Semantic Stored Procedures Programming Environment and performance analysis

Marjan Efremov[1], Vladimir Zdraveski[2], Petar Ristoski[2], Dimitar Trajanov[2]

[1] Open Mind Solutions – Skopje, bul. Kliment Ohridski 20a/1-3, 1000 Skopje, Republic of Macedonia
[2] Faculty of Electrical Engineering and Information Technologies – Skopje, Rugjer Boskovic bb, 1000 Skopje, Republic of Macedonia

marjan.efremov@oms.com.mk, vladimir.zdraveski@feit.ukim.edu.mk, petar.ristoski@feit.ukim.edu.mk, dimitar.trajanov@feit.ukim.edu.mk

Abstract. Supporting the idea of the semantic web, we developed a Semantic Stored Procedures (SSP) programming environment for the Oracle11g database, similar to the existing ones, available for the stored procedures in the relational databases. This show-case SSP-environment supports the basic test/store/execute/remove scenarios for sem_match-queries, but is also a quite extensive system. An eclipse plug-in that provides a GUI for these operations and java API which helps programmers to use the SSP-call functionality were released in the package with the Oracle11g SSP-extension. The performances of our system are comparable (much better in most of the cases) with the existing adapters, as shown with the measurements in comparison with the well-known Jena adapter. Thus, a complete development environment for SSP-programming is now available and the semantic query management, as a simple concatenation of strings inside the code, seems to becoming just a forgotten nightmare of our semantic web programmer's life.

Keywords: Semantic Web, Semantic Database, Stored Procedure, Sem_match, Eclipse Plug-in

1. Introduction

The technologies of the Semantic Web allowed development of novel approaches to data storage and retrieval. A semantic search system is essentially an information retrieval system which employs semantic technologies in order to enhance different parts of the information retrieval process. This is achieved by semantic indexing and annotation of content, query expansion, filtering and ranking the retrieved information [1]. The semantic search also introduces additional possibilities, such as search for an online ontology [2], search for online (distributed) knowledgebase, retrieval of facts from the ontology and knowledgebase and question answering.

In terms of providing better database support for the Semantic Web based applications, Oracle invented the Oracle 11g semantic database [3], which provides environment for storage and retrieval of semantic data, represented as RDF triples [4]. Together with the database, a Jena adapter for Oracle Database was written [5]. By use of the adapter, any Jena-based semantic application can be migrated on the Oracle 11g database. The performance and opportunities obtained by the Oracle 11g are comparable with the basic Jena (file based) solution. Retrieval is made by use of Jena Graph API [6] or with direct use of SPARQL [7]. With the use of SPARQL appears the problem, which is the impossibility of storing the queries inside the database. In the current environment, a query is managed as a simple string that is quite difficult scenario for the programmer.

Our solution of the problem is based on the existing stored procedures philosophy. A stored procedure (sometimes called a proc, sproc, StoPro, StoredProc, or SP), a subroutine available to applications accessing a relational database system, is a very essential part of the software, when any kind of a database storage is used.

We succeeded to implement a semantic stored procedures programming environment for sem_match queries. Sem_match is a query format, used by the Jena adapter for Oracle 11g (to query the database, programmer's SPARQL is converted to sem_match and then sent to the database), and is comparable with the SPARQL in terms of power, but more powerful in terms of response time. We designed a simple eclipse plug-in [8] as a user frontend for query testing, storing and removing scenarios.

2. Related Work

The Oracle 11g is the latest database release by Oracle, leading and only commercial database with native RDF/OWL data management. It is open, standards-based, scalable, secure, reliable and performing RDF management platform. Based on a graph data model, RDF data (triples) are persisted, indexed and queried, like other object-relational data types. Application developers use the power of the Oracle Database 11g to design and develop a wide range of semantic-enhanced business applications. [9] Stores rich, real-world relationships in the data beyond columns, table joins and Boolean to obtain more semantically complete query results. Enables machine-driven discovery of new relationships using the native Oracle Database inference engine, ontologies, and RDFS /SKOS/OWL semantics and user defined rules.

Inferred data is persisted in the database for faster querying. [10] Oracle Database Strengths are Scalability (Trillions of triples), Availability (tens of thousands of users), Security (protect sensitive business data), Performance (timely load, query & inference), Accessibility (to enterprise applications) and Manageability (leverage IT resources) [11].

On the other hand, there is the well-known Jena. It is a Java framework for building Semantic Web applications and provides an API to extract data from and write to the RDF graphs. The graphs are represented as an abstract "model". A model can be sourced with data from files, databases, URLs or a combination of these. A Model can also be queried through SPARQL and updated through SPARUL [12]. Jena also provides support for OWL [13].

Sesame [14] is another existing standard framework for processing RDF data. It can be deployed on top of a variety of storage systems, relational databases, in-memory, file systems, keyword indexers, and offers a large scale of tools to developers to leverage the power of RDF and related standards. Sesame fully supports the SPARQL query language for expressive querying and offers transparent access to remote RDF repositories using the exact same API [15] as for local access.

Adapters that enable semantic frameworks to be deployed over Oracle Database 11g were also released. The Jena Adapter for Oracle Database provides a Java-based interface to Oracle Database Semantic Technologies by implementing the well-known Jena Graph and Model APIs. Similarly, The Sesame Adapter for Oracle Database [16] integrates the popular Sesame Java APIs with Oracle Semantic Technologies support.

All these tools are quite complete and easy to use, except that you have to build your queries as a concatenation of strings and send the whole query every time you need to execute it. Thus, the management of the queries becomes quite chaotic in large applications and sometimes it is almost impossible to code and/or make changes. Our main problem is that there isn't (wasn't) an environment similar to the stored procedures framework or of any other kind that provides those functionalities for the Semantic Web querying process.

3. Solution Description

In the context of solving the described problem we developed a simple show-case Semantic Stored Procedures Framework and implemented the most required operations and tools.

The system architecture, shown in Fig. 1, consists of the main block, the Semantic Stored Procedures (SSP) Adapter placed inside the Oracle 11g and provides the whole functionality related with the semantic database. Further, the eclipse plug-in was designed to provide interaction with the adapter during the development phase and make it easier. And at the end, classes and methods (the SSP API) for a Semantic Stored Procedures call were written.

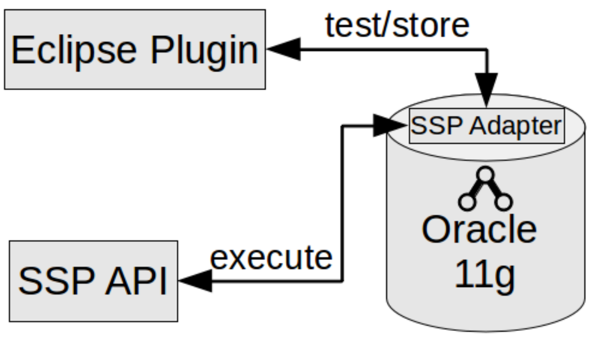
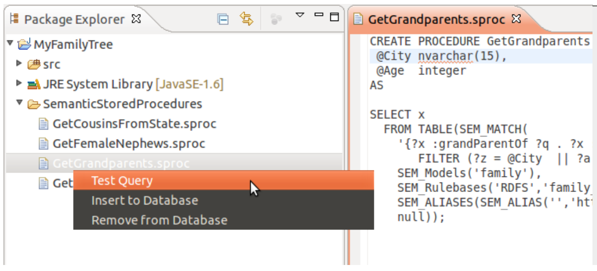


Fig. 1 – The System Architecture

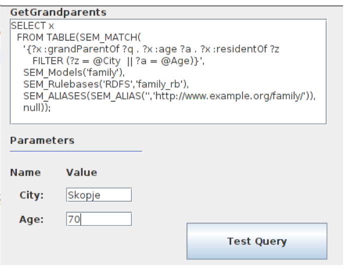
3.1. Client Side (the eclipse plug-in)

The user of the plug-in is able to create a new semantic stored procedure via the New File GUI Wizard. After completing his sem_match query in the editor (shown in Fig. 2), he could test it, store it and remove it from the database directly via the GUI. These commands are shown in the context menu in Fig. 2. The server address and other server access parameters are red from an xml configuration file, placed inside the client project.



```
CREATE PROCEDURE GetGrandparents
 @City nvarchar(15),
 @Age  integer
AS

SELECT x
  FROM TABLE(SEM_MATCH(
    '{?x :grandParentOf ?q . ?x
       FILTER (?z = @City  || ?a
    SEM_Models('family'),
    SEM_Rulebases('RDFS','family_
    SEM_ALIASES(SEM_ALIAS('','ht
    null));
```



```
SELECT x
  FROM TABLE(SEM_MATCH(
    '{?x :grandParentOf ?q . ?x :age ?a . ?x :residentOf ?z
       FILTER (?z = @City  || ?a = @Age)}',
    SEM_Models('family'),
    SEM_Rulebases('RDFS','family_rb'),
    SEM_ALIASES(SEM_ALIAS('','http://www.example.org/family/')),
    null));
```

Name	Value
City:	Skopje
Age:	70

Fig. 2 – The Eclipse plug-in **Fig. 3 - The variable parameters**

If the query contains variables, then the “Test Query”-command opens a dialog (shown in Fig. **3**) that requires values for the variables to be entered, so the query can be tested properly.

3.2. Server Side

First of all, we must set the semantic database, create a semantic model, create a table which will hold the semantic data, create a semantic entailment, some additional custom rule bases etc. [17] After the prerequisite step the database is ready to be queried.

On the server side four functionalities are implemented. They allow testing any SEM_MATCH query, creating and storing a procedure, execute previous created procedure (it can have variables in sem_match query and in runtime will be replaced with concrete values) and delete a previous saved procedure from database. All of this is programmed in stored procedures and kept in the database, on the server machine. The goal is all of the requested queries to be executed on the server side, the client just send the demand throw the plug in.

RDF/OWL data can be queried using SQL. The Oracle SEM_MATCH table function, which can be embedded in a SQL query, has the ability to search for an arbitrary pattern against the RDF/OWL models, and optionally, data inferred using RDFS, OWL, and user-defined rules. The SEM_MATCH function meets most of the requirements identified by W3C SPARQL standard for graph queries. [18] This function has the following attributes:

```
SEM_MATCH(query VARCHAR2,
        models SEM_MODELS, rulebases SEM_RULEBASES, aliases SEM_ALIASES,
        filter VARCHAR2,  index_status VARCHAR2,  options VARCHAR2)
RETURN ANYDATASET;
```

Only the query attribute is required. The other attributes are optional (that is, each can be a null value) [19]. The query attribute is a string literal (or concatenation of string literals) with one or more triple patterns, usually containing variables. (The query attribute cannot be a bind variable or an expression involving a bind variable).

Our goal is to skip any adapter and just do a direct connection between the client and the server, in our case with thin client from the Eclipse. With this, we achieve better performance, by avoidance of the transformation from one type to another type of query and converting the result data among different types of queries. In our case, the data grid from the execution of the query is converted to an xml file, with concrete defined format. Although this server-side formatting iterations slightly slows down the system, our framework allows us to define the xml format of the response, unlike the others, that return a standard grid-like result.

3.3. Advantages and system usability

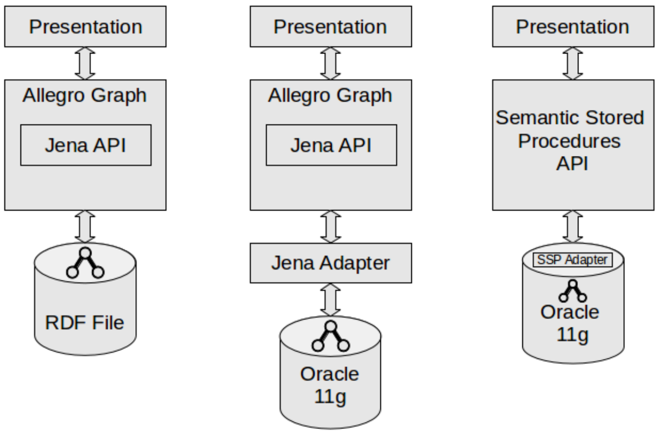


Fig. 4 - Standard Jena from file (left), Jena adapter for Oracle 11g (middle), Semantic Stored Procedures Environment (right)

A simple preview of the available semantic architectures is shown in Fig. 4. Here are presented the two Jena architectures compared to our solution. It is obvious that our solution places the SSP Adapter inside the Oracle 11g database (on the Server side) and that is the key advantage of the SSP solution.

Further, the SSP solution can generate customized format of the response data directly at the server-side, thus minimizing the parsing time on the client-side. The query is stored in the database once and then only called by its name. That makes the system more user-friendly and saves on network traffic and reduces the response time of the query, as will be shown with the measurements bellow.

4. Performance and Comparison with the Jena SPARQL

The thin lines on the graphs from Fig. 5 represent execution time of a single query and the thick lines represent the average execution time, averaged from 50 query executions. We tested with 7 different queries with increasing complexity. Testing was done on a local machine, so there is not network delay influence.

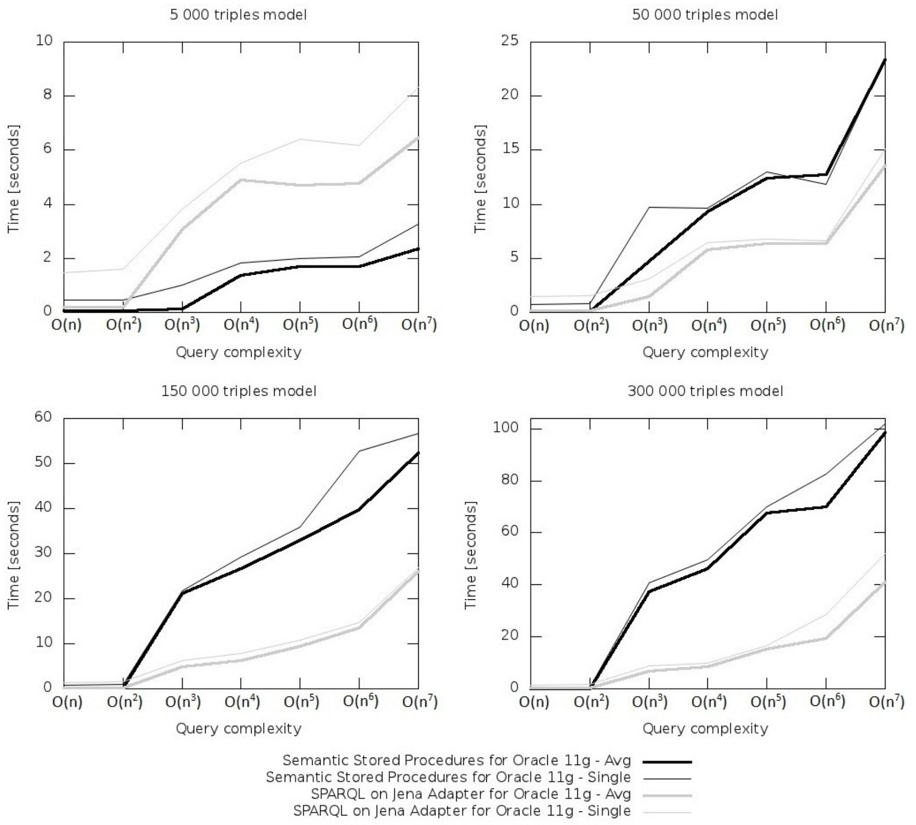


Fig. 5 - The Performance Measurements

The SSP solution (darker lines) is almost twice faster than the one based on the Jena adapter for Oracle 11g, in case of small data sets. Performance decreases with the size of the dataset, especially because the size of the response dataset. That is a consequence of the server-side parsing of the result. The result of the execution of the sem_match query is a multi line (multi row) data set, which must be concatenated and formatted in one big set, to be easy operable and dynamically transformed. When executing sem_match query or storing sem_match query in a procedure, first of all, the requested columns in the “SELECT” of the query are parsed, spaces are removed and the final string is replaced in exact place in the dynamically built xml file. Aliases of the columns in the select statement are not allowed, it handles only column names.

Whatever the case is, first the requested sem_match query is executed and the result data set is treated like CLOB and formatted using Oracle XML Functions [20]. That is a reason why our framework has worse times than SPARQL over Jena.

When the result is not formatted, the SSP solution response time is about a half of the Jena Adapter for Oracle 11g solution, in both of the tests, the single query and the average of many executions of it, shown in Fig. 6. In this measurement, Jena is also called without parsing the result, thus only the effective database retrieval time is measured in the both systems.

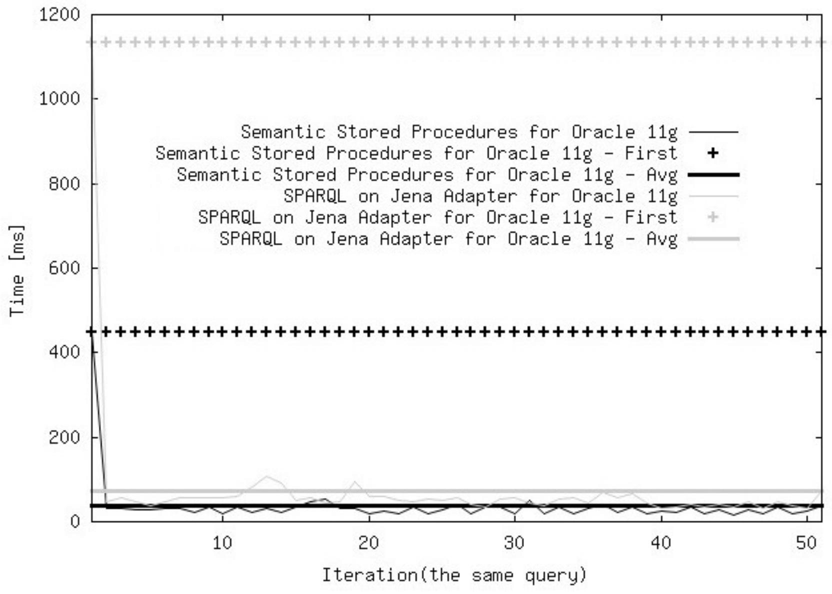


Fig. 6 – Response time without server-side parsing

5. SEM_MATCH vs. SPARQL power

There are two ways to query a semantic data stored in the Oracle Database: the SEM_MATCH-based SQL statements and the SPARQL queries through the Jena Adapter. The queries using each approach are similar in appearance, but there are important behavioral differences [21]. Further details about a comparison between the sem_match and the SPARQL and how the different parts are positioned, when a query is built can be found in [22].

SPARQL queries involving DISTINCT, OPTIONAL, FILTER, UNION, ORDER BY, and LIMIT are converted to a single Oracle SEM_MATCH table function (this is supported in the new Oracle release 11.2) [23]. If a query cannot be converted directly to SEM_MATCH because it uses SPARQL features not supported by SEM_MATCH (for example, CONSTRUCT), the Jena Adapter employs a hybrid approach and tries to execute the largest portion of the query using a single SEM_MATCH function while executing the rest using the Jena ARQ query engine. In this case, the Jena Adapter converts the inner UNION query into a single SEM_MATCH table function, and then passes on the result set to the Jena ARQ query engine for further evaluation [24].

Thus, there is a 1:1 mapping between the SPARQL and the sem_match, so you can do the same things with both of them, except that SPARQL is a bit more human-understandable and more widespread.

6. Conclusion and Future Work

As we mentioned previously, the SSP solution only implements the pattern provided for the SQL stored procedures in scope of the Semantic Web. The need of the stored procedures scenario is already proven, thus the need for a Semantic Stored Procedure is obviously clear. The measurements only confirmed our logical expectations. Thus, the show-case scenario was successfully implemented and the SSP environment is worth to be finalized and spread among the Semantic Web programmers.

The SSP eclipse plug-in can be improved in various aspects. The UI can be redesigned to a more user friendly one. The semantic stored procedure editor will offer syntax highlighting and code completion, so the user can easily construct complex queries. The plug-in will also offer synchronization with the database, so the user will be aware of the current state of the database and can take actions over the database via the plug-in GUI.

The user will be offered to write SPARQL queries instead of SEM_MATCH queries, so it will be much easier to use the plug-in. In order to work with SPARQL queries, a SPARQL to SEM_MATCH query converter will be developed (or used an existing), so every SPARQL query will be translated into one or more SEM_MATCH queries, which will be stored in database as a semantic stored procedure.

The query result is a XML formatted string, thus further extensions with the XSLT can take place and enable the user to provide a XML Schema string as a variable, within the query call and receive an answer formatted according to that XML Schema.

References

1. Strasunskas, D. and Tomassen, S.L. On Variety of Semantic Search Systems and Their Evaluation Methods. Proceedings of International Conference on Information Management and Evaluation, University of Cape Town, South Africa, 25-26 March 2010, Academic Conferences Publishing, pp. 380-387.
2. Pan, J.Z., Thomas, E. and Sleeman, D. (2006) “Ontosearch2: Searching and querying web ontologies”, In Proc. of the IADIS International Conference, pp 211-218.
3. Oracle 11g Database - http://www.oracle.com/technetwork/database/enterprise-edition/overview/index.html
4. RDF, Resource Description Framework, http://www.w3.org/RDF/, 2010
5. Jena Adapter for Oracle Database - http://download.oracle.com/docs/cd/E18283_01/appdev.112/e11828/sem_jena.htm
6. Jena – A semantic web, java framework,
   Official API documentation and examples for Jena libraries, http://jena.sourceforge.net/,2010
7. SPARQL - http://www.w3.org/TR/rdf-sparql-query/
8. Eclipse - plug-in-based editor - http://eclipse.org/
9. Semantic Technologies Center - Oracle http://www.oracle.com/technetwork/database/options/semantic-tech/whatsnew/index-088828.html
10. Oracle Database 11g Semantic Features http://www.oracle.com/technetwork/database/options/semantic-tech/whatsnew/semtech-makes-ed-sm-195114.html
11. Oracle Database 11g Semantics Technical Talk http://www.oracle.com/technetwork/database/options/semantic-tech/whatsnew/oracle-33.pdf?ssSourceSiteId=otnjp
12. SPARUL, SPARQL update, http://www.w3.org/Submission/SPARQL-Update/
13. OWL, Web Ontology Language, http://www.w3.org/TR/owl-features/
14. Sesame, A semantic web, java framework, http://www.openrdf.org/about.jsp
15. API, application programming interface, http://en.wikipedia.org/wiki/API
16. Sesame Adapter for Oracle Database – http://download.oracle.com/docs/cd/E18283_01/appdev.112/e11828/sem_sesame.htm
17. Chuck Murray (2010) “Oracle Database Semantic Technologies Developer's Guide 11g Release 2 (11.2)“ http://download.oracle.com/docs/cd/E18283_01/appdev.112/e11828/sdo_rdf_concepts.htm#CIHHEDAC
18. ORACLE FEATURE OVERVIEW, ORACLE DATABASE SEMANTIC TECHNOLOGIES (2009) http://www.oracle.com/technetwork/database/options/semantic-tech/semtech11gr2-featover-131765.pdf
19. Oracle Semantic Technologies Overview http://download.oracle.com/docs/cd/B28359_01/appdev.111/b28397/sdo_rdf_concepts.htm
20. Generating XML Data from the Database http://download.oracle.com/docs/cd/B19306_01/appdev.102/b14259/xdb13gen.htm
21. SEM_MATCH and Jena Adapter Queries Compared, http://download.oracle.com/docs/cd/E18283_01/appdev.112/e11828/sem_jena.htm
22. Zhe Wu, Matthew Perry, Vladimir Kolovski (2010) “Oracle Database Semantic Technologies: Understanding How to Install, Load, Query and Inference”
23. Compilation of SPARQL queries to a single SEM_MATCH Call, http://www.filibeto.org/sun/lib/nonsun/oracle/11.2.0.1.0/E11882_01/appdev.112/e11828/sem_jena.htm#sthref298
24. Chuck Murray (2010) “Oracle Database Semantic Technologies Developer's Guide 11g Release 2 (11.2)“ http://download.oracle.com/docs/cd/E18283_01/appdev.112/e11828/sem_jena.htm