International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:7, No:1, 2013

Database Modelling Using WSML in the Specification of a Banking Application

Omid Sharifi, *Member, ACM,* and Zeki Bayram, *Member, ACM*

Abstract—We demonstrate through a sample application, E-banking, that the Web Service Modelling Language Ontology component can be used as a very powerful object-oriented database design language with logic capabilities. Its conceptual syntax allows the definition of class hierarchies, and logic syntax allows the definition of constraints in the database. Relations, which are available for modelling relations of three or more concepts, can be connected to logical expressions, allowing the implicit specification of database content. Using a reasoning tool, logic queries can also be made against the database in simulation mode.

Keywords—Semantic web, ontology, E-banking, database, WSML, WSMO, E-R diagram.

I. Introduction

AN ontology is an explicit formal shared conceptualization of a domain of discourse [13]. Basically, an ontology acts like a dictionary, defining the common terminology in some domain. Ontologies form a very significant foundation of the semantic web on which other components are built [4]. One of the major components of an ontology is the "concept". Concepts are used to establish the basic elements of the agreed terminology for a problem domain. From a high-level perspective, a concept is described by a concept definition and provides attributes with names and types [15]. A "concept" corresponds pretty much to the "class" construct in object-oriented programming languages.

An Entity-Relationship (E-R) [3] diagram is used to graphically depict the structure of data in a database, as well as the relationships among the data. It allows the specification of class (entity) hierarchies, relationships among entities and other relationships (called aggregations), multiplicities of relationships (one-one, one-many, many-many), as well as other constraints such as candidate keys, partial keys and total participation. It is used in the initial phase of the database design, and needs to be mapped onto an actual table design later on. An E-R diagram can be viewed as the main deliverable of a conceptual data model. Despite the fact that the newer approaches to E-R modelling have developed (such as UML, a general-purpose visual modeling language used to specify, visualize, analyze, and document the artifacts of a software system [19] [16]), the E-R approach is still cited by some professionals as the premier model for conceptual database design [5].

O. Sharifi is with the Department of Computer Engineering, Eastern Mediterranean University, Famagusta, Cyprus. e-mail: omid.sharifi@emu.edu.tr

Z. Bayram is with the Department of Computer Engineering, Eastern Mediterranean University, Famagusta, Cyprus. e-mail: zeki.bayram@emu.edu.tr.

Web services are computational units that "sit" on the World Wide Web (WWW), and can be called through standard interfaces and protocols, such as HTTP [2] and SOAP [1]. They represent a paradigm shift in Computer Science, where abstraction from hardware to software has been replaced by abstraction from software to service-ware in terms of Service Oriented Computing [10]. *Semantic* Web Services (SWS) are web services with machine interpretable semantic descriptions [11], which give a formal specification of their functionality and behaviour. This formal specification allows their automatic discovery and invocation through appropriate semantic web service frameworks. Semantic web services make extensive use of ontologies.

Web Service Modelling Ontology (WSMO) [6] is a framework for semantic description of Semantic Web Services based on the Web Service Modeling Framework (WSMF) [9]. Its four main components are ontologies, web services, goals and mediators. Web Service Modelling Language (WSML) [12] is a language for modeling web services, ontologies, and related aspects of WSMO framework, to provide the description of semantic web services so that automatic discovery and invocation becomes possible. Five language variants of WSML exist based on Description Logic and Logic Programming. Each language variant provides different levels of logical expressiveness [12]. The variants are: WSML-Core, WSML-DL, WSML-Flight, WSML-Rule and WSML-Full.

WSML has an ontology component that acts like an intelligent, object-oriented database system that the other components of the framework utilize for "common understanding" of the data and terminology involved in the web service discovery and invocation process [20]. WSMO-based discovery engines make extensive use of ontologies as well [14] [8] [17] [7] [18].

In this paper, we focus on the ontology component of WSML, and demonstrate its capabilities as a database design language. We take an E-R diagram of a banking application, and convert it to a WSML ontology. Due to its inherent logical capabilities, WSMO ontology component is in fact much more expressive than E-R diagrams in specifying constraints on the database. We demonstrate this through an axiom which guarantees that marriages of only different genders are allowed.

The reminder of the paper is organized as follows. Section II depicts the E-banking ontology in WSML. This ontology contains concepts, instances, relations and axioms of the E-banking domain. Section III is the conclusion and future work.

II. E-banking Ontology

We used WSML-Rule as the design language for the E-banking database application. Logic-wise, WSML-Rule is sim-

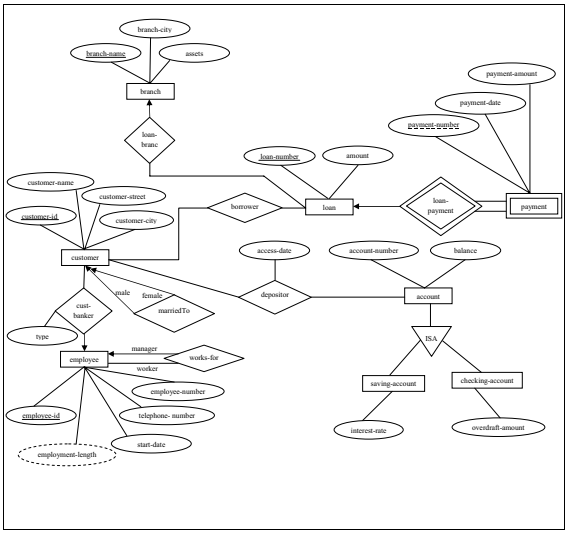

Fig. 1. E-R diagram for a banking enterprise

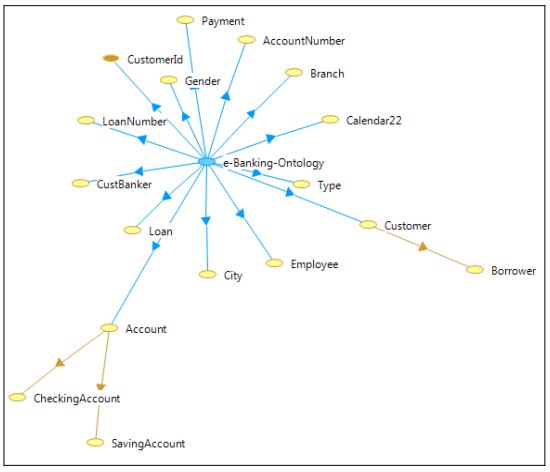

Fig. 2. WSML visualizer depicting the concepts of the E-banking ontology

ilar to Prolog, and is a good compromise between expressivity/complexity and practicality. We used the E-R diagram given in figure 1 as the starting place of our specification of the E-banking application. The E-R diagram itself is a modification of the banking E-R diagram from [3].

A. E-banking Ontology concepts

Figure 2 depicts a graphical view of some portion of the E-banking concepts that are used in defining the ontology. We shall elaborate on some of the essential concepts the following subsections.

1) "Branch" concept: In the E-R diagram in figure 1, the "branch" entity set has three attributes namely: "branch-name," "branch-city" and "assets." Figure 3 depicts the corresponding "branch" concept of E-banking ontology with the same attributes as in the E-R diagram.

```
concept Branch
    branchCity ofType City
    branchName ofType _string
    assets ofType _decimal
```

Fig. 3. Defining the Branch concept in WSML

```
concept Customer
    customerId ofType CustomerId
    customerName ofType (1 1) _string
    customerStreet ofType _string
    customerCity ofType City
    gender ofType Gender
    marriedTo symmetric ofType (0 1) Customer
    hasLoan ofType (0 *) Loan
```

Fig. 4. Defining the Customer concept in WSML

2) "Customer" concept: The "Customer" concept includes attributes "customerId," "customerName," "customerStreet" and "customerCity," which correspond to the attributes of the "customer" attributes in the E-R diagram. In addition, we deliberately added two attributes "gender" and "marriedTo" to represent the one-to-one relationship in between two customer objects, in case such a relationship between customers exists. Figure 4 illustrates customer concept. Note that the "marriedTo" attribute was defined as *symmetric*, since a person being married to another person implies that the second person is married to the first. The symmetricity constraint cannot be expressed in E-R diagrams. Further restrictions on marriages, also not possible in E-R diagrams directly, will be given later as an axiom.

3) "Employee" concept: The "Employee" concept, given in figure 5, defines the attributes of an employee object. These attributes are "employeeId," "employeeName," "telephoneNumber," "startDate" and "employmentLenght." In addition, we explicitly defined the transitive "worksFor" attribute of the "Employee" concept to stand for the "works-for" relation in the E-R diagram. Additional descriptive features are the base attribute "startDate," and the derived attribute "employmentLengh." "EmploymentLenght" has no explicit value, since its value is dependent on the current date and start date of the employee, and needs to be computed. The "computeEmploymentLength" axiom, given in figure 14, does this computation in a logical way.

4) "Account" concepts: In the E-R diagram two account entity sets are present: "saving-account" and "checking-

```
concept Employee
    employeeId ofType (1 1) _integer
    employeeName ofType _string
    telephoneNumber ofType _string
    startDate ofType Calendar
    employmentLenght ofType (1 1) _integer
    worksFor transitive impliesType Employee
```

Fig. 5. Defining the Employee concept in WSML

```
concept Account
   accountNumber ofType AccountNumber
   balance ofType (1 1) _decimal

concept SavingAccount subConceptOf Account
   interestRate ofType (1 1) _decimal

concept CheckingAccount subConceptOf Account
   overdraftAmount ofType (1 1) _decimal
```

Fig. 6. Defining the Account concepts in WSML

```
concept Loan
   loanNumber ofType (1 1) LoanNumber
   amount ofType (1 1) _decimal
   inBranch ofType Branch
   borrower ofType (1 *) Customer
   payment ofType (0 *) Payment
```

Fig. 7. Defining the Loan concept in WSML

account" with common attributes "account-number" and "balance." Figure 6 depicts the definition of these concepts in WSMO. The concepts "SavingAcount" and "CheckingAccount" inherit from tha base concept "Account," both in the E-R diagram and E-banking ontology. The "SavingAccount" concept has the attribute "interestRate" and the "CheckingAccount" concept has the attribute "overdraftAmount."

5) "Loan" concept: The "Loan" concept, standing for the "Loan" entity set in the E-R diagram, is deceptively simple. It seems to have only two attributes, "loanNumber" and "amount." However, in the E-R diagram, the "Loan" entity set is involved in a

- many-to-many relationship with the "Customer" entity set through the "borrower" relation,
- many-to-one relationship with the "Branch" entity set through the loan-branch relationship,
- and a special many-to-one total participation relationship to the "Payment" entity set through the "loan-payment" relationship.

Furthermore, the "Payment" entity set in the E-R diagram is *weak* in the sense that it has only a partial key, depending on the "Loan" entity set to form its primary key.

There is no need for any explicit relations in the ontology to model the relationships that "Loan" is involved in, since we can use set valued attributes as necessary. Figure 7 shows loan concept that is a class of E-banking ontology. However, for the total participation "loan-payment" relationship, we need an axiom which enforces the constraint that a payment object cannot exist unless it is related to a loan object. This constraint is given as an axiom in figure 12.

6) "Payment" concept: The last entity in the E-R diagram is "Payment," which, as we mentioned, is a weak entity set, meaning that it has no primary key of its own, but depends on some other entity set (in this case the "Loan" entity set to form its key). Its corresponding concept in the ontology has attributes "paymentDate" and "paymentAmount," and "paymentNumber" which come from the entity set "Payment," as well the attribute "forLoan" which links payment objects to

```
concept Payment
   paymentNumber ofType _integer
   paymentDate ofType Calendar
   paymentAmount ofType (1 1) _decimal
   forLoan ofType Loan
```

Fig. 8. Defining the Payment concept in WSML

```
concept Calendar
   year ofType (1 1) _integer
   month ofType (1 1) _integer
   day ofType (1 1) _integer
```

Fig. 9. Defining the Calendar concept WSML

loans. The "Payment" concept is depicted in figure 8.

7) "Calendar" concept: The "Calendar" concept, depicted in figure 9, is a utility concept, used as attribute types in other concepts, such as "Employee" and "Payment."

B. E-banking Instances

Instances in an ontology correspond to actual records (data) in a database. Although instances usually play no role at the design stage of database, or corresponding ontology, in the presence of constraints specified as axioms, it becomes necessary to populate the ontology to test the validity of the axioms and detect any mistakes in their definition. This is also true in databases, where we may have complex constraints on the database which need to be tested. So, for illustrative purposes, we have in figure 10 instances of the "Customer" concept.

C. E-banking relations and axioms

A relation can be be defined between concepts in ontology. Membership in a given relation can be specified logically, through the definition if axioms, which basically are logical expressions with a name. In WSML, axioms preceded by "!-" are *constraints*, meaning that the logical expression that follows "!-" must never be true, otherwise an error condition is reported.

Usually, relationships in an E-R diagram can be mapped to attributes in concepts. When a relationship has attributes,

```
instance cust019283746 memberOf Customer
   customerId hasValue id019283746
   customerName hasValue "smith"
   customerStreet hasValue "north"
   customerCity hasValue rye
   gender hasValue male
   marriedTo hasValue cust182736091

instance cust182736091 memberOf Customer
   customerId hasValue id182736091
   customerName hasValue "turner"
   customerStreet hasValue "sarah"
   customerCity hasValue stamford
   gender hasValue female
   marriedTo hasValue cust019283746
```

Fig. 10. Instances of Customer concepts in the E-banking ontology

```
axiom custBankerMapperAxiom
   definedBy
      ?a[
         customer hasValue ?cus,
         employee hasValue ?emp, type hasValue ?typ
       ] memberOf CustBanker
      equivalent custBankerRel(?cus,?emp,?typ).

relation custBankerRel(ofType Customer,
                       ofType Employee,
                       ofType CustBankerType
                      )

concept CustBanker
     customer ofType Customer
     employee ofType Employee
     type ofType CustBankerType
```

Fig. 11. Defining the CustBanker concept and custBankerRel relation in WSML

```
relation loan_payment( ofType Payment,
                       ofType Loan)

axiom paymentForLoan
   definedBy
      loan_payment(?p,?loan):-
      ?p[forLoan hasValue ?loan] memberOf Payment.

axiom noPaymentForLoan
   definedBy
      no_loan_for_payment(?p):-
      naf loan_payment(?p,?loan).

axiom totalParticipationCheck
   definedBy
      !- ?p memberOf Payment
       and no_loan_for_payment(?p).
```

Fig. 12. Defining the payment-related axioms in WSML

however, and is a many-to-many relationship, then we need a relation on the ontology side as well. This is the case with the "depositor" relationship, which has an "accessDate" attribute. The "cust-banker" relationship, with attribute "type," is many-to-one, but it is not total participation, so there may be some customers who have no banker. Again in this case it is wise to have a relationship on the ontology side.

Also, to provide flexibility in dealing with relations with attributes, it is good practice to be able to access individual members of the relation as objects. This can be done by providing a concept for the relationship, and a mapping between the concept and the relation.

1) E-banking axiom and relation "custBankerMapperAxiom": In figure 11 we have the definition of the relation "custBankerRel," the its corresponding concept "CustBanker," as well an axiom that maps instances of the "CustBanker" concept to members of the "'custBankerRel."

2) E-banking axiom and relation "loan-payment": "Loan-payment" is a one-to many relationship from loan to payment to determine which payments are made on a loan. Figure 12 denotes related axioms that enforces every payment to be related with a specific loan (no payment can exist without a related loan).

```
axiom helpers
   definedBy
      opposite(male,female).
      opposite(female,male).

axiom marriageOk
   definedBy
      happilyMarried(?p):-
      ?p[ gender hasValue ?g1,
          marriedTo hasValue ?hw
        ] memberOf Customer and
      ?hw[
          gender hasValue ?g2
         ] memberOf Customer and opposite(?g1,?g2).

axiom marriageCheck
   definedBy
      !- ?p[
             marriedTo hasValue ?SB
           ] memberOf Customer
           and naf(happilyMarried(?p)).
```

Fig. 13. Axioms constraining marriages to be between different genders

```
axiom difference
  definedBy
    diff(?startDate,?EmploymentLenght):-
       system#currentDate(?curr) and
       ?curr[year hasValue ?currYear]
             memberOf Calendar and
       ?startDate[year hasValue ?startYear] and
       wsml#numericSubtract(?EmploymentLenght,
                            ?currYear,?startYear).


axiom omid#computeEmploymentLenght
  definedBy
     ?empl[omid#startDate hasValue ?sd,
           omid#employmentLenght hasValue ?el]
            memberOf omid#Employee
     equivalent
              omid#diff(?sd,?el).
```

Fig. 14. Axioms for computing the value of the derived attribute employmentLength

3) E-banking axioms relating to marriage status of customers: Figure 13 depicts axioms that enforce the traditional custom that marriages take place between opposite genders.

4) E-banking axiom "computeEmploymentLenght": "employmentLenght" is a derived attribute which denotes how long an employer is working in the bank. Figure 14 depicts the related axioms to find out this integer value. There is an assumption in this part that a "currentDate" predicate supplied by the system.

5) E-banking axiom "date validity": To complete axioms in E-banking ontology the following axioms related to date are defined. Figure 15 depicts the "validDay," "validMonth" and "validYear" axioms to test date validity of "Calendar" instances.

6) E-banking relation "depositor": Figure 16 illustrates the "depositor" relation, which denotes a many-to-many relationship set between "Customer" and "Account," with "Calendar" being the attribute of the relationship. Note that we do not need an explicit "borrower" relationship since the relationship is

```
axiom validDay
  definedBy
    !- ?c[day hasValue ?v] memberOf Calendar
    and ?v > 31.

axiom validMonth
  definedBy
    !- ?c[month hasValue ?v] memberOf Calendar
    and ?v > 12.

axiom validYear
  definedBy
    !- ?c[year hasValue ?v] memberOf Calendar
    and ?v < 2012.
```

Fig. 15. Axioms related to validity of dates (Calendar concept)

```
relation depositor( ofType Customer,
                    ofType Account,
                    ofType Calendar )
```

Fig. 16. Defining the "depositor" relation in WSML

encoded implicitly in the attributes of "Loan" and "Customer."

III. Conclusion and Future Work

We have demonstrated through a reasonably comprehensive example that WSML-rule ontology component is perfectly suited as a database design language. We have taken an E-R diagram for a banking application which had a near-complete set of E-R features, including inheritance, total participation, weak entity sets, relations with attributes, and shown how the same informational content can be coded as a WSML-rule ontology. We have also shown some features of WSML-rule that are not easily duplicated in E-R diagrams. Our conclusion is that WSML-rule ontology component can be used as a powerful database design language.

Future work in this area would include comparing WSML-rule with the capabilities of more expressive software design languages, such as UML.

References

[1] D14v1.0. ontology-based choreography, WSMO final draft 15 february 2007. http://www.w3.org/TR/soap/. Last visited: 13 August 2012.
[2] HTTP - hypertext transfer protocol overview. http://www.w3.org/Protocols/. Last visited: 13 August 2012.
[3] S. Sudarshan Abraham Silberchatz, Henry F. Korth. *Database System Concepts*. McGraw Hill, 2006.
[4] T. Berners-Lee and J. Hendler. Scientific publishing on the semantic web. *Nature*, 410:1023–1024, 2001.
[5] M.A. Chilton. Data modeling using entity relationship diagrams: A step-wise method. *Journal of Information Systems Education*, 17(4):385, 2006.
[6] J. De Bruijn, C. Bussler, J. Domingue, D. Fensel, M. Hepp, U. Keller, M. Kifer, B. König-Ries, J. Kopecky, R. Lara, H. Lausen, E. Oren, A. Polleres, D. Roman, J. Scicluna, and M. Stollberg. Web service modeling ontology(WSMO). W3C Member Submission 3 June 2005, 2005.
[7] E. Della Valle, D. Cerizza, and I. Celino. The mediators centric approach to automatic web service discovery of glue. *MEDIATE2005*, 168:35–50, 2005.
[8] J. Domingue, L. Cabral, S. Galizia, V. Tanasescu, A. Gugliotta, B. Norton, and C. Pedrinaci. Irs-iii: A broker-based approach to semantic web services. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(2):109–132, 2008.
[9] D. Fensel and C. Bussler. The web service modeling framework wsmf. *Electronic Commerce Research and Applications*, 1(2):113–137, 2002.
[10] D. Fensel, F.M. Facca, E. Simperl, and I. Toma. *Semantic web services*. Springer, 2011.
[11] D. Fensel, H. Lausen, A. Polleres, J. Bruijn, M. Stollberg, D. Roman, and J. Domingue. *Enabling Semantic Web Services: The Web Service Modeling Ontology*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
[12] WSML Working Group et al. D16. 1v1. 0. wsml language reference. *WSML Working Draft*, 2008.
[13] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, June 1993.
[14] M. Herold. Wsmx documentation. *Digital Enterprise Research Institute Galway, Ireland*, 3, 2008.
[15] R. Lara, D. Roman, A. Polleres, and D. Fensel. A conceptual comparison of wsmo and owl-s. *Web Services*, pages 254–269, 2004.
[16] B. Meyer. *Object-oriented software construction*, volume 2. Prentice hall New York, 1988.
[17] D. Ognyanov, A. Kiryakov, and J. Henke. Dip d2. 3: Ontology representation and data integration (ordi) framework. *DIP Integrated Project*, 2006.
[18] M. Stollberg, M. Hepp, and J. Hoffmann. A caching mechanism for semantic web service discovery. In *Proceedings of the 6th international The semantic web and 2nd Asian conference on Asian semantic web conference*, pages 480–493. Springer-Verlag, 2007.
[19] P. Szekely. Retrospective and challenges for model-based interface development. In *Design, Specification and Verification of Interactive Systems*, volume 96, pages 1–27, 1996.
[20] H.H. Wang, N. Gibbins, T.R. Payne, and D. Redavid. A formal model of the semantic web service ontology (wsmo). *Information Systems*, 2011.

Omid Sharifi is a Ph.D. research assistant of computer engineering department at Eastern Mediterranean University (EMU) in Famagusta, Cyprus, since September 2007. His research and experience encompass web semantics, specification and discovery of semantic web services, knowledge representation and ontology, constraint programming and evolutionary algorithms. He has developed and published several semantic agents. He graduated B.Sc. (2005) and M.Sc (2009) degrees from computer engineering department, also he is studying Ph.D. in computer engineering department at EMU. Address: Department of Computer Engineering, Eastern Mediterranean University, Famagusta, Cyprus. Email:omid.sharifi@emu.edu.tr

Dr. Zeki Bayram is a faculty member in the computer engineering department at Eastern Mediterranean University. He holds an Associate Professorship position in the department. Dr. Bayram obtained his Ph.D. degree in 1993 at the University of Alabama in Birmingham (U.S.A), and took a faculty member position at the computer engineering department of Bogazici University (Turkey). He joined Eastern Mediterranean University in 2000. His current research interests include logic, automated deduction, constraint programming, semantic web services and declarative programming languages. Address: Department of Computer Engineering, Eastern Mediterranean University, Famagusta, Cyprus. Email:zeki.bayram@emu.edu.tr