

# MDS++: Supporting Ontology-based Dynamic Classification in WebSphere Metadata Server

Shengping Liu<sup>1</sup>, Yang Yang<sup>1</sup>, Guotong Xie<sup>1</sup>, Chen Wang<sup>1</sup>, Feng Cao<sup>1</sup>, Cassio Dos Santos<sup>2</sup>, Bob Schloss<sup>3</sup>, Yue Pan<sup>1</sup>, Kevin Shank<sup>2</sup>, John Colgrave<sup>2</sup>

<sup>1</sup>IBM China Research Laboratory  
Zhongguancun Software Park, Beijing, 100094, China

{liusp, yanggy, xieguot, chwang, caofeng, panyue}@cn.ibm.com

<sup>2</sup>IBM Software Group  
50 Washington Street, Westborough, MA 01581, USA

{scdos,kshank}@us.ibm.com, colgrave@uk.ibm.com

<sup>3</sup>IBM Watson Research Center  
P.O.Box 704, Yorktown Heights, NY 10598, USA

{rschloss}@us.ibm.com

## ABSTRACT

Metadata management systems are growing from tool-specific repositories to enterprise-wide metadata repositories. In this context, one challenge is the management of the evolving metadata whose schema or meta-model itself may evolve, e.g., dynamically-added properties, which are often hard to predict upfront at the initial meta-model design time; another challenge is to organize the metadata by semantically-rich classification schemes. In this paper, we demonstrate a practical system which provides support for users to dynamically manage semantically-rich properties and classifications in the IBM WebSphere Metadata Server (MDS) by integrating an OWL ontology repository. To enable the smooth acceptance of Semantic Web technologies, the system is designed to consist of integrated modeling paradigms, with an integrated query language and runtime repository.

## 1. INTRODUCTION

Within the enterprise-wide IT environment, metadata management has become more and more challenging because of rapidly-changing business requirements. Metadata repositories are growing from tool-specific, application-specific systems to enterprise-wide, asset-management and architecture decision support systems, in which metadata are shared and integrated across multiple applications or even third party tools [1]. While the metadata and their relationships dramatically grow, it is impossible to design a unified meta-model for all kinds of metadata with all possible attributes and relationships at design stage as the business requirements evolve. Therefore there is a requirement to dynamically add properties for classes in the registered metamodel. For example, after a WSDL meta-model which describes WSDL documents has been registered, a service administrator might add QoS (Quality of Service) metadata to the “WSDLService”, such as the “responseTime”. Another example is to dynamically build particular relationships across registered meta-models. After the metadata repository has run and collected entries for a period of time, a user needs to create a dynamic relationship “dependsOn” from the class “Activity” in the business process meta-model to the class “WSDLService” in the WSDL

meta-model, which later can be used to enable traceability and impact analysis across those models. Moreover, semantic annotations are required to enrich the semantics of dynamic relationships, e.g. annotating “dependsOn” as “transitive”.

In metadata management, a classification scheme is used to classify the metadata objects in a metadata repository. Examples of classification schemes range from simple tags (keywords), thesauri, taxonomies to formal ontologies. With the growing volumes of metadata in different applications and users of metadata from various business units of enterprises, a flexible and semantic-rich classification scheme is needed to help different users to organize metadata from different viewpoints. This is because: (1)the classification scheme itself needs reasoning on the classifier hierarchy; (2)users need to define rich expressions on the classification scheme to declare dynamic classifiers, in addition, the expression can be defined on dynamic properties. For example, after the dynamic property “dependsOn” is declared, user can define “DataDependentService” as a new classifier, which contains the WSDL services that depend on a “DataService”.

With the emergence of the Semantic Web [5], Web metadata markup languages, i.e. RDF (Resource Description Framework) and OWL (Web Ontology Language), have become W3C Recommendations. RDF originated from the W3C Metadata Activity, and is particularly intended for representing metadata about Web resources, such as the title and author of a Web page. The most important feature of the RDF data model is that it treats properties as first-class citizens and allows them to be attached to a class dynamically. OWL is a formal logic language to define the vocabularies in RDF documents. It is intended to represent structured metadata ranging from a simple taxonomy, a thesaurus, or to a formal ontology. In practice, OWL is an emerging standard to represent the classification schemes, because of its rich expressivity, formal semantics and reasoning capabilities. Therefore, it is natural to apply the Semantic Web technologies, namely RDF and OWL, to meet the emerging requirements of enterprise-wide metadata management.

In this demonstration, we show a practical system which provides support for users to dynamically manage semantically-rich

properties and classifications in the IBM WebSphere Metadata Server (MDS) by integrating an OWL ontology repository. To enable the smooth acceptance of Semantic Web technologies, the system is designed to consist of integrated modeling paradigms, with an integrated query language and runtime repository.

## 2. SYSTEM ARCHITECTURE

To support ontology-based dynamic property and classification in metadata management, one approach is to store all the structured metadata and data on dynamic properties and classifications into an ontology repository. However, current state of the art RDF databases (triple-stores) scale poorly since most queries require multiple self-joins on the vertical triples table [4], and the large volumes of structured metadata in enterprise are commonly stored in horizontal tables in a relational database. So we chose the hybrid approach that integrates the MDS with an ontology repository SOR [2], i.e., the structured metadata are still stored in the horizontal tables of MDS and the data on dynamic properties and classifications are stored in the vertical triples table of SOR. This system is called MDS++. The system is designed to consist of integrated modeling paradigm, query language and runtime repository. Fig. 1 shows the overall system architecture.

For the modeling paradigm, MDS was developed using Model-Driven Architecture and is based on EMF. The SOR architecture follows the model-driven approach for ontology engineering [3]. In this approach, the RDF and OWL is defined based on Ontology Definition Metamodel (ODM), the ODM specification is implemented by EODM, based on the EMF framework with additional inference and model transformation functions. From Fig. 1, we can see that MDS and SOR provide a unified EMF view for users to access the metadata through query and CRUD API.

For the query language, we extend the XMQL query language of MDS to XMQL++, with additional classification query functions and dynamic property extensions. Users can simultaneously query over the structured metadata stored in MDS, the information about dynamic properties and classifications stored in SOR.

For the runtime repository, we make the ontology repository tightly-coupled with the MDS repository, i.e., the tables of ontology repository will be deployed in the same database with MDS and are visible to MDS. The two repositories are connected by the unified object identifier in MDS, which is also used as the internal identifier for the RDF resource in SOR. When a hybrid query is issued, the system will translate the query to one single SQL query which will access the tables of both repositories simultaneously. The advantage of this approach is that it provides high performance because queries are translated to SQL queries which can fully utilize the optimization provided by the underlying DBMS.

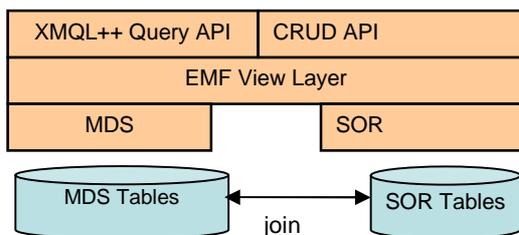


Fig. 1. The system architecture of MDS++

## 3. DEMONSTRATION

To illustrate the feature of MDS++, we design a scenario of using MDS++ regulating WSDL models. In the scenario, different roles of users can declare dynamic properties to WSDL meta-model which has been registered and has run for a period of time according to their needs. E.g. a service administrator would like to add “responseTime” information to WSDLService object; a business analyst would like to add “dependsOn” relationship between WSDLService. He can also build up “usedBy” relationship across different meta-models to capture impact analysis information. User can also build up ontologies to classify the metadata in the MDS++. For example, a business analyst can use ontology about service functions to classify WSDLService objects, while a service administrator can use ontology about life cycle of services.

After setting value for dynamic properties and classification information, user can use XMQL++ to query the repository. Automatically reasoning will be made by the system based on the semantic annotations of dynamic properties and the ontology definition.

We also developed an eclipse-based UI to demonstrate the scenario. User can use the simple UI program to declare dynamic properties, build up classification ontologies, assign value of dynamic properties, classify objects and submit XMQL++ queries. Fig. 2 shows a snapshot of the UI program.

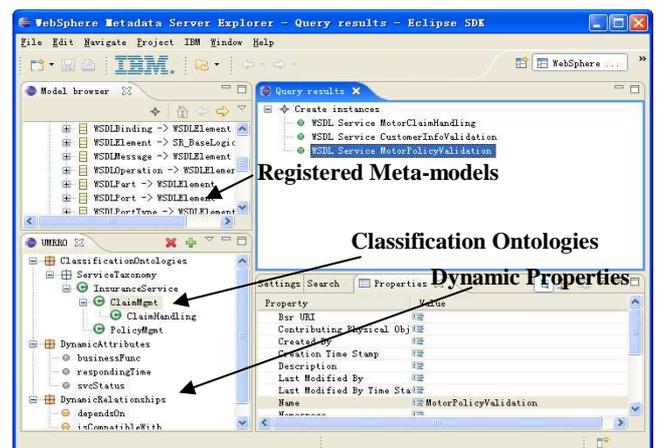


Fig. 2 . Snapshot of UI Program

## 4. REFERENCES

- [1] Arun Sen, Metadata management: past, present and future, *Decision Support Systems*, 37(1), 2004.
- [2] Li Ma, Lu Jing, Chen Wang, Feng Cao and Yue Pan. *Effective and Efficient Semantic Web Data Management over DB2*, in Proc. of the 28th ACM SIGMOD Conference, 2008.
- [3] Yue Pan, Guotong Xie, Li Ma, Yang Yang, ZhaoMing Qiu, and Juhnyoung Lee. Model-driven ontology engineering. *Journal of Data Semantics VII*, pages 57–78, 2006.
- [4] Abadi, D., A. Marcus, et al.. *Scalable Semantic Web Data Management Using Vertical Partitioning*. In Proc. of VLDB 2007, pp. 411-422, 2007.
- [5] Tim Berners-Lee, Jame Handler, and O. Lassila. The semantic web. *Scientific American*, 184(5): 34–43, 2001.