Storing Multifaceted Ontologies with Natural Language Addressing

Viacheslav Lanin  
Business Informatics Department  
National Research University Higher School of Economics  
Perm, Russian Federation  
vlanin@hse.ru

Lyudmila Lyadova  
Business Informatics Department  
National Research University Higher School of Economics  
Perm, Russian Federation  
llyadova@hse.ru

Irina Shalyaeva  
Business Informatics Department  
National Research University Higher School of Economics  
Perm, Russian Federation  
ishalyaeva@bk.ru

Abstract — Multifaceted ontology definition is suggested and an approach to storing multifaceted ontologies in the ontology-driven system for monitoring processes based on Internet sources is presented. This system can be adapted for events analysis and global processes monitoring, for social network analysis and etc. Ontology-driven architecture intends using an ontology. Ontologies are used to search data in the Internet sources with information retrieval tools and to structuring information according to the data structure defined by users. The multifaceted ontology combines different types of ontologies: the sources ontology defines information sources used by experts for analysis; the domain ontologies describes research interests of the experts needing analysis (they describe ontology of problem including domain ontologies); data formats are described with structures ontology used for structuring data. Multifaceted ontology is the information kernel of the analysis system. Researchers need tools for effective storing and processing information. An approach based on natural language addressing is proposed as perspective method of multifaceted ontology storing. Data structures and algorithms are described and discussed.

Index Terms — multifaceted ontology, natural language addressing, data extraction, data structuring.

I. INTRODUCTION

Political, social and economic public processes need to be analyzed in the context of real global social-natural processes, considering all aspects as components of the global system. Thus, the task of monitoring of the phenomena, processes in different areas, detections of dependences and tendencies becomes especially urgent now.

Global processes are considered as the processes initiated by human activities in different spheres and enveloping essentially all space of the planet and society. Existence of social, political, economic and natural interactions in scales of large regions and the whole planet gives to the considered processes (and to problems) the status of globality characterizing unity of cross impact multiple-factor of communications in social, biological and technical sphere. Global process can include different events as components. Global processes evolve, create new tendencies and problems.

The approach to development of the system intended for formation of global processes visual models on the basis of searching of events, their possible reasons and consequences, significant for the user, is offered [1].

Boundaries of processes are defined by user (area of analyst's interests), and “starting point” for information search and monitoring of events which can be included in the investigated processes is the event interesting the user. User formulates request extended automatically with ontology and results of information retrieval describing related events are included to ontology.

Any significant event receives very fast response in news. News reports represent the status of events almost “in real time”. Texts extracted from news describe events in a natural language, enveloping all aspects of the events and phenomena. Internet news are one of the largest information sources about the modern society. News texts contain a lot of factual data, being one of the best data sources for the existing methods of text processing. News lines don't specialize in certain event classes, so, news can be data source for any scale and expert’s interest areas, given by users (they can select one and more data domains, types of events interesting them).

Events of different domains and attributes characterizing them (dates, places of events, participants) can be derived from texts of news. The modern text mining systems successfully extract information on events, allow to define relations between events and to build chronological chains of events.

Extracted data fills up ontology developed by user. Multifaceted ontology is the information kernel of the event analysis system. Researchers need tools for effective storing and processing information on events at analysis process. An approach based on natural language addressing (NL-addressing) is proposed as perspective method of ontology storing in the event analysis system [2].

II. MULTIFACETED ONTOLOGY DEFINITION

Multifaceted ontology is a kernel of a system focused on an interdisciplinary research. Users which develop the ontology can describe different aspects of their research. And specialists engaged in various data domains are given opportunity to
develop appropriate ontology parts focusing on their task in the overall research project.

Let’s define a multifaceted ontology as an ontology joining describing various aspects of a research data domain
\[ MFO = \text{Join}_R(O_1, \ldots, O_n) \]

where \( O_1, \ldots, O_n \) are ontologies describing specific aspects of the research and \( \text{Join}_R \) is a rule for ontology joining set by users. Ontology joining rules are presented also as an ontology which is especially developed for each particular research. This ontology includes concepts and relations that reflect the developer requirement important to a study.

For example, in a described system for analyzing global processes based on information extracted from sources on the Internet, three identified by users research aspects are considered: 1) interesting for researchers Internet data sources; 2) types of events in a particular data domain (or several data domains in interdisciplinary research), between which the relationship is established; 3) data structures that can be used for analysis.

Examples of ontologies for the study of global processes associated with environmental disasters and data structures extracted from sources on the Internet (event log formats used in the construction of process formal models by means of Process Mining) are given in [3]. A part of the described ontologies joining is shown in figure 1.

To solve the project problems, it is necessary to find information about events related to the different data domains (for example, environmental disasters \( O_2 \) and economic risks \( O_3 \)). When joining in the \( \text{Join}_R \) ontology, «an event» element is created that combines the events described for different areas (only a specific type of events in each area can be selected, for example, oil spills). For this element a link is created to the concept representing the event in the event log ontology \( O_1 \). This relationship allows to define the structure of information about events retrieved from Internet sources. Thus, when developing the project ontology \( \text{Join}_R \), objects and links are created that reflect the project tasks solving peculiarities.

Using previously created ontologies, users have the opportunity to create their own resources that reflect the specifics of research and to analyze data using various tools. The architecture of the global process monitoring system is given in [1]. Multifaceted ontologies allow flexible adjustment to the solution of research problems. However, this solution requires efficient data storage and processing.

![Fig. 1: An example of Join Ontologies](image)

### III. Storing Ontological Resources

Storing and processing databases and knowledge bases have significant differences. Traditionally enterprise information systems use variety of database management systems (DBMS) for storing data. Such systems are oriented for high performance, scalability and other essential for enterprise solution requirements. Systems based on knowledge such as ontology are not widespread in an enterprise area, because of existing solutions commonly are research projects. Performance issues are going to play quite important while processing large amounts of information.

Our project deals with ontology resources in RDF and
OWL format. There are two main ways to storing and handling such kind of resources. First one is using ordinary file via operation system services. The main disadvantage of the approach is low extensibility and scalability. Second way is using specialized system for handling ontologies, but in this case there is a problem of choosing of variety technological solutions with different data models and of physical and logical implementation of data representation methods.

The modern ontology processing technologies are based on the PDF format. There are four widespread formats for representation RDF triples [4]:

- **Notation 3 (N3)** format extends the RDF data model by adding formulae (literals which are graphs themselves), variables, logical implication, and functional predicates, as well as providing a textual syntax alternative to RDF/XML. N3 is a very complex language in order to store RDF-Triples.
- **N-Triples** is a recommendation of W3C which suggests a line-based syntax for an RDF graph. It is a subset of N3 in order to reduce its complexity.
- **Turtle** is a recommendation of W3C which suggests a line-based syntax for an RDF graph. It is a subset of N3 in order to reduce its complexity. Turtle provides levels of compatibility with the existing N-Triples and Notation 3 formats as well as the triple pattern syntax of the SPARQL.
- **RDF/XML** is XML syntax for RDF in terms of Namespaces in XML, the XML Information Set and XML Base.

The RDF format is undoubtedly important for the export and import, representation data in human-readable form but storage system features play the main role. The following system types can be used for solving the tasks of ontology storage and processing:

- **In-memory storage** allocates a certain amount of the available main memory to store the given RDF data. The advantages of In-memory storage are implementation simplicity and quick response. However, keeping data in memory can lead to its loss and resources lack.
- **Native storage** is a way to save RDF data permanently on the file system. These implementations may fall back on (in this terms) well investigated index structures, such as B-Tree. This kind of storage has been specially designed to store RDF triplets and its implement special processing capabilities inherent in the RDF graph structure. However, the main disadvantage is significant increase of response time while processing large ontologies.

- **Relational database** storage makes use of widespread relational database systems (PostgreSQL, MySQL, Oracle and so on) to store RDF data permanently. This approach relies on research results in the database domain. Two different mapping strategies are passable. The first one is an universal table, which contains all RDF triples. The second strategy is to create a mapping of the ontology into a table structure, consequently number of tables must be equal to number of concepts of ontology. Also, it is necessary to take into account that ontology can hold both data domain classes and its instances, so changes in data domain leads to changes the ontology this in turn causes a change in the structure of the tables.

- **NoSQL database** provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. There are such kinds of NoSQL database as graph database, document store, key-value and many others (http://nosql-database.org/index.html). Comparing results are presented in Table 1.

### IV. GENERAL ARCHITECTURE OF MONITORING SYSTEM

Figure 2 represents the general interaction scheme of ontological resources, forming the Knowledge Base (KB), with the data source – the Internet, and the target system into which the processed data is imported.

The KB is used in the process of extracting data and generating an event log. The source ontology is used to provide the user with the ability to select different/multiple sources of information. The event log ontology describes the required data format used by the system implementing Process Mining methods. Since different systems can use different event log formats, and the current standard of the ProM system can also be changed, it was decided to store the current format as an ontology for the possibility of convenient manual modification. The ontology of the event log is used at the stage of XES file generation.

<table>
<thead>
<tr>
<th></th>
<th>In-memory storage</th>
<th>Native storage</th>
<th>Relational database</th>
<th>NoSQL database</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Scalability</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Platform support</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>API</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Access control</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Logical inference</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Transactions</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Aggregation and mathematical calculations</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
The figure also presents the five major operations involving a knowledge base:

- retrieving data according to the web-source structure described in the source ontology;
- filling the problem ontology;
- creating an event log in the RDF format using event ontology;
- import of event log into Process Mining tools for later use;
- generation of an extended query to Internet search engines.

**Task ontology** contains all the necessary information for extracting and structuring the data about events (their order, characteristics). To extracting event from the text, it must be described in task ontology - thus, the expansion of the subject area, increase of the data set and even each single use of the system, will cause fast knowledge base spreading (the mechanism of the semi-autonomic expansion of the task ontology proposed in [1].

Since it is the task ontology that contains both the content and the structure of future models (the causal relationship is also represented in the ontology by means of “caused by” and “consequence of” properties), it is involved in all stages of searching, extracting and results generation. Operations on the problem ontology will be the most frequent actions within the system. Therefore, to demonstrate the approach aimed to optimize the processing of the knowledge base, it was decided to use natural language addressing. The following are the operations performed on the ontology of the problem.

- **Subject bag** contains a description of all denominations represented in the ontology (objects subjects, relation).
- For relationship storage, **Relation bag** are used. As there can be several **Objects** for one pair of **Subject-Relation**, the **Relation bag** consist of **Object bag**. **Object bag** contain the object hash-value and its text description (this will allow to form the final result directly, without re-traversing the Subject bag).
- Since the task ontology is extensible and can contain an arbitrary number of classes, individuals and relationships, and individuals also have multiple relationships with other individuals, for example “isSynonymous”, so for each class and instance, not only Subject bag but also Inheritance bag to demonstrate relation type “is_a”.

**V. COMMON ONTOLOGY STORING ALGORITHMS**

Data structure realized for ontology storing is presented in Figure 3.

**Ontology filling** (Figure 4) according to common storing algorithm based on natural language addressing (NL-addressing):

1. Read collocation (for example “oil spill”).
3. Before writing objects to ontology, they are searched for, if at least one of them is present in the ontology, the class and type of connection of the second object is determined.

If such objects are not already in the ontology, then it is necessary to determine at the text processing stage, which class they belong to, then find the type of connection for appropriate classes and continue the algorithm.
Fig. 3: Data structure for NL-addressing.

Fig. 4: Ontology filling.
4. During the search, it was determined that "oil" belongs to the class “Material” which already has relationship "hasMaterial" with "spill" individual, so word “oil” was added to the appropriate archive of values.

5. Store the structures: – {NC} in the “subject” archive using the path (NP, NO).

Search through ontology according to common reading algorithm based on NL-addressing (user request: "oil spill"):  
1. Read collocation «oil spill».
3. Perform NL-archives traversal, using values of each object individually.
4. Extend the search query with the data found.
   Possible extended requests: “petrol spill”, “Mexico petrol spill”, “Mexico oil spill”. Algorithm is illustrated in Figure 5.

VI. CONCLUSIONS

The expected advantages of using NL-addressing are two main achievements:
- High speed for storing and accessing the information.
- The possibility to access information with dynamic changing structure immediately after storing without recompilation the database and rebuilding indexes.

For static structured datasets it is more convenient to use standard utilities and compiled indexes. NL-addressing is more suitable for dynamic processes of creating and further development of structured datasets due to avoiding recompilation of the database index structures and high-speed access to every data element. The loss is additional memory for storing data structures.

REFERENCES


