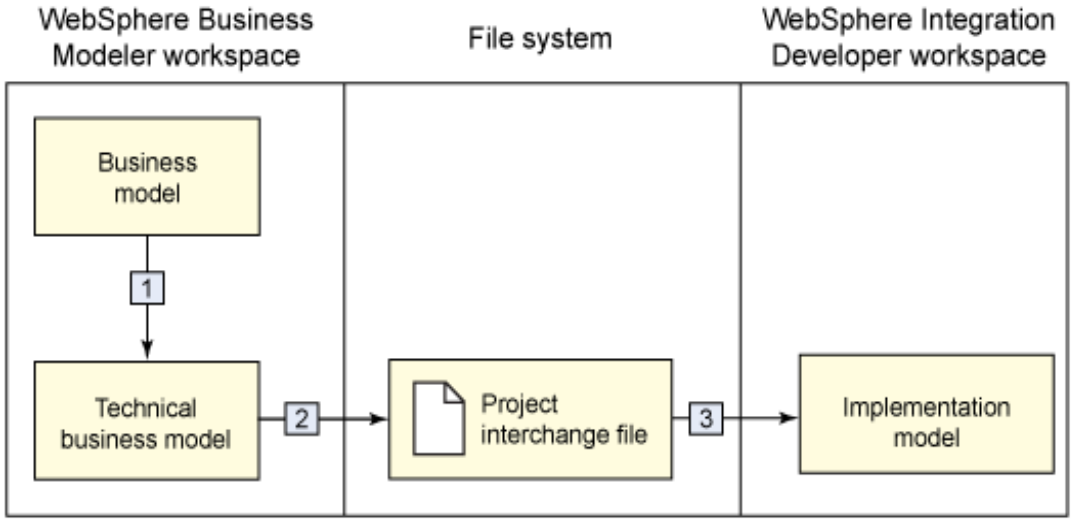

Figure 2. Business Model to Multimedia Implementation Model

A business analyst or business process architect creates a business model in Business Modeler in either the Basic or Advanced mode. The business model graphically represents the business process and uses semantics relevant to business analysts and subject matter experts. The business model might also be used for simulation.

A technical business analyst or technical process architect refines the model in Process Server mode, producing a technical business model that graphically represents the business process for implementation and begins to translate the business semantics into technical semantics. The technical business analyst exports the technical business model from Business Modeler as a project interchange file, which is a .zip file containing all the run time artifacts that make up the implementation model.

A process architect or integration developer imports the implementation model into Integration Developer. The implementation model graphically represents the business process to be implemented. The integration developer further refines the model and completes translating the business semantics into technical semantics.

2.2. Multimedia Model Description Information

Model description information which is foundation of model management and shared services should be describable, practical and detailed. There are three levels to describe model information: syntax, semantic and pragmatics as in Figure 3.

Model pragmatics information which is about model classification, model function and version contain the following information:

-Model name;

-Model information about the application

domain and classification, by which models are describe from several different aspects;

-Model function;

-Model version.

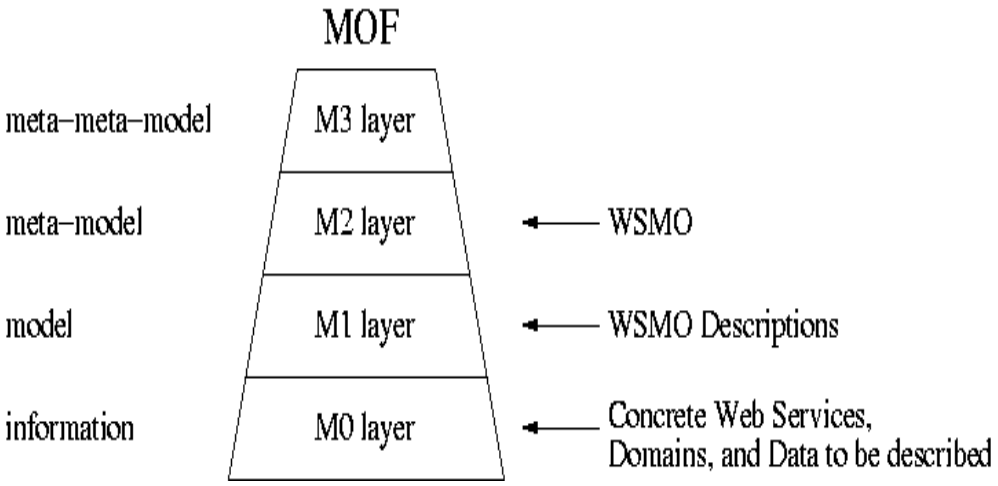

Figure 3.Multimedia Model Description Method

Model semantic information which is about model principle, instruction of the development, results of check and interface parameters information contain the following information:

- Model principle;
- Model instruction of the development;
- Developers information;
- Version modified information;
- Authorization information;
- Verification information;
- Information of methods provided by model and whose function;

- Information of in-out parameters
- Information about files associated the model such as data files, help files, source files, reference files, temp files, etc. Model syntax information which is about model's formats, orientations and grammatical structure of the in-out parameters contain the following information:
- Information about environment under which models are developed and running.
- Model formats;
- Location information of the model resources such as IP and port of the model service provider, file path, web services path and so on;
- Information about how to invoke the methods of the model such as class name, method name, the count of parameters and so on;
- In-out parameters information such as ID, input or output tag, name, type, length, precision, default value, max value, min value and so on.

We classify the model description and services component for interaction of MDA. The model description pattern consists of a meta model for MDA infrastructure, with the service interaction points, or end points, shown in Table 1.

Table 1. Multimedia Model Description Design

Model name	Based on the meta-model of the message under consideration (The ESB should be capable of supporting different types of message models flowing between the service provider and service requestor, thus creating a message-model-agnostic exchange.)
Model Information	Capabilities to federate, replicate, and transform disparate data sources
Model function and version	Control capabilities to manage the message flow and interactions across multiple services according to business processes and flows.
Model Interaction	Capabilities and functions to deliver content and data using a portal, or other related Web technologies, to consumers or users.
Model Partner	Capabilities to integrate partner electronic data interchange (EDI) and legacy systems into the corporate enterprise architecture
Model Application	Capabilities for service consumers to be called by the business application services
Model Access	Capabilities to integrate core applications with external data repositories and packaged applications

3. Model Driven Multimedia Development Description

3.1. System Design

On the basic principle and design philosophy, centralized management, unified service and flexibly configure, the general model management and service system is divided into four tools: model description tool, model information management tool, model application tool and model service tool, and is deployed into three nodes on network: model information management node, model service node and mode application node. Figure 4 describes the system which is flexibly configured based on actual environment and situation: (1) the database servers are either configured on the separate node or attached to other nodes; (2) the minimum system which can be found by these nodes that integrated into one computer; (3) there can be several model application nodes and model service nodes in one system.

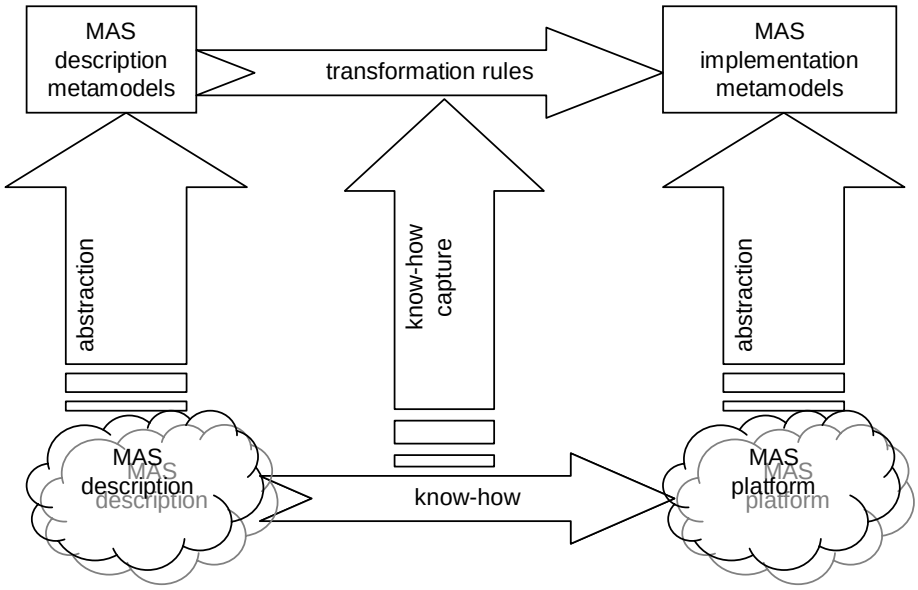

Figure 4. System of Multimedia Model Description

1) Model description tool. It semiautomatic converts the model resource such as DLL, COM, EXE and Web Services to the standard model information description files on the model service node, and automatic generate standard managed code based model resource, which is foundation of providing unified model service.

2) Model information management tool. It is located on the model information node, whose main functions are as follows: (1) receive the files noted about model description information from model service nodes and register the standard model description information into the model information base; (2) logout model description information from model information base passively or initiatively; (3) find and select model description information from model information according to users' requirement; (4) update model description information by sending requirement from the model service nodes; (5) to classify and apply models, build and modify classification system dynamically.

3) Model application tool. It provides two mechanisms to support model-aided decision on model application nodes: (1) as model functions are converged, the tool allows to assemble available models bottom-up into combination models to solve decision problem, just like building block. Reference [9] presents the principle and process. (2) As analyzing decision problems, the tool allows disassembling decision problems step by step top-down into several interactional meta-problems which are associated to available models, and then automatic generates combination models frame accorded with decision problems solution logic by this way, Reference [10] presents the principle and process. The first mechanism can generate more flexible and abundant model sources based actual available models; the second mechanism does accord with the human habit of thought. These mechanisms complement and bring out the best in each other. The following are the main functions of this tool: (1) provides the visible editor for the control flows and information flows of the combination models; (2) automatic generates script to control the combination models; (3) provides the mechanism to interpret and submit service requests of the meta-models to the related model service nodes.

4) Model service tool. It is located on the model service node, whose main functions are as follows: (1) listens and receives the service requests of the models from the model application nodes; (2) analyzes the service requests of the models, and as setting the in-out parameters of the running models, forms the standard running orders of the models; (3) manages the running of the models by service engine and records the results and logs of the models running; (4) according to the flow direction of the output parameters, sends the results of running to the model application nodes.

3.2. Generation of PIM Code

There are several carriers of the models by which aid to decision. The formats supported by the following system: (1) the standard dynamic link library (DLL); (2) Component Object Model (COM) whose format is DLL or EXE; (3) the standard executable program (EXE); (4) Web Services.

On the one hand, it is the foundation of centralized management to describe the model by the standard, which is also the first step of model management and service. The system must be adequate to automatic analyzes and extracts model information, particularly model syntax information, in order to ensure model description information in a uniform standard and convenient to register models for users. On the other hand, as model invoking modes of the four carriers are different, it is necessary to generate intermediate code which is identified with standard service interface for implement the principles, centralized management.

Reflection mechanism which is the key feature of .NET framework is used to get information of the type member of .NET, such as information about methods, attributes, properties, events and constructors and so on, and create and invoke the instances by the meta-data. The system implements the function to generate automatic managed code and extracts model description information by using Reflection mechanism and some tools of the Visual Studio.Net (such as TlbImp.exe, dumpbin.exe and wsdl.exe), and then carries out the principle, centralized management and unified service.

It is necessary for each of carriers of the models to use corresponding tool to generate the managed code and extract model information. Figure 5 shows the domain model driven process:

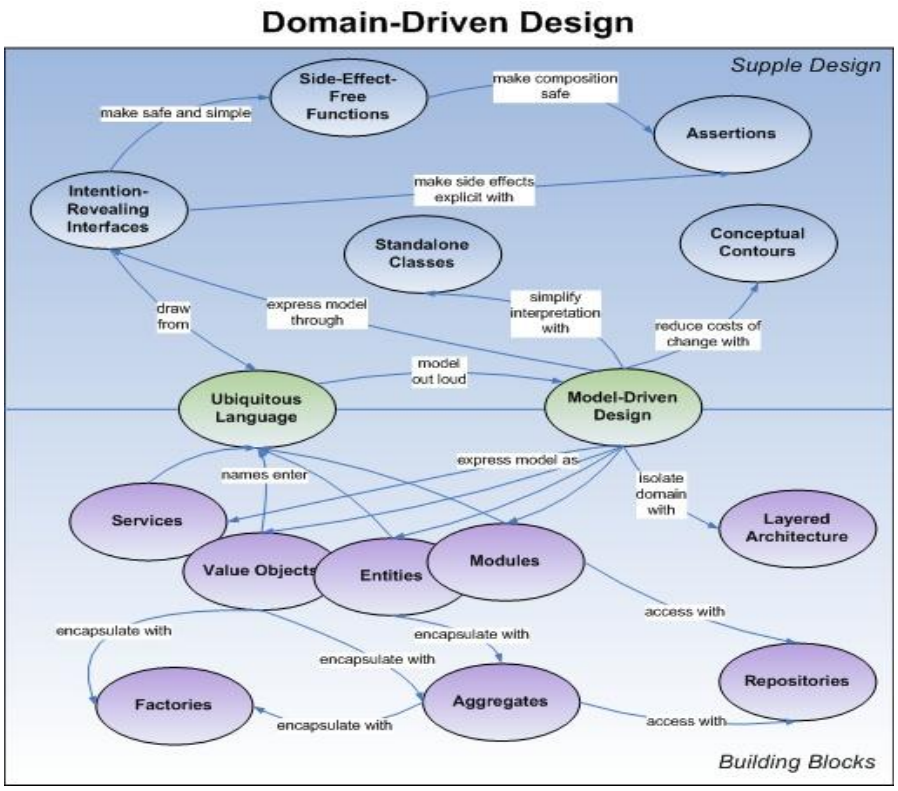

Figure 5. Multimedia Model Description Process

1) COM model: extracts and analyzes the information of the COM file by using Tlbimp.exe, and then transforms to the assembly in the .NET Framework.

2) Dynamic link library model: extracts and analyzes the information of the DLL file by using Dumpbin.exe, and then transforms it to the assembly in the .NET Framework.

3) Web Services model: extracts and analyzes the information from the WSDL file, XSD file or .discomap file by using Wsdl.exe, and then transforms it to the assembly in the .NET Framework.

The process of generating the corresponding managed code by using technologies of the reflection is as follows:

1) Defines and load the assembly, and then creates instance by the Assembly type.

2) Gets available classes and global methods from the assembly by the Module type.

3) Gets the constructor's information such as name and parameters and so on, and then creates instance by using GetConstructors method or GetConstructor method of the Type type.

4) Gets the methods' information such as name, parameters and the type of return value and so on, and then gets the methods by using GetMethods method or GetMethod method of the MethodInfo type.

5) Gets the events' information such as name, corresponding procedure and the type and so on by the EventInfo type and add or remove event handler.

6) Gets the properties' information such as name and type and so on, and then sets or gets value by using PropertyInfo type.

7) Gets the parameters' information such as name, type and location and so on by the ParameterInfo type.

3.3. Model Application Frameworks

The decision problem often is the complex problem which is formed by a series of sub-problems and only can be solved by multi-models. There are two model application frames provided in the system as in figure 6. (1) Aimed at converging model functions, the tool allows assembling available models bottom-up into combination models to solve decision problem. (2) Aimed at analyzing decision problems, the tool allows disassembling decision problems top-down, and then automatic generates combination models frame. These functions are implemented under the principle that the operation interface and flow keep in line.

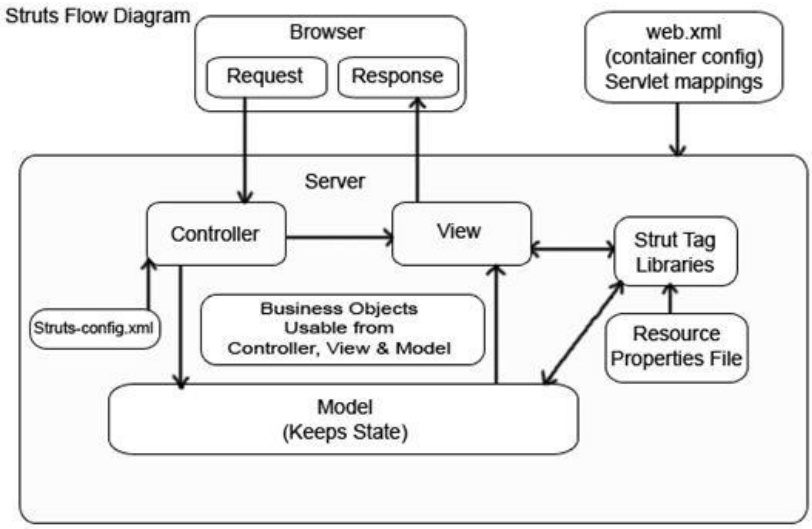

Figure 6. Multimedia Model Application Frameworks

1) Visual editing module. It is the visual editor aimed at assembling models and disassembling decision problems, by which information flow and control flow are edited. The properties of the decision problems and combination models are described by problem description sub-module. The relationships among the sub-problems' or sub-models' sequence, selection and circulation, are edited visually to generate the control flow which is consistent with the solutions of the decision problems by control flow definition sub-module. As the standard and descriptive information of the models is displayed by the model selection sub-module which is under the principle that selection is dominant by users and prompt is assisted by system, users can select the suited models. The relationships of the in-out parameters are set by information flow definition module.

2) Script generating module. The essential part of scrip generate is to make use of the information and relationship of the meta-problems to describe the user view. In the process, it is necessary to check whether the parameters and configuration of the model frames and combination models are logical and integrated, and then based EBNF format, generates .XML

script files which are logical and clear and can be validated. The model script is provided for the sophisticated users, which is shown in reference [9].

3) Service requesting module. It sends service request messages which are generated according to the control and information flow in scrip of machine view and parameter information to model service node, and then receives the results.

The rules of design and implementation of the system are as follows.

1) The rules of the standard control flow. The three basic structures between sub-models or sub-problems are sequence structure, selection structure and circulation structure according to structuring theory studied by G. Jacopini and C. Bohm. (1) Sequence structure. It means that the sub-models and sub-problems are executed in sequence, and parallel structure can be transform to sequence structure. (2) Circulation structure. It means that the sub-models and sub-problems are executed repeatedly under certain conditions. (3) Selection structure. It means that sub-models and sub-problems are executed under the certain conditions. Although these structures can be nested each other in theory, the nested level should be simplified.

2) The rules of the data preprocessing function modeled. The type, length and precision of the in-out parameters are different from each other in the process of parameters transmission. Therefore, it is necessary to implement data preprocessing such as the data transformation and data cleaning etc. in the process of defining information flow. The burden of the visual editing module is relieved and the procedure of decision problem solving becomes legible and pellucid by modeling the data preprocessing function.

3) The rules of the in-out parameters. The type of the parameters which express the connection character string is restricted as string type, and the read-write operations are executed by the models, when the models transfer parameters by the database. The type of the parameters which express the path of the data file is restricted as string type, and the operations of file are executed by the models, when the models transfer parameters by the data file. The parameters are needed to transform to data file or database, when the models transfer parameters by the multidimensional array.

4. Conclusion

Model-Based Multimedia Software Development has many advantages over its few disadvantages. The idea of automatic code generation and effective communication between shareholders and stakeholders is something that is continually being researched. With the right process, Model-Based Development can conquer high complexity systems with more modeling and less coding ultimately being an effective way to develop software. Model-aided description is the key of enterprise modeling and decision support systems. Based on present available technology system, the general model management and service system is implemented in this paper by the author, which provides the functions for the different format models to manage and service on network and automatic analyze, extract and manage the model information. The two model application frames of the system are provided, which is model functions assembling and models frame generating. As the increasing of the model resource, decision problem become more complex, in future work we plan to study how to select model automatic and share model in hybrid environment.

References

[1] A. Martens and K. Heiko, "Automatic, Model-Based Software Performance Improvement for Component-based Software Designs", http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.152.8697, **(2007)**.

[2] D. Niz, "Diagrams and Languages for Model-Based Software Engineering of Embedded Systems: UML and AADL", from http://www.sei.cmu.edu/library/reportspapers.cfm, **(2009)**.

[3] D. Ziegenbein, R. Bosch and P. Braun, "AutoMoDe - Model-Based Development of Automotive Software", http://ieeexplore.ieee.org/Xplore/, **(2005)**.

[4] M. Baleani, A. Ferrari and L. Mangeruca, "Correct-by-Construction Transformations across Design Environments for Model-Bases Embedded Software Development", http://ieeexplore.ieee.org/Xplore/, **(2005)**.

[5] M. Parastoo, D. Vegard and N. Tor, "Definitions and Approaches to Model Quality in Model-Based Software Development-A Review of Literature", http://portal.acm.org, **(2009)**.

[6] W. Grieskamp, N. Tillmann and M. Veanes, "Instrumenting Scenarios in a Model-Driven Development Environment", http://research.microsoft.com/apps/pubs/default.aspx?id=77816, **(2004)**.

[7] The Middleware Company, "Model Driven Development for J2EE Utilizing a Model Driven Architecture (MDA) Approach", http://www.omg.org/mda/mda_files/MDA_Comparison-TMC_final.pdf, **(2003)** June.

[8] C. Buckl, A. Knoll and S. Gerhard, "Model-Based Development of Fault-tolerant Embedded Software", http://ieeexplore.ieee.org/Xplore/, **(2007)**.

[9] S. Bohner and S. Mohan, "Model-Based Engineering of Software: Three Productivity Perspectives", http://ieeexplore.ieee.org/Xplore/, **(2009)**.

[10] P. Bowler and T. Gordon, "Model-Based Software Engineering", http://ieeexplore.ieee.org/Xplore/. **(2008)**.

[11] J. Jacky, M. Veanes, C. Campbell and W. Schulte, "Model-Based Software Testing and Analysis with C#", http://www.cambridge.org/gb/knowledge/isbn/item1163498/?site_locale=en_GB. **(2008)**.

[12] S. Miller, M. Whalen and D. Cofer, "Software Model Checking Takes Off", from http://portal.acm.org. **(2011)** January.

[13] Microsoft. Model-Based Testing. http://msdn.microsoft.com/en-us/library/ee620469.aspx.

[18] Microsoft Model-Driven Development. http://msdn.microsoft.com/en-us/library/aa964145.aspx#mdldrvdv_topic1

[19] IBM. An introduction to Model Driven Architecture. http://www.ibm.com/developerworks/rational/library/3100.html