DATABASE INTEGRATION VIA FRAMEWORK BASED ON OWL

DEVIPRIYA RAJU

E-mail: Devipriya.k.raju@gmail.com

Abstract- A key goal of the Semantic Web is to shift social interaction patterns from a producer-centric paradigm to a consumer- centric one. Treating customers as the most valuable assets and making the business models work better for them are at the core of building successful consumer-centric business models. It follows that customizing business processes constitutes a major concern in the realm of a knowledge-pull-based human semantic Web. This paper conceptualizes the customization of service-based business processes leveraging the existing knowledge of Web services and business processes. We represent this conceptualization as a new Extensible Markup Language (XML) markup language Web Ontology Language-Business Process Customization (OWL-BPC), based on the de facto semantic markup language for Web-based information [Web Ontology Language (OWL)]. Furthermore, we report a framework, built on OWL-BPC, for customizing service- based business processes, which support customization detection and enactment. Customization detection is enabled by a business- goal analysis, and customization enactment is enabled via event – condition–action rule inference. Our solution and framework have the following capabilities in dealing with in-consistencies and misalignments in business process interactions:1)resolve semantic mismatch of process parameters; 2)handle behavioral mismatches which may or may not be compatible; and 3)process misaligned rendezvous requirements. Such capabilities are applicable to business processes with heterogeneous domain ontology; and 4) to integrate databases into the Semantic Web, we use Semantic Web ontologies to incorporate database schemas. An expressive first order ontology language, Web-PDDL, is used to define the structure, semantics, and mappings of data re- sources. A powerful inference engine, OntoEngine can be used for query answering and data.

Keywords- Business Process Customization, Composite Web Services, Consumer Centric, Goal Analysis, Ontology-Based, Semantics, Web Search, OntoGrate, Database, WebPDDL

I. INTRODUCTION

During the last several years, we have seen many achievements towards realizing the Semantic Web [5], where ontologies play a key role in defining the semantics of data. A knowledge framework offered by the Se- mantic Web should support a shift of social interaction patterns from a producer-centric paradigm to a consumer-centric one. The topic of human semantic Web has thus emerged to meet this need Besides developing Semantic Web ontology languages (e.g., OWL [2]) and applications (e.g., web agents and services), it is necessary to make existing data resources, such as databases, available for Semantic Web applications to access and share. Currently, relational databases are some of the largest data resources in the world, but the structure and integrity constraints of relational tables are defined by schemas, which are not as expressive as ontologies when representing the semantics of data. To deal with both the expressivity gap between schemas and ontologies and the syntax difference between different definition languages, we apply our ontology-based information integration system, Onto Grate, and thereby integrate relational databases into the Semantic Web. We refer to the customization of business process as a machine - enabled capability of adapting a business process of a company according to the process of the customer or business partner that it is collaborating with.

Our efforts reported in this paper seek to establish a generic solution to the problem of customization of service- based processes from the following three aspects. First, we present a conceptualization definition for business process customization that leverages existing knowledge of business processes and Web services. Second, we present a representation of this conceptualization in a new Extensible Markup Language (XML) markup language, based on the de facto semantic markup language for Web-based information, i.e., OWL. Third, we present a framework for customizing service-based business processes based on OWL- BPC by first identifying the possible causes of discrepancies/ inconsistencies between collaborating business processes (customization detection) and then taking suitable remedial actions (customization enactment).

Then, we are extending the expressivity of Semantic Web ontologies to incorporate database schemas, defining both structure and semantics. Here secondly, we are using machine learning and data mining techniques to pro- vide more meaningful mapping rules to users, and stimulate further interaction with domain experts to justify the rules and make them executable. Third, our inference engine, OntoEngine, is being extended not only to help check the consistency and redundancy of mapping rules, but also to conduct optimized query answering and data translation. Once users in multiple domains generate large amounts of metadata (e.g., mapping rules), we expect to provide online services that make available to the public the storage and efficient access to large collections of metadata.

The Semantic Web has been a vision that helps shape the future directions of many research topics in computer science and information systems. For example, there is analysis work dedicated to a Web Services Business Process Execution Language (WS- BPEL) [5]. One issue is that the design process must be user centered. The testing must be user centered, and the interface of the final software product must be user centered [8]. Another issue is that software in different domains may demonstrate different user-centered requirements. A branch of research efforts on semantic Web seeks to integrate a machine-

understandable knowledge framework with the user-centric human factors. This is called “human semantic Web”.

II. PREVIOUS WORK

2.1 Schemas, Ontologies, Web-PDDL

In previous work, we have described in some detail how to represent database schemas as ontologies and how to use logic axioms to merge two database schemas. We use Web-PDDL, a strongly typed first order logic language with Lisp-like syntax, to internally descr ibe and process these representations. We now briefly review and summarize this work. Consider an Informix database schema, Stores7, from the online sales domain, in Figure 1. It defines the relations, such as customer, order and item, and associated attributes. Although Web-PDDL was originally designed as a language for representing ontologies, their mappings, data in stances, and queries on the Semantic Web,

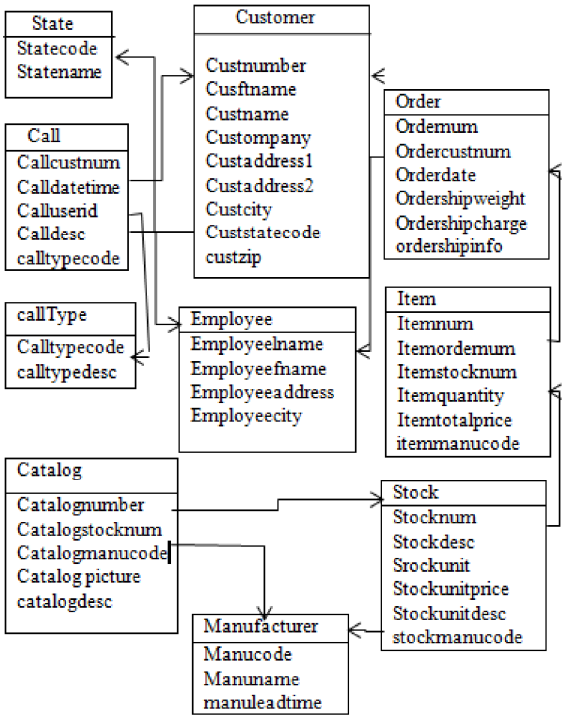

Figure 1. The database schema of Stores7

It also can be used to represent Stores7 as an ontology as shown in Figure 1.

```
(define (domain stores7-
   ont) (:extends
(uri "http://www.cs.uoregon.edu/~paea/research/sql.pddl#"
        :prefix sql) ...)
   (:types
      Customer State - @sql:Relation
      String - @sql:Varchar
   ...) (:predicates
      (customerfname x - Customer y - @sql:varchar)
      (customerlname x - Customer y - @sql:varchar)
      (customerstatecode x - Customer y -
      @sql:varchar) ...)
   (:axioms
      (forall (c - Customer code
         @sql:varchar) (if (customerstatecode
         c code)
            (exists (s - State) (statecode s code))))...)
   (:facts
      (@sql:primarykey Customer "customernumber") ...))
```

Figure 2. The Stores7 schema as ontology.

Some of the highlights of the Web-PDDL syntax include:

- *Inheritance*: The :extends declaration in Figure 2 expresses that Stores7 inherits features from the “sql” domain, which defines concepts for relational databases such as relations, data types, and aggregate functions. Therefore, the inheritance capability of Web-PDDL allows us to incorporate some of the more common and desirable features of SQL.
- *Namespaces*: To avoid symbol clashes, symbols imported from other ontologies are given prefixes, such as @sql:Relation .These correspond to XML namespaces, and when Web-PDDL is translated into RDF, that is exactly what they become.
- *Types*: Types start with capital letters. A type T_1 is declared to be a subtype of a type T_0 by writing “T_1 – T_0” in the: types filed of a domain definition.
- *Predicates*: Predicates correspond roughly to properties in the web ontology language, but they can take any number of arguments.
- *Axioms*: The axiom in the Figure 2 is the Web-PDDL expression for the foreign key relationship between customer and state, where customerstatecode is the foreign key.
- *Functions*: There are also functions in Web-PDDL,inc-luding Skolem functions and built in functions such as + and -, that can be evaluated when appropriate.

• *Facts*: Assertions (facts) are written in the usual Lisp style: e.g. (@sql:primarykey Customer “customer- number”) states that the primary key attribute of the Customer relation is “customernumber.”

In our initial findings, a few simple rules relating schemas to ontologies can accomplish the majority of trans-formations. These rules also play an important part in developing the SQL wrappers (PDDSQL) for OntoGrate architecture:

Relation	↔	Type
Attribute	↔	Predicate
Integrity Constraint	↔	Axiom
Primary Key	↔	Fact

Although these simple rules appear to work well for rudimentary transformation, we believe further work is required to capture more subtle database semantics.

III. CONCEPTUALIZATION OF SERVICE – BASED PROCESS CUSTOMIZATION

A general solution to the problem can be described as follows: A business process needs

to communicate with another by calling its Web services and exchanging XML messages with it. In order to meet this need, it has to observe its partner within the scope of the relevant collaboration. Then, it customizes its own process accordingly in order to ensure a smooth collaboration.

Automatic Customization Detection: Automatic customization detection is an automated process of detecting possible elements or variables of a business process that need to be especially treated in order to suit the requirement of the other process (es).
Automatic Customization Enactment: Automatic customization enactment is an automated process of taking actions to perform the customization on the PBP according to the detected customization spots and the automatic reasoning on the customization conceptualization knowledge framework. Metadata and Reasoning: We use resource descriptions to define concepts in customization and relations between them using OWL-S statements. We connect these statements to form a s emant i c network. A basic statement in the RDF is a<subject, property , object> triple, which models a relation in the metadata. Its definition in the RDF is shown in Fig. 2.

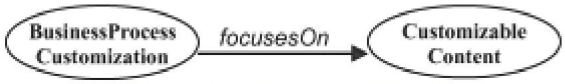

Fig-3. Concepts and relations of Metadata.

A final thing that we need is a "bridge" that translates the formal semantics in processing information encoded in OWL-S

2.2 Merging Ontologies with Bridging Axioms
A merged ontology consists of common elements from a source and target ontology, but also defines the semantic mappings between them as bridging axioms [12]. A merged ontology allows all the relevant symbols in a domain to interact with each other, so that facts can be translated from one ontology to another using inference over the bridging axioms.

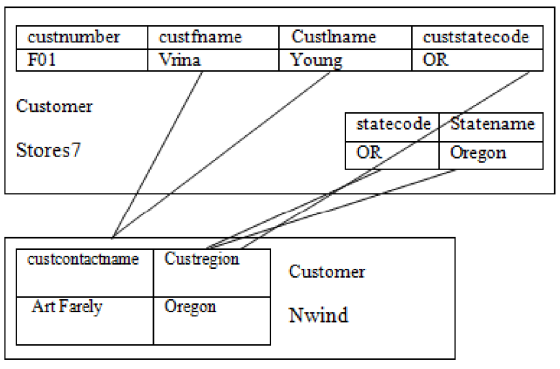

Figure.4 Portions of two database schemas and the mapping between them

Suppose we have a different s a l e s s chema , such as Nwind from Microsoft. Although Stores7 and Nwind are similar schemas, they do have some differences. We can write bridging axioms in Web-PDDL to express mappings between Stores7 and Nwind. Consider, for example, the mappings depicted in Figure 3. The three ways mapping from NWind's region to Store7's state name and state code informat ion can be expressed in Web PDDL as:

(forall (x - @nwind:Customer y - @sql:varchar) (if
(@nwind:customerregion x y)
(exists (z - @stores7:State t - @sql:varchar) (and
(@stores7:customerstatecode x t)
(@stores7:statename z y) (@stores7:statecode z t)))))

We can finally put all bridging axioms into the new, merged ontology, called Stores7-Nwind.\

2.3 Inferential Data Integration
Given a merged ontology between two sources expressed in the first order ontology language, Web-PDDL, we can then perform integration using first order theory. In this section, we specify the problem of integration and show how inference can solve it, forming the basis of our inferential data integration model.

• Query Translation: The process of extracting data ex- pressed by one schema to answer a query posed using another schema, also known as query answering.

• Data T r a n s l a t i o n: Translating data from a source schema to a target (or integrated) schema for the purpose of information exchange

III. ONTOGRATE ARCHITECTURE

We are developing OntoGrate, a system that, given different but related ontologies or schemas and their associated data, will be able to learn or mine a set of first order mapping rules that accurately describes how the input ontologies or schemas relate to each other. These rules will be used by OntoEngine to perform data integration. For ex- ample, different genomic databases, which are used to study NIH model organisms such as the zebrafish, mouse and fruit fly, and different genomic ontologies could benefit from integration when used to answer more interesting questions geneticists may have about the human genome.

To explain how OntoGrate can help domain experts (e.g., geneticists) integrate their databases and Web resources, we explain the architecture of OntoGrate, shown in Figure 4.
It is composed of six modules and interfaces with four different kinds of data resources related to our specific aims.

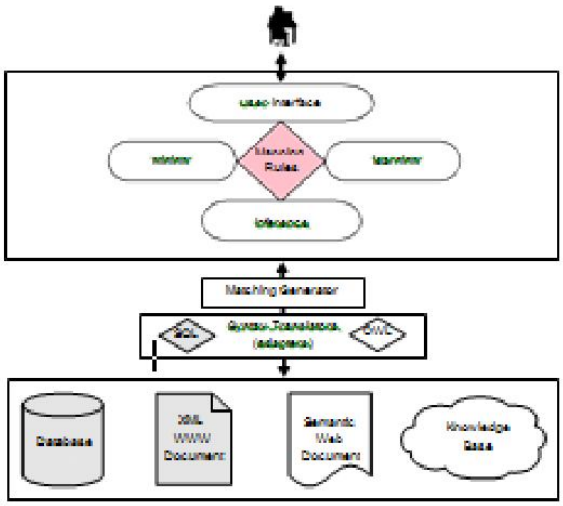

Figure 5.Architecture of OntoGrate: The system is composed mainly of six modules: Learning Module, Mining module, machine generation module, syntax translators, user interface module.

Integration of schemas and ontologies: Since databases are defined by schemas, which focus more on structure than semantics, we first need to build an automatic translator to represent schemas as ontologies.

We have discovered some basic heuristics for transforming schemas into ontologies. These heuristics can be embedded into an automatic tool to perform most schema trans- formations, and it has worked well so far in our preliminary database experiments. While it appears that this task can be automated, we anticipate some theoretical challenges, especially given some application-oriented database schema designs with complex constraints, such as biomedical databases. User interaction may be required to capture subtle semantics. Matching (correspondence) generation: To derive candidate matchings for users, we are building a matching generator based on the names, structure, and relationships of concepts in ontologies (schemas). A user interface will depict the ontologies and candidate matchings based on the input ontologies and data. The system will then enter into an interactive loop with the learning and mining modules, during which the user will make further refinements.

Learning mappings from the knowledge of domain experts: Based on the matching suggestions, domain experts may be able to specify simple relationships, such as "equal," "subclass" or "subproperty." However, more complex relationships may be too subtle for a domain expert to specify accurately at first. We expect machine learning to be helpful in this case: all a user has to do is to provide specific examples and let the system generalize them into formal rules expressed in first order logic. In general, we expect that this learning module will benefit from the extensive literature that deals with learning in first order logic (Inductive Logic Programming). Well-suited learning algorithms will have to be developed, depending on (i) the amount of information available, (ii) the quality of user input, (iii) the complexity of mapping rules needed, and (iv) the complexity of the input ontologies.

Mining large data sets to find candidate mappings: We are using association rule mining in the OntoGrate system to discover candidate mappings, which the user can then select or refine to generate final executable mapping rules. Association rule mining [3] is a data mining technique that has been successfully used to find frequent pat- terns in large data sets. For instance, this technique might help discover patterns in different databases that share some partial data, such as different gene databases that share the same GenBank [1] ID. Similarly, me d i c a l databases use term IDs to point to medical terms in UMLS.

User Interface: Interaction between the system and do- main experts through a user interface is important to realizing our vision of an interactive integration system. Our goal is to have the user interface module present information about input ontologies or data resources, the current candidate mappings generated by data mining and machine learning, and any intermediate and final data translation or query answering results in a clean, concise, and accessible fashion.

Also, the user interface will be designed to facilitate user feedback, such as having the user select nodes in the ontologies or facts in the databases, choose mapping rules mined by the data mining module or learned by the learning module to refine or verify them, or select suggestions from the matching generation module to further explore and incorporate. Inference Engine:

The inference module is in charge of using mapping rules to answer queries and exchange (i.e., translate) data among available data sources.

It will also be used for rule refinement, that is, to check the consistency and redundancy of the generated rule set. OntoEngine is a special purpose first order theorem prover developed for ontology translation on the Semantic Web [12].

We are ex- tending it for the new purposes mentioned in this paper. In particular, the output from the inference engine can be fed back into the learning and mining processes for better results.

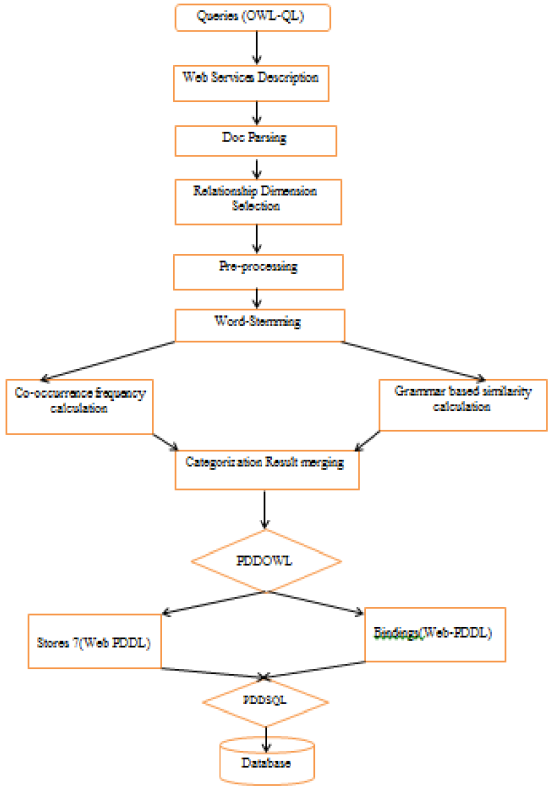

Fig 6: Integrating databases into semantic Web

Fig-7.Screenshot

we can define mapping rules as bridging axioms and merge the schemas (ontologies) together. The data integration can then be implemented as an inference process by our special purpose theorem prover, OntoEngine, with the help of syntax wrappers (i.e., PDDSQL and PDDOWL). Our new OntoGrate framework not only has the advantage of the rich expressiveness of first order logic for conceptual modeling, but also exploits both the desirable features of SQL and OWL. We elaborate on our two approaches to query sales databases from the Semantic Web applications in two scenarios. This testing so far demonstrates that OntoGrate is promising for integrating databases into the Semantic Web. In the immediate future, besides implementing other research aims described in section 3, the scalability and efficiency of OntoGrate needs to be investigated in larger-sized relational databases of various domains. Also, another interesting integration task is to integrate current XML documents in the world-wide web with the Semantic Web in the OntoGrate framework. We believe this will be more difficult than integrating relational databases into the Semantic Web, since not all XML documents have XML schemas (which we can treat similarly as relational schemas) or DTD definitions. We plan to tackle these challenging problems in our immediate future work while making progress on the other important aspects of our system outlined in section 3.

REFERENCES

[1] GenBank Database. http://www.ncbi.nlm.nih.gov/Genbank/.

[2] OWL Web Ontology Language. http://www.w3.org/TR/owl-ref/.

[3] R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. In Very Large Data Bases (VLDB) Conference, pages 487–499. Morgan Kaufmann, 1994.

[4] M. Arias and R. Khardon. Complexity Parameters for First Order Structures. To appear in the Machine Learning Journal, 2005.

[5] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. Scientific American, 284(5), May 2001.

[6] P. A. Bernstein and E. Rahm. Data warehouse scenarios for model management. In ER 2000, pages 1–15, 2000.

[7] D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi, and R. Rosati. Knowledge representation approach to information integration. In Proc. of AAAI Workshop on AI and Information Integration, pages 58–65. AAAI Press/the MIT Press, 1998.

[8] R. Dhamankar, Y. Lee, A. Doan, A. Y. Halevy and P. Domingos. iMAP: Discovering Complex Mappings between Database Schemas. In Proceedings of the ACM Conference on Management of Data, pages 383–394, 2004.

[9] A. Doan and A. Y. Halevy. Semantic-integration research in the database community: a brief survey. AI Magazine, 26(1):83–94, 2005.

[10] A. Doan, J. Madhavan, P. Domingos, and A. Y. Halevy. Learning to Map Between Ontologies on the Semantic Web. In International World Wide Web Conferences (WWW), pages 662–673, 2002.

[11] D. Dou and P. LePendu. Ontology-based Data Integration for Relational Databases. 2006. To appear in ACM SAC'06 DTTA Track.

[12] D. Dou, D. V. McDermott, and P. Qi. Ontology Translation on the Semantic Web. Journal of Data Semantics, 2:35–57, 2005.

[13] E. Dragut and R. Lawrence. Composing mappings between schemas using reference ontology. In Proceedings of International Conference on Ontologies, Databases and Application of Semantics (ODBASE), 2004.

[14] L. M. Haas, M. A. Hernandez, H. Ho, L. Popa, and M. Roth. Clio Grows Up: From Research Prototype to Industrial Tool. In Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, pages 805–810, 2005.

[15] Bhavithra, DeviPriya, MahaLakshmi.: Semantic web search for business process customization using OWL.