Modeling Views for the Semantic Web: As a Mechanism for Modeling Sub-Ontologies

Rajugan, R. [1], Elizabeth Chang[2], Tharam S. Dillon[1] and Ling Feng[3]
[1] *eXel Lab, Faculty of Information Technology, University of Technology, Sydney, Australia*
E-mail: {rajugan, tharam}@it.uts.edu.au
[2] *School of Information Systems, Curtin University of Technology, Australia*
E-mail: Elizabeth.Chang@cbs.curtin.edu.au
[3] *Faculty of Computer Science, University of Twente, The Netherlands*
E-mail: ling@ewi.utwente.nl

Abstract

The emergence of Semantic Web (SW) and related technologies promise to make the web a meaningful experience. Web services (WS) enable distributed access and discovery of internal, Enterprise functions and services over the web, in a secure controlled environment. Yet, high level modeling, design and querying techniques proves to be a challenging task for organizations that are hoping to utilize the SW paradigm for their industrial and WS applications. Conversely both, SW and WS paradigm can benefit, if some of the traditional database concepts, functionalities and models are made adoptable to the new SW/WS paradigm. To address one such issue, in this paper, we propose a view model for Semantic Web that can be utilized in the new web information system framework, the SW/WS. First we outline our view model and then briefly provide discussion on its properties and some modeling issues with the help of an industrial case study example.

1. Introduction

Web Service (WS) enable distributed access and discovery of internal enterprise applications and services over the web, in a secure and controlled environment [1]. Many traditional database concepts and techniques have been transformed and adopted to this new web application platform, which is based on core Object-Oriented (OO) principles. For example, works such as [2, 3] are good examples in this direction. The emergence of Semantic Web (SW) [4] and the related technologies promise to make the web a meaningful experience and it is another step towards the next generation of Enterprise Information Systems (EIS). However, success of such SW and its applications depends heavily on utilization and interoperability of well formulated ontology bases in an automated, heterogeneous environment. For example, utilization, integration and extraction of ontology bases in the context of EIS, where, enterprise vocabularies can be automatically extracted from various distributed sources and be used in one or more SW (or traditional) applications and e-services.

This creates the need investigate successful database technologies, such as views, in the context of SW, where (materialized) ontology views [5] can be used for; (a) ontology extraction, (b) ontology versioning and (c) sub-ontology generation, in an industrial settings. However, unlike traditional database systems, high level modeling, design and querying techniques still proves to be a challenging task for SW paradigm. This is mainly due to the nature of ontology bases and views, where. definitions and querying have to be done at high-level abstraction [5, 6]. Such a high-level view models can also be utilized in SW paradigm and also support and co-exist with existing WS architecture and/or traditional enterprise transactional systems.

The traditional databases systems (from relational to active and deductive systems) have matured enough to face growing challenges faced by the organizations (both commercial and governments) EIS. They have well defined concepts and principles [7] on which they are built upon. Due to this, supporting technologies such as transaction processing, data warehousing, data mining etc. have evolved to a level that can be considered as "matured". Many new and ongoing research directions in data intensive domains still follow these basic principles of databases [8], namely meta-data, schema and instance data. This, in our view is one of the major differences between the database and the SW principles, where meta-data schemas and instance data may overlap and the data extraction process of usually automated, in direct contrast to user queries in database systems).

On the other hand, Semantic Web directives are still at its infancy in areas such as data organization, meta-data models and query languages. As a result, in the current stage of SW developments, there are lots of contradictions than agreements in regards to basic concepts and definitions of the SW vocabularies. Regardless of these contradictions, many organizations, both academic and industry are working tirelessly in proposing new methodologies, models and are vigorously formulating standards to streamline the SW paradigm .

In spite of known drawbacks, there is an exponential growth in new research directions in SW applications. These applications range from SW enabled traditional enterprise meta-data repositories to time-critical medical information and infectious decease classification databases. For such vast ontology bases to be successful and to support autonomous computing in a distributed environment, the preliminary design and engineering of such ontology bases should follow a strict software engineering discipline [9]. Furthermore, supporting technologies for ontology engineering such as data extraction, integration and organization have be matured to provide adequate modeling and design mechanism to build, implement and maintain successful ontology and sub-ontology bases. For such purpose, Object-Oriented (OO) paradigm seems to be ideal choice as it has been proven in many other complex applications and domains [10, 11].

During mid relational and early Object-Oriented (OO) revolution, during similar phase of the technological development and standardization, all (both academia and industry) agreed that the data models should be independent of the underlying language semantics and syntaxes and be able to provide needed abstraction and model portability [10, 12]. Today, this notion still holds true for SW paradigm.

To address such an issue, in this paper, we propose a mechanism for modeling and designing views for SW paradigm (SW-view). In direct contrast to SW language specific view (e.g. RDF [13], OWL [14]), the proposed abstract view formalism is defined using a high-level modeling OO language that is capable of modeling ontology bases (for e.g. OMG's UML [15] with extensions for ontology engineering or Ontology Web Language (OWL)). Our main aim here is "re-use" and "share" of view definitions among multiple implementation paradigms and frameworks, thus, we provide view definitions at the highest level of abstraction (i.e. conceptual level) which enables us to transform and map one view definition to one or more technology specific platform, at the required level of abstraction (i.e. conceptual, schema or instance).

The rest of this paper is organized as follows. In section 2, we look at some of the work done in view models for SW, followed by the discussion on our view model in section 3. Section 4 presents an illustrative case study example to highlight some of our view model characteristics. Section 6 concludes the paper with some discussion on our future research directions.

2. Related Work

We can group the existing view models into four categories, namely; (a) classical (or relational) views, (b) OO view models, (c) semi-structured (namely XML) view models and (d) views for SW. A detailed, comprehensive discussion on these view models can be found in [16, 17]. Here we only look at views for SW.

In the SW [18] paradigm, some work has been done in views for SW [6, 19], where the authors proposed a view formalism for RDF document with support for RDF [13] schema (using a RDF schema supported query language called RDQL). This is one of the early works focused purely on RDF/SW paradigm and has sufficient support for logical modeling of RDF views. The extension of this work (and other related projects) can be found at [20]. RDF is an object-attribute-value triple, where it implies object has an attribute with a value [21]. It only makes intentional semantics and not data modeling semantics. Therefore, unlike views for XML, views for such RDF (both logical and concrete) have no tangible scope outside its domain. In related area of research, the authors of the work [5, 22] propose a logical view formalism for ontology with limited support for conceptual extensions, where materialized ontology views are derived from conceptual/abstract view extensions.

Another area that is currently under development is the logical view formalism for SW Meta languages such as OWL. In some SW communities, OWL is considered to be a conceptual modeling language for modeling ontologies, while some others consider it to be a crossover language with rich conceptual semantics and RDF like schema structures [22]. It is outside the scope of this paper to provide argument for or against OWL being a conceptual modeling language. Here, we only highlight one of view formalism that is under development for OWL, namely views for OWL in the "User Oriented Hybrid Ontology Development Environments" [23] project.

3. A View Model for Semantic Web

In this paper, we propose a view model for the SW (SW-view) paradigm. Initially, we proposed a layered view model in our work for semi-structured data (namely XML) [17], with clear distinction between three levels of abstraction namely; (a) conceptual, (2) logical (or schematic) and (c) document (or instance). But in the case of ontology domain, though there exists a clear distinction between conceptual and logical models/schemas, the line between the logical (or schema) level and document (or instance) level trends to overlap due to the nature of ontology bases, where concepts, relationships and values may present mixed sorts, such as schemas and values [24]. This unique nature of ontology bases together with the notion of shared conceptualization, is in direct contrast to the principles employed in the traditional and semi-structured view models such as in [17].

Therefore, in the SW-view model, we provide a clear distinction between conceptual and logical views, but depending on the application, we allow an overlap between logical and document views. This is one of the main differences between the XML views and the SW-views. To our knowledge, other than our work, there exist no research directions that explore the conceptual and logical view formalism for the Semantic Web (SW) paradigm. The SW-view model has explicit constraints and an extended set of expressive conceptual operators [24] that provide the mechanism for the design and development of sub-ontologies, such as in the MOVE [5, 22] system.

3.1. Conceptual Views

The *conceptual views* are views that are defined at the conceptual level with conceptual level semantics using a higher-level modeling languages such as UML [15]. To understand the SW-view and its application in constructing ontology views, it is imperative to understand its concept and its properties. First, an informal definition of the view concept is given followed by a formal definition that serves the purpose of highlighting the view model properties and the modeling issues associated with such a high-level construct.

Definition 1: A **conceptual view** is the one which is defined at the conceptual level with higher level of abstraction and semantics.

It should be noted here that, though there can be more elaborated definitions are possible depending on the application domain, here we provide a simplified generic conceptual view definition that can be easily applied.

***Definition 2*:** A **conceptual view** $\boldsymbol{V^c}$ is a 4-ary tuple $\boldsymbol{V^c} = (V^c_{name}, V^c_{obj}, V^c_{rel}, V^c_{constraint})$, where V^c_{name} is the name of the XML conceptual view V^c, V^c_{obj} is a set of objects in V^c, V^c_{rel} is a set of object relationships in V^c, and $V^c_{constraint}$ is a set of constraints associated with V^c_{obj} and V^c_{rel} in V^c.

Definition 3*:** Let $C = (C_{name}, C_{obj}, C_{rel}, C_{constraint})$ denote a context which consists of a context name C_{name}, a set of objects C_{obj}, a set of object relationships C_{rel}, and a set of constraints associated with its objects and relationships $C_{constraint}$. Let $\bar{\lambda}$ be a set of conceptual operators. $\boldsymbol{V^c} = (V^c_{name}, V^c_{obj}, V^c_{rel}, V^c_{constraint})$ is called a *valid conceptual view of the context* ***C, if and only if the following conditions satisfy;

- For any object $\forall o \in V^c_{obj}$, there exist objects $\exists o_1, \ldots, o_n \in C_{obj}$, such that $o = \lambda_{1 \ldots}\lambda_m (o_1, \ldots, o_n)$ where $\lambda_{1 \ldots}\lambda_m \in \bar{\lambda}$. That is, o is a newly derived object from existing objects $o_1, \ldots, o_n$ in the context via a series of conceptual operators [24] $\lambda_{1, \ldots}\lambda_m$ like select, join, etc.
- For any constraint $\forall c \in V^c_{constraint}$, there exists a constraint $\exists c' \in C_{constraint}$ or a new constraint c'' constraints associated with V^c_{obj} or V_{rel}.
- For any hierarchical relationship $\forall r^h \in V^c_{rel}$, there *does not exist* a relationship between one or more and V^c_{obj} and C_{obj}.
- For any association relationship/dependency relationships $\forall r^a \in V^c_{rel}$, there *may exist a relationship between one or more V^c_{obj} and C_{obj}.*

The term ***context*** refers to the domain that interests an organization as a whole. It is more than a measure and implies a meaningful collection of objects (or concepts), relationships among these objects, constraints that are associated with the objects and their relationships, which are relevant in a give context.

A Context is presented in UML using modeling primitives like object, attribute, relationship and constraint in our work. To enable the construction of a valid conceptual view from a context, we introduce the notion of *conceptual operator*. These conceptual level operators are comparable to relational operators in the relational model, but they operate on conceptual level objects and relationships.

Conceptual operators are grouped into set operators, namely union, difference, intersection, Cartesian product and unary operators namely projection, rename, restructure, selection and joins, and can facilitate systematic construction of conceptual views from context. These conceptual operators can be easily transformed into query segments, user-defined functions and/or procedures for implementation. By doing so, they help the modeler to capture view construct at the abstract level without knowing or worrying about query/language syntax. The set of binary and unary operators provided here is a complete or basic set; i.e. other operators, such as division operator and compression operator [24] can be derived from these basic set of operators.

3.2. Modeling Conceptual Views

In this paper, to model conceptual views, we use OMG's UML/OCL [15, 25] . The reason we use this notations here is only to demonstrate our concepts and applications and not to emphasis or promote UML as the only modeling notation for conceptual views. Other modeling notations used to model conceptual views can be found in [21, 26].

UML has established itself as the *defacto* modelling language of choice in OO conceptual modelling paradigm and well-understood by both academia and the industry. It supports multiple implementation frameworks, for both WS and SW. UML provide a well defined collection of tools to visually model a given domain into needed level of abstraction. It can be said that, UML helps to provide a well-defined blue print for a software system that is easily understood by both users and developers alike. UML also provides extensibility to the modelling language in the form of *stereotypes* which we utilise in defining our *conceptual views*. In the case of ontology engineering, UML provide classes (similar to concepts in ontology), attributes and relationships that are used in defining ontology models [5, 27].

Another reason we adopt UML is that, its models are portable, i.e. many schemata transformation rules and mapping techniques exists for transforming UML models to; (a) XML Schema [21, 26], (2) Ontology Web Language (OWL) [28, 29], (c) RDF and (d) XMI [30]. Therefore, for the purpose of this paper, UML is visual modelling language of choice for OOCM and support abstraction from classical data models to ontology bases. An illustrative case study example model (*see* section 4) is given in Fig. 2.

Here, for our view formalism, we look into using UML/OCL as our view constraint specification language. Since our conceptual view mechanism is defined using a high-level OO modeling language, we can provide explicit view constraint specification model, as most high-level OOCM languages provide some form constraint specification. In UML, the Object Constraint Language (OCL)[25], which is now a part of the UML 2.0 standard [15], can support unambiguous constraints specifications for UML models including specification of ontology model elements . In our conceptual view model, we incorporate OCL (in addition to built-in UML constraint features) as our view constraint specification language to explicitly state view constraints. It should be noted that, we do not use OCL to define views, rather state additional constraints using OCL. OCL also supports defining *derived* classes [25, 31], which we do not use. Some examples of constraints for conceptual views are given in section 4.

3.3. An Application of Conceptual Views

Here, we briefly discuss how conceptual views can be applied in extracting sub-ontologies in the Materialized Ontology View Extractor (MOVE) [22]. The MOVE system was initially proposed by Wouters et al. [5, 22], for the construction *optimized* materialised ontology views, with emphasis on automation and quality of the views generated.

Definition 4: [24](Informal) A Strict Semantic Web View (or ontology view) is a materialized SW-view that is derived from an ontology (called the base ontology). The derivation can consist of any (combination) of the following operations; synonymous rename, selection and compression.

The MOVE view process includes model and design of conceptual views with the utilization of restricted conceptual operators in deriving materialized ontology views. Some of the restricted view operators include [5, 24]; (a) synonymous rename (2) selection and (3) compression.

4. An Illustrative Case Study

To help illustrate our concepts, we conduct a real-world case study in a fictitious global logistic company called LWC & e-Solutions Inc., e-Sol in short in the following. The e-Sol Inc. aims to provide logistics, warehouse, and cold storage space for its global customers and collaborative partners. The e-Sol solution includes (Fig. 1) a standalone and distributed Warehouse Management System (WMS/e-WMS), and a Logistics Management System (LMS/e-LMS) on an integrated e-Business framework called e-Hub [32] for all inter-connected services for customers, business customers, collaborative partner companies (Fig. 2), and LWC staff (for e-commerce B2B and B2C). Some real-world applications of such company, its operations and IT infrastructure can be found in [32, 33]. Here, we use this system as the base to model and integrate ontology bases (using ontology views) and various sub-ontology vocabularies used at various customer and collaborative partner locations.

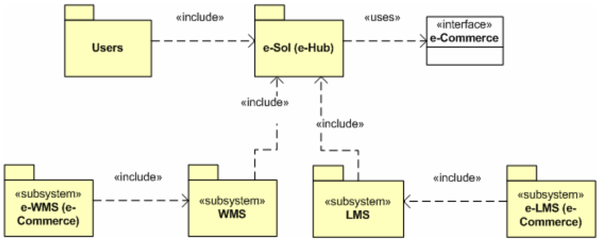

Figure 1: e-Sol context diagram

In e-Sol, due to the business process, data semantics have to be in different formats (ontology bases and vocabularies) to support multiple systems, customers, warehouses and logistics providers. Also, data have to be duplicated at various points in time, in multiple databases, to support collaborative business needs. In addition, since new customers/providers to join the system (or leave), the data formats has to be dynamic and should be efficiently duplicated without loss of semantics. This presents an opportunity to investigate how to integrate and utilize various customers' and collaborative partners' ontology bases for mutual benefit and SW applications. The following examples highlight some of the conceptual views developed for the e-Sol. ***Note:*** It should be note that, the examples and the figures given for the e-Sol are demonstration purpose only and do not provide the complete ontology base model of the system.

In Fig. 2, the e-Sol users are shown (both views and concrete (or context) objects). Stereotyped classes ("`view`") and relationships (together with OCL statements) are used to show the conceptual views and the associated relationships.

In Fig. 2-3, `Warehouse-Manager` is a valid conceptual view, named in the context of `Staff`. It is constructed using the conceptual SELECT conceptual operator, which can be shown as;

$$\sigma_{\texttt{warehouse-Staff.Role="manager"}}.$$

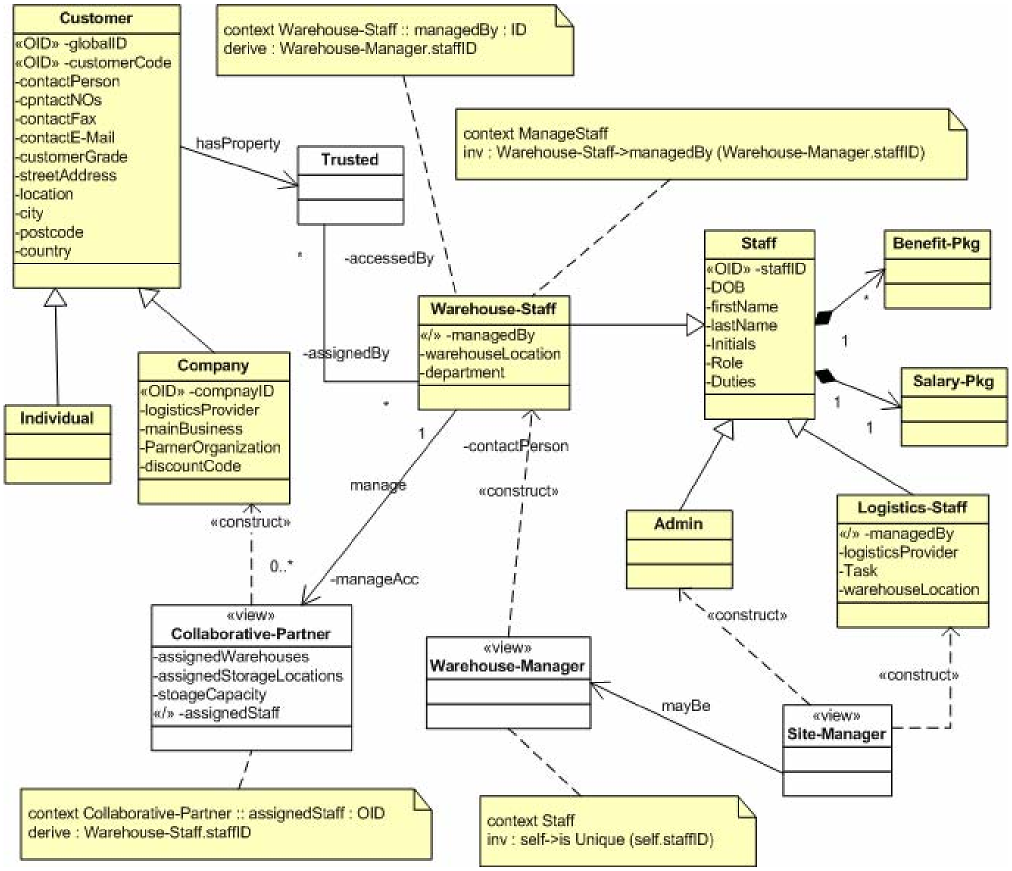

Figure 2: e-Sol user model (simplified)

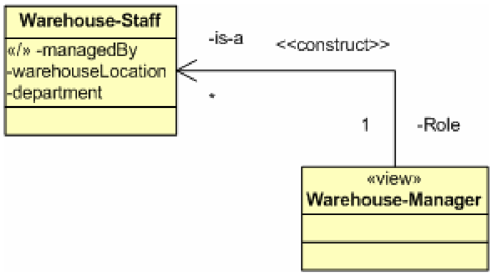

Figure 3: A Conceptual view and its dependencies

In real-world, composite objects being in an aggregation with one or more sub-objects, they also can be in a pre-defined order. This signifies an important OO concept, *ordered composition*. To capture thid notion, we add an UML stereotype that allows capturing of the ordered composition utilizing stereotypes to specify the objects' order of occurrence such as `<<1>>, <<2>>, <<3>>, .… ,<<n>>`, as shown in Fig. 4.

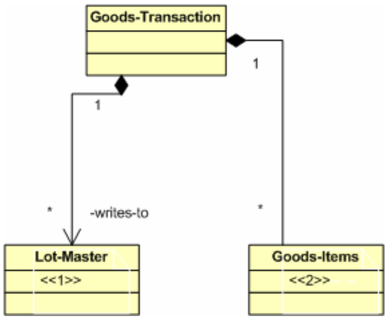

Figure 4: *Ordered* composition (UML/OCL)

In the case of conceptual views "`Lot-Movement`", the exclusive disjunction between `Internal-Lot-Movement` (stored goods change owners) and External-Lot-Movement (goods shipped outside the warehouse) can be show via the OCL statement "OR" between the relationships, as shown in Fig. 5.

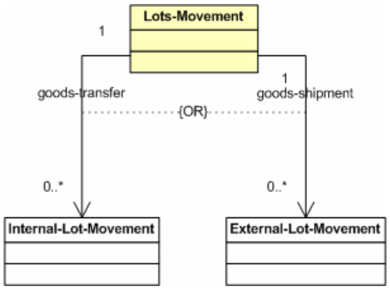

Figure 5: Exclusive disjunction (UML/OCL)

The *cardinality constraint* shows the number of instances of one class (or concept) that may relate to single instance of another, as shown in Fig. 6.

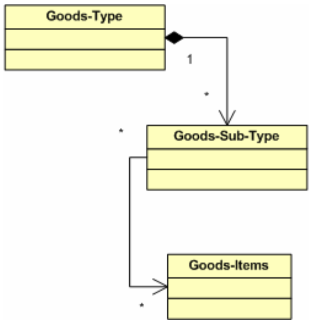

Figure 6: Cardinality constraints (UML/OCL)

In the case of conceptual view "`Income`" (Fig. 7), the following OCL statements hold true;

```
context Income :: Staff : ID
derive : Staff.staffID

context Income :: benefits : Real
derive : Benefit-Pkg.totalBenefits

context Income :: baseSalary : Real
derive : Salary-Pkg.baseSalary

context Income :: totalSalary : Real
derive :
       totalSalary =   (self.baseSalary -
               self.tax) + benefits -
               self.totalDeductions
```

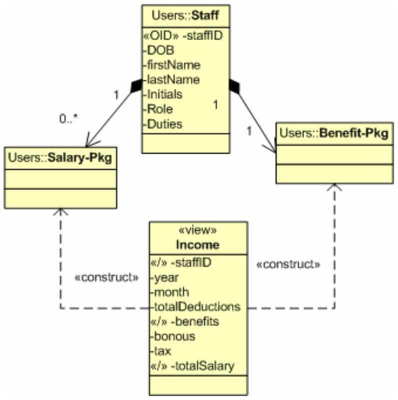

Figure 7: Dependency / Adhesion constraint (UML/OCL)

5. Conclusion and Future Work

Views have proven to be very useful in databases and here, we proved a brief, yet descriptive discussion and an abstract view model (SW-view) for SW. First, we described the opportunities and challenges for utilizing SW and WS technologies for EIS. Then we briefly provided some arguments for an abstract view model and discussed its properties, definitions and modelling aspects. Finally, we presented a practical walkthrough of the view model using an industrial case study example.

For future work, some further issues deserve investigation. First, the investigation of a formal mapping approach to conceptual view constraints, to automate the view constraint model transformation between the SW-view model to SW languages such as RDF and OWL schema constraints. Second, the automation of the mapping process between conceptual operators to various SW language specific constructs.

6. References

[1] G. Alonso, *Web services : concepts, architectures and applications*. Berlin ; London: Springer, 2002.
[2] S. Abiteboul, O. Benjelloun, I. Manolescu, T. Milo, and R. Weber, "Active XML: A Data-Centric Perspective on Web Services," BDA, 2002.
[3] S. Abiteboul, O. Benjelloun, I. Manolescu, T. Milo, and R. Weber, "Active XML: Peer-to-Peer Data and Web Services Integration," Proceedings of the 28th Int. Conf. on VLDB, HK, China, 2002.
[4] W3C-SW, "(http://www.w3.org/2001/sw/)," W3C, 2005.
[5] C. Wouters, T. S. Dillon, J. W. Rahayu, E. Chang, and R. Meersman, "A Practical Approach to the Derivation of a Materialized Ontology View," in *Web Information Systems,*. USA: Idea Group Publishing, 2004.
[6] R. Volz, D. Oberle, and R. Studer, "Views for light-weight Web ontologies," Proceedings of the ACM Symposium on Applied Computing (SAC '03), USA, 2003.
[7] R. Elmasri and S. Navathe, *Fundamentals of database systems*, 4th ed. New York: Pearson/Addison Wesley, 2004.
[8] W. Kim and W. Kelly, "Chapter 6: On View Support in Object-Oriented Database Systems," in *Modern Database Systems*: Addison-Wesley Publishing Company, 1995, pp. 108-129.
[9] P. Spyns, R. Meersman, and J. Mustafa, "Data Modeling Versus Ontology Engineering," SIGMOD, 2002.
[10] T. S. Dillon and P. L. Tan, *Object-Oriented Conceptual Modeling*: Prentice Hall, Australia, 1993.
[11] I. Graham, A. C. Wills, and A. J. O'Callaghan, *Object-oriented methods : principles & practice*, 3rd ed. Harlow: Addison-Wesley, 2001.
[12] OMG-MDA, "The Architecture of Choice for a Changing World®, MDA Guide Version 1.0.1 (http://www.omg.org/mda/)," OMG, 2003.
[13] W3C-RDF, "Resource Description Framework (RDF), (http://www.w3.org/RDF/)," 3 ed: The World Wide Web Consortium (W3C), 2004.
[14] W3C-OWL, "OWL: Web Ontology Language 1.0 reference (http://www.w3.org/2004/OWL/)," W3C, 2004.
[15] OMG-UML™, "UML 2.0 Final Adopted Specification (http://www.uml.org/#UML2.0)," 2003.
[16] R.Rajugan, E. Chang, T. S. Dillon, L. Feng, and C. Wouters, "Modeling Ontology Views: An Abstract View Model for Semantic Web," 1st Int. IFIP/WG 12.5 Working Conf. on Industrial Applications of Semantic Web (IASW '05), Jyvaskyla, Finland, 2005.
[17] R.Rajugan, E. Chang, T. S. Dillon, and L. Feng, "A Three-Layered XML View Model: A Practical Approach," 24th Int. Conf. on Conceptual Modeling (ER '05), Klagenfurt, Austria, 2005.
[18] W3C-SW, "Semantic Web, (http://www.w3.org/2001/sw/)," W3C, 2005.
[19] R. Volz, D. Oberle, and R. Studer, "Implementing Views for Light-Weight Web Ontologies," Seventh Int. Database Engineering and Applications Symposium (IDEAS'03), Hong Kong, SAR, 2003.
[20] KAON, "KAON Project (http://kaon.semanticweb.org/Members/rvo/Folder.2002-08-22.1409/Module.2002-08-22.1426/view)," 2004.
[21] L. Feng, E. Chang, and T. S. Dillon, "Schemata Transformation of Object-Oriented Conceptual Models to XML," *Int. Journal of Computer Systems Science & Engineering*, vol. 18, No. 1, pp. 45-60, 2003.
[22] C. Wouters, T. S. Dillon, J. W. Rahayu, E. Chang, and R. Meersman, "Ontologies on the MOVE," 9th Int. Conf. on Database Systems for Advanced Applications (DASFAA '04), Jeju Island, Korea, 2004.
[23] HyOntUse, "User Oriented Hybrid Ontology Development Environments, (http://www.cs.man.ac.uk/mig/projects/current/hyontuse/)," 2003.
[24] C. Wouters, R.Rajugan, T. S. Dillon, and J. W. Rahayu, "Ontology Extraction Using Views for Semantic Web," in *Web Semantics and Ontology*,. USA: Idea Group Publishing, 2005.
[25] OMG-OCL, "UML 2.0 OCL Final Adopted specification (http://www.omg.org/cgi-bin/doc?ptc/2003-10-14)," OMG, 2003.
[26] L. Feng, E. Chang, and T. S. Dillon, "A Semantic Network-based Design Methodology for XML Documents," *ACM Transactions on Information Systems (TOIS)*, vol. 20, No 4, pp. 390 - 421, 2002.
[27] D. Gašević, D. Djuric, V. Devedzic, and V. Damjanovic, "Approaching OWL and MDA Through Technological Spaces," 3rd Workshop in Software Model Engineering (WiSME 2004), Lisbon, Portugal, 2004.
[28] D. Gaševic, D. Djuric, V. Devedzic, and V. Damjanovic, "Converting UML to OWL Ontologies," Proceedings of the 13 th Int. World Wide Web Conf., NY, USA, 2004.
[29] D. Gaševic, D. Djuric, V. Devedžic, and V. Damjanovic, "UML for Read-To-Use OWL Ontologies," Proceedings of the IEEE Int. Conf. Intelligent Systems, Vrana, Bulgaria, 2004.
[30] OMG-XMI, "XML Metadata Interchange (XMI) Version 2.0, (http://www.omg.org/technology/documents/formal/xmi.htm)," OMG, 2003.
[31] J. B. Warmer and A. G. Kleppe, *The object constraint language : getting your models ready for MDA*, 2nd ed. Boston, MA: Addison-Wesley, 2003.
[32] E. Chang, T. S. Dillon, W. Gardner, A. Talevski, Rajugan R., and T. Kapnoullas, "A Virtual Logistics Network and an e-Hub as a Competitive Approach for Small to Medium Size Companies," 2nd Int. Human.Society@Internet Conf., Seoul, Korea, 2003.
[33] E. Chang, W. Gardner, A. Talevski, E. Gautama, Rajugan R., T. Kapnoullas, and S. Satter, "Virtual Collaborative Logistics and B2B e-Commerce," e-Business Conf., Duxon Wellington, NZ, 2001.