Using ontologies to build a database to obtain strategic information in decision making

Érica F. Souza[1], **Leandro E. Oliveira**[1], **Ricardo A. Falbo**[2], **N. L. Vijaykumar**[1]

[1]Computação Aplicada – Instituto Nacional de Pesquisas Espaciais (INPE)
São José dos Campos – São Paulo – SP – Brazil

[2]Departamento de Informática – Universidade Federal do Espírito Santo – UFES
Vitória – Espirito Santo – ES – Brazil

{erica.souza, vijay}@lac.inpe.br, leandro.oliveira5@fatec.sp.gov.br,

falbo@inf.ufes.br

Abstract. *The manipulation of ontologies in databases can represent gains in the recovery of strategic information in decision-making process within the software development organizations. The software testing processes are strategic elements to develop projects and to the quality of the final product. Thus, this study investigates strategies to promote data handling of testing processes that are generated from a testing ontology. For this, a knowledge database is structured in a dimensional model for Data Warehouse to support storage and processing of data to obtain strategic information that can facilitate decision making.*

Resumo. *A manipulação de ontologias em bancos de dados pode representar ganhos na recuperação de informações estrátegicas no processo de tomada de decisão dentro das organizações de desenvolvimento de software. Os processos de teste de software são elementos estratégicos para a condução de projetos de desenvolvimento e qualidade do produto final. Diante disso, este trabalho tem como objetivo investigar estratégias que possam promover a manipulação de dados de processos de teste que são gerados a partir de uma ontologia de teste de software. Para isso, estrutura-se uma base de conhecimento em um modelo dimensional de Data Warehouse que apoie o armazenamento e o processamento dos dados para obtenção de informações estratégicas que podem facilitar a tomada de decisão.*

1. Introduction

With the exponential growth of data from several different sources of knowledge within an organization, it becomes necessary to provide automatized support for tasks of acquiring, processing, analyzing and disseminating knowledge. Organizations need to effectively manage the information generated in its production environment to promote the improvement of the processes used to generate knowledge and also support future decisions. Such data can provide important information for decision making, involving the identification and implementation of corrective actions.

One of the characteristics of software engineering projects is to deal with a great deal of information that are generated and manipulated. People involved in the project

face problems, such as: organize in a systematic way the information generated through the software process; reuse the knowledge generated from one project to another; loss of intellectual capital of the organization due to better opportunities; and no knowledge representation [Andrade et al. 2010].

In the area of software development, testing is a critical factor in product quality, and thus there is a greater concern with related research. Studies indicate that the quality of the software product is strongly dependent on the quality of the processes that are part of the project, especially the software testing process. However, finding relevant information (knowledge) in these processes can be a difficult and complex task, and it is related mainly to the lack of semantics associated with the large volume of information. There is a need to represent knowledge, to make it affordable and manageable. In this context, ontologies have been pointed out as an important way for representing knowledge [Rios 2005].

The manipulation of big ontologies with a high number of instances in the form of text files has a number of disadvantages, such as processing and query optimization [Filho et al. 2010]. Because of this, ontologies can be incorporated into a knowledge base to facilitate its management and access. Depending on the structure the database is created, the analysis of large data volumes and mining strategic information can facilitate decision making. Related work is found in [Astrova et al. 2007], [Vysniauskas and Nemuraite 2006], but they propose approaches that transform ontology representation into a relational database, but not in a dimensional model as this enables dealing with large volumes of information.

Given the above context, this paper aims to investigate strategies that can promote the manipulation of data generated from large ontologies. For this, a knowledge base in a dimensional model for Data Warehouse is structured to support the storage and processing of data to obtain strategic information that can facilitate decision making. A software testing ontology is being developed to support this work and instances of this ontology is used as a data source for the structure created. Section 2 briefly discusses ontologies and storage structures. Section 3 presents the proposed structure to store ontology data. Section 4 presents conclusions and future directions to follow.

2. Ontologies and structures for storage

During the last decades, ontologies have been shown useful in the field of Computer Science [Guizzardi 2005]. In a nutshell, an ontology is a formal specification of a shared conceptualization [Gruber 1993], i.e., a description of concepts and relationships that may exist for an agent or an agents community. Representation of a shared conceptualization requires a representation language. There are many representation languages. Some are defined based on the syntax of the eXtensible Markup Language (XML), like Resource Description Framework (RDF), Ontology Interchange Language (OIL) and Web Ontology Language (OWL). There is also graphical language for ontologies; an example is the Graphical Language for Expression Ontologies (LINGO) [Falbo et al. 1998].

Literature provides several tools to store and manipulate content from an ontology in a database and several strategies can be found [Filho et al. 2010]. Most tools use relational databases. Depending on the number of instances of an ontology, it becomes necessary to create structures that support high volumes of data and allow employing

techniques to find relationships among these data. An alternative is the use of Data Warehouse (DW) as the storage structure. It is a large repository of integrated data obtained from several sources for the specific purpose of data analysis [Christian et al. 2010]. A DW may take on different models: Cube and Star. The difference is in the Database Management System (DBMS). When the dimensional model is implemented in a relational database it is implemented as a Star architecture and when implemented in a multidimensional database it is known as Cube. Considering that DW stores a large volume of data, it optimizes and reduces the complexity of consultations, thus decreasing the response time and gaining in performance.

3. Proposed structure for storage of ontology content

The ontology used in this paper is in the context of the Ph.D. thesis of the first author [Souza 2011]. The ontology aims at software testing for Knowledge Management (KM). Test activity is incorporated into a process as it consists of several steps to add quality to the final products [Bastos et al. 2007]. Since the test activity is a process, the improvement can be incorporated with the use of KM and ontologies are identified today as being crucial in the KM in processes improvement.

Given the complexity of the software testing domain, an ontology and its sub-ontologies were created and used. Currently the main ontology created has: Steps, Techniques, Types, Artifacts and Environment. SABiO (Systematic Approach for Building Ontologies) was adopted to develop the software testing ontology [Falbo 2004].

As the ontology is still under development it may suffer some changes. However, though in the initial phase, it is already capable of describing execution phase of the tests, and from this premise that DW will be created to store ontology data. This phase contains data related to dates, for example, date of test case execution, date of defect submission and date of defect correction.

Instances of the software test ontology were extracted from an actual project developed at the (*Technological Institute of Aeronautics - ITA*) - (*Project of Amazon Integration and Cooperation for Modernization of Hydrological Monitoring - ICA-MMH B*) [Cunha 2010]. We used test data generated from Organizational Testing Management Maturity Model (OTM3) testing process [Lamas et al. 2010].

For converting the content of an ontology written in RDF into a DW architecture, we follow the process shown in Figure 1.

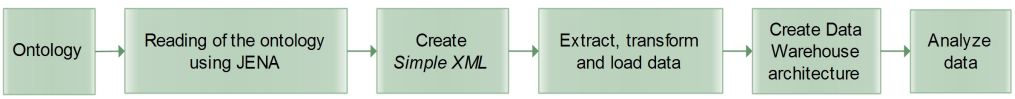


Figure 1. Process Proposed for conversion of an ontology in a DW dimension

To enable reading the ontology in the Pentaho Data Integration tool it was necessary to transform the RDF/OWL in a simple XML and export the data of the instances for the DW model created. For this, we used the Jena framework [Jena 2012] that provides an Application Programming Interface (API) in Java allowing writing, reading and extracting the description of fields, classes and instances from a file in the RDF/OWL in XML format. With the support of Jena framework, a Java class that converts the data into a pure

XML was developed, that is, using only the native tags of XML without the RDF/OWL specific tags. We call this *simple XML*, as shown in step three in Figure 1. The XML with simple pattern is to be read by Pentaho.

From the *simple XML* Pentaho is used to extract, transform and load the data for DW using ETL (Extract Transform Load). The data is loaded in DW table of facts. Star model was chosen to create the DW. The model consists of table *ft_test_execution* which is the table of facts, and five dimensions, namely: (i) three dimensions of time: *lk_execution_time* contains the date of the test execution, *lk_submission_time* refers to the date of a defect submission, and *lk_last_change_time* refers to the date of the last change made in the request to repair a bug; the *lk_team* dimension contains information about the tester, such as the name of the tester and his or her level within the team; and *ii*) the *lk_project* dimension contains information with respect to the project for which the test was performed. Figure 2 shows the Star model created with its respective tables.

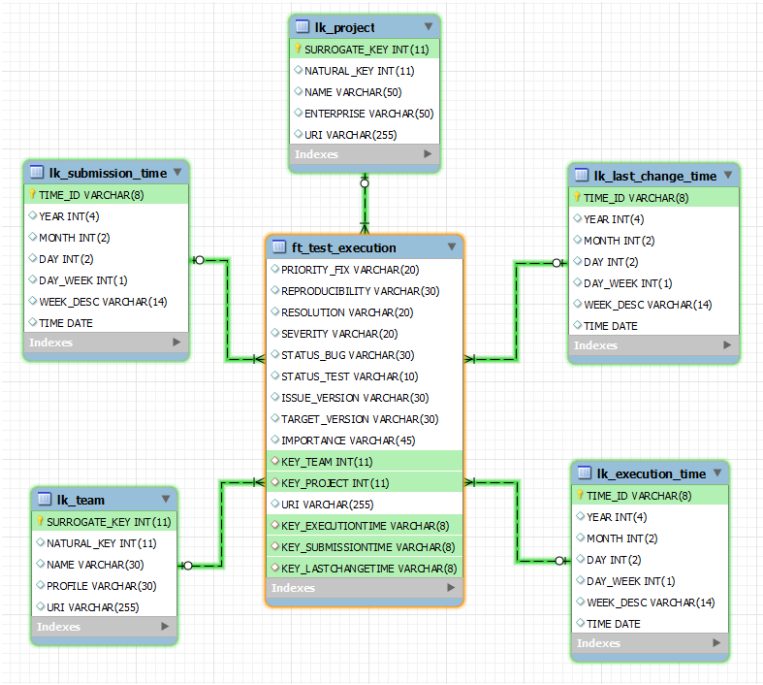


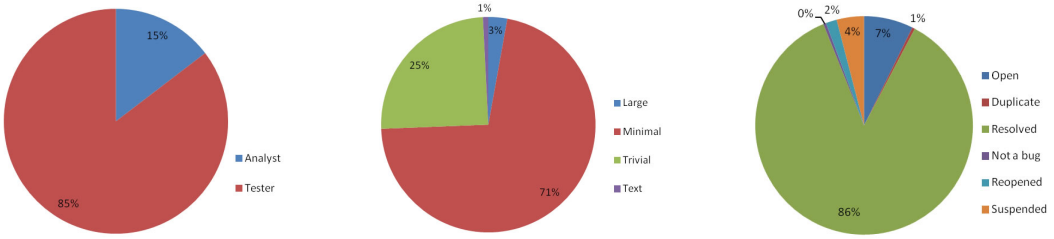
Figure 2. Development of Star Model

In order to see the contribution of the methodology, some questions (that might be important for decision makers) were posed. The idea is whether the table of facts can, in fact, answer such questions in a satisfied manner and useful to decision makers or managers. Therefore, just as an example, the following questions were defined to exercise the table of facts and the different dimensions of the DW:

1. What is the average time between the report of a defect and the test run to check if the bug was repaired? To this question the result obtained was an average of 5.85 days. This information can be useful for the manager or responsible for project to analyze whether the time between a request and the execution of the tests are on schedule.
2. What is the percentage of tests executed per profile of the team? Figure 3(a) shows the results obtained for this question. Most tests were executed by team members

with the position of tester, which is logical since the rest of the tests were executed by analysts and the main function of the analyst is to develop test plans and not to execute them. The result is consistent with the reality of a team of software testing.

3. How severe are the defects reported? Most defects reported have minimal severity and only 3% of the defects are of large severity. This suggests that the process of system development is a reasonable quality control (Figure 3(b)).
4. What is the relation between testing and defect solving? Most of the reported defects have been solved so far. Only 2% of the requests were reopened indicating a recurrence of an already reported defect, and only 1% of duplicate requests that are still under correction (Figure 3(c)).



(a) Number of Executions by profile.

(b) Number of tests by defect severity.

(c) Number of tests by resolution of the defect.

Figure 3. ***Results of preliminary analyzes***

4. Conclusions

This paper presented structuring and storing of ontology content in a DW model. We used a preliminary ontology of software testing process that is under development. The data analyzed refers to the test execution phase. The DW model created is expandable and may include other phases within the process of software testing.

The model facilitates queries and can also provide strategic information that can be used to improve the development process as well as for the process of software testing. The model can be enriched and provide more information that can be used in decision making. Some difficulties were encountered as lack of support for the Jena framework and real case studies with test procedures clearly defined.

Future directions include the automation of data entry in the ontology from other sources, model expansion to store data from other phases of the software testing process and integration with the software development process.

Acknowledgements

FAPESP and CNPq (PIBIC) for the financial support. ITA, Brazilian Water Agency (ANA), Brazilian Agency of Research and Projects Financing (FINEP) and the Casimiro Montenegro Filho Foundation (FCMF) for providing the data of Project FINEP 5206/06 for this work.

References

Andrade, M. T. T., Ferreira, C. V., and Pereira, H. B. B. (2010). Uma ontologia para a gestão do conhecimento no processo de desenvolvimento de produto. *Gestão e Produção*, 17:537–551. http://dx.doi.org/10.1590/S0104-530X2010000300008. Access in: Ago 2012.

Astrova, I., Korda, N., and Kalja, A. (2007). Rule-based transformation of sql relational databases to owl ontologies. *In: Proceedings of the 2nd International Conference on Metadata & Semantics Research*.

Bastos, A., Rios, E., Cristalli, R., and Moreira, T. (2007). *Base de conhecimento em testes de software*. Martins Editora Livraria, São Paulo, 2 edition.

Christian, S. J., Pedersen, T. B., and Thomsen, C. (2010). *Multidimensional Databases and Data Warehousing*, volume 2. Morgan and Claypool, 1 edition.

Cunha, A. M. (2010). Relatório Técnico do 5° Semestre do Projeto FINEP 5206/06. Technical report, São José dos Campos.

Falbo, R. A. (2004). Experiences in using a method for building domain ontologies. *In: International Workshop on Ontology in Action*, pages 474–477. Banff, Canada.

Falbo, R. A., Menezes, C., and Rocha, A. (1998). A systematic approach for building ontologies. *In: In Proceedings of the 6th Ibero-American Conference on AI, IBERAMIA98*. Lisbon, Portugal.

Filho, S. N. V., Moura, A. M. C., and Cavalcanti, M. C. R. (2010). Armazenamento e manipulação de ontologias utilizando sistemas gerenciadores de banco de dados. Technical report, Instituto Militar de Engenharia (IME), Rio de Janeiro.

Gruber, T. R. (1993). Toward principles for the design of ontologies used for knowledge sharing. *In: Formal Ontology in Conceptual Analysis and Knowledge Representation*. Padova, Italy.

Guizzardi, G. (2005). Ontological foundations for structural conceptual models. *Telematica Institute Fundamental Research Series, The Netherlands*. ISBN 90-75176-81-3.

Jena (2012). Apache Software Foundation - Jena. http://jena.apache.org/documentation. Access in: Aug. 2012.

Lamas, E., Souza, E. F., Nascimento, M. R., Dias, L. A. V., and Silveira, F. F. (2010). Organizational testing management maturity model for a software product line. *In: Seventh International Conference on Information Technology, ITNG'2010*, pages 1026–1031. Las Vegas, Nevada, USA.

Rios, J. A. (2005). Ontologias: alternativa para a representação do conhecimento explícito organizacional. *In: Proceedings CINFORM - Encontro Nacional de Ciência da Informação VI*. Salvador, Bahia.

Souza, E. F. (2011). Estratégias de reuso para melhoria de processo de teste de software baseado em ontologias. Technical report, INPE, São José dos Campos/SP. http://urlib.net/8JMKD3MGP7W/3BFFA9H. Access in: June 2012.

Vysniauskas, E. and Nemuraite, L. (2006). Transforming ontology representation from owl to relational database. *In: Information Technology and Control*.