SemanticWare: An EMF-Compatible RDF Infrastructure

Guotong Xie, Shixia Liu, Zhuo Zhang, Li Ma, Yue Pan
Li Zhang, Zhong Su, Li Qin Shen
IBM China Research Lab, 2/F, HaoHai Building, No. 7, 5th street, ShangDi, Beijing 100085, PRC
{xieguot, liusx, zhuoz, malli, panyue,lizhang,suzhong,shenlq}@cn.ibm.com

Extended Abstract

The Resource Description Framework (RDF) is a Web-based metadata framework for representing information about resources in the World Wide Web. It is a technology positioned to change the way concepts are synthesized into applications, both in localized data-driven systems and in the future Semantic Web. To improve productivity and quality in the management of RDF information, systems must be developed to facilitate its storage, manipulation, query, and inference.

We have developed SemanticWare, an Eclipse Modeling Framework (EMF) compatible middleware for storage, manipulation, query, and inference of RDF(S) information. A major goal of SemanticWare is to establish an end-end ontology engineering environment with tight integration with dominant Meta-Object Facility (MOF) based modeling and application development tools. As such, it provides a platform for managing RDF metadata and reduces the amount of programming required for the development of metadata-intensive applications.

Figure 1 shows the system architecture. There are three major tools in SemanticWare: Orient (a RDF GUI editor), EODM (EMF Ontology Definition Metamodel, an EMF implementation of ODM), and RStar (a high performance RDF repository). SemanticWare covers the whole ontology-based solution development lifecycle, from business modeling to application development. More exactly, the lifecycle contains the following steps:

Business modeling

(1) Domain Analysis: it's main task is to collect domain document and extract domain vocabulary and business concept;
(2) Build Conceptual Model: it focuses on designing business ontology based on existing domain ontology;

Application development

(3) Build Repository and Programming Model: it's main task is to provide solution developers a easy to use programming environment;
(4) Develop Application: it's major objective is to build ontology-based solution;
(5) Assetization, it focuses on accumulating work products, such as business ontology, for future reuse.

In business modeling perspective, business staffs can use Orient to build their ontology visually, serialize it into EODM, and then store it in RStar. Or they can use their favorite modeling tools to generate EMF compatible models, such as UML profiles or XML schemas, and transform these models into RDF through the XMI (XML Metadata Interchange) adaptor of EODM. In application development perspective, solution developers can manipulate the ontology and metadata stored in SemanticWare using EODM in the following three different ways: 1) RDF(S) programming model provided by RDFS Manipulator; 2) Service Data Object (SDO) DataGraph interface provided by SDO Adaptor; 3) RSQL,

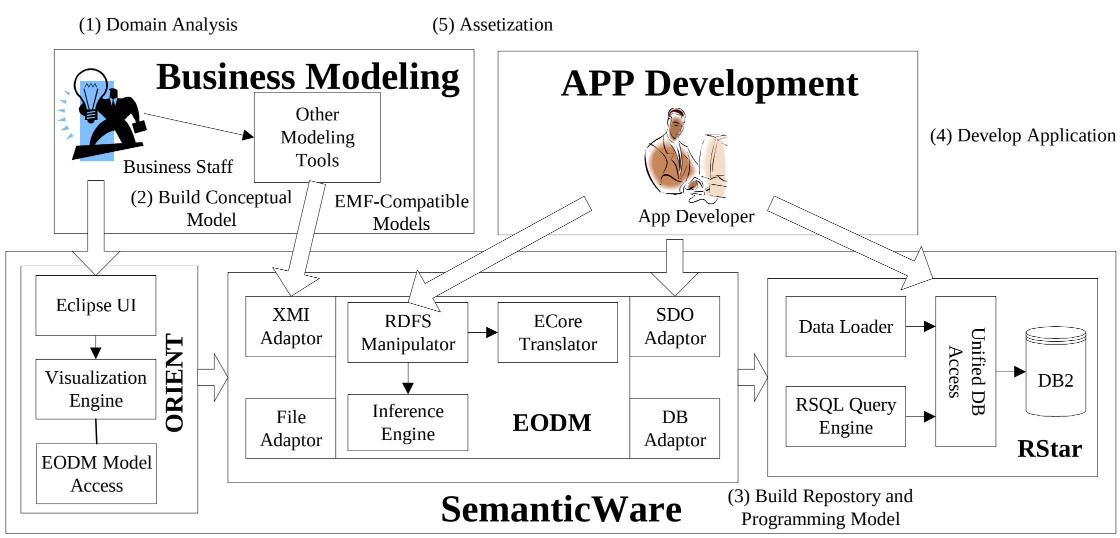

Figure 1 SemanticWare System Architecture

a SQL style query language provided by RStar.

As mentioned above, SemanticWare contains three main tools. Next, we introduce these three tools in detail.

The first tool, Orient, is responsible for the authoring of high quality ontologies, either from scratch or reusing other existing ontologies. It is designed as a set of loosely-coupled cooperative Eclipse plugins. Orient supports all the basic primitives of knowledge modeling: ontologies, classes, instances, functions, and relations. It contains three main components: Eclipse UI, visualization engine, and EODM model access. To better leverage the well-designed plugin architecture of Eclipse, Orient builds its UI elements as Eclipse plugins. This provides extensible user interface diagrams for various plugins to share and extend. Orient has its own perspective in Eclipse to group and manage a set of editors and views for authoring usage. New perspectives, views or editors can be integrated into the current Eclipse UI of Orient through its plugin mechanism smoothly, with the sustaining improvements of Orient. Visualization engine allows the user to visualize and navigate ontology freely. This engine treats ontology as a graph and applies graph visualization techniques to visualize ontology. The main feature of this engine is that it applies Level-Of-Detail (LOD) strategy to graph visualization and navigation, and allows the user to navigate the graph in real time. EODM Model Access layer provides Orient with an in-memory model and access API for the ontologies stored in RStar. All the user interface events can be translated into the operations at EODM level, which guarantees the separation between Model and View in Model-View-Controller (MVC) perspective.

The second tool, known as EODM, provides a programming model for SemanticWare, which is an extension and implementation of interfaces defined for RDF(S) in Ontology Definition Metamodel (ODM). With EODM, solution developers can access the functionalities of SemanticWare by using its programming interface.

EODM implements the standard meta-model for ontology modeling defined in ODM which includes 15 key interfaces and limited predefined methods. In order to put ODM interface to practice, EODM adds many methods to these interfaces and utility classes for practical usage. This provides all the necessary classes and operations needed for the representation, creation, modification, navigation and serialization of primitive modeling concepts in RDF(S). Based on the memory implementation of EODM, a serialization layer is provided with four serialization adaptors. The first one is a file adaptor, which makes use of an efficient RDF parser to load RDF files into EODM and vise verse. The second one is a DB adaptor, which translate the ontology represented in EODM into RDF triples and populate them into RStar, and vice versa. These two adaptors mainly focus on serialization capability to decouple the ontology user from actual ontology persistence storage mechanisms. The third one is an XMI adaptor, which is built on top of our embedded translator between RDF(S) memory representation and ECore model. The XMI adaptor defines the mapping relationship and methods for translating RDFS primitive concepts into basic building blocks of EMF meta-model. This kind of support makes the interchange and inter-operation between RDF and MOF compatible models possible, which will provide a comprehensive and open environment for SemanticWare to be well leveraged in the entire solution development lifecycle. The last one is an SDO adaptor, which provides an SDO DataGraph interface for developers to access.

There are three main components in EODM: RDFS manipulator, ECore translator, and inference engine. RDFS manipulator provides a set of classes for accessing and manipulating ontologies represented in RDFS. ECore translator supports transformation capability between ECore model and EODM model, which offers a loosely-coupled way to build linkages between these two kinds of models. The inference engine provides inference capabilities for EODM. It supports two modes of RDF(S) entailment. One is real-time inference mode and the other is non real-time inference mode. In the non real-time mode, inference will not be done until the user invokes the inference API; while in real-time mode, the inference will be done after each modification to the RDF(S) ontology.

The third tool, RStar, focuses on providing a high performance RDF storage and query system. It takes RDF/XML files or RDF triples as input and stores these triples in the backend database for efficient retrieval. A RDF query language called RSQL is used to retrieve these triples. One main feature of RStar is that it respectively stores ontology information and instance data in different tables for effective query, namely separating ontology from instance data. This is expected to make ontology navigation more effective as well as instance data retrieval faster. RStart includes two major components: Data loader and RSQL Query Engine. Data loader takes RDF triples generated by EODM as input and provides both the original and inferred triples to the backend database. The RSQL Query Engine allows the user to query the RDF repositories and acquire the desirable information. Its main objective is to make resource retrieval easier.