A SOFTWARE ARCHITECTURE FOR KNOWLEDGE DISCOVERY IN DATABASE

Maria Madalena Dias, Lúcio Gerônimo Valentim and José Rafael Carvalho
Departamento de Informática, Universidade Estadual de Maringá, Av. Colombo 5790, 870200-900 Maringá-PR, Brazil

Keywords: Reference Architecture, Software Architecture, KDD system, Data Warehouse, and Data Mining.

Abstract: Currently, most of the companies have computational systems to provide support to the operational routines. Technologies of data warehousing and data mining have appeared to solve the problem of search for necessary and trustworthy information for decision-making. However, the existing solutions do not enclose those two technologies; some of them are directed to the construction of a data warehouse and others to the application of data mining techniques. This paper presents a software architecture that defines the components necessary for implementation of knowledge discovery systems in database. Some standards of best practices in projects and in development were used since the definition until the implementation.

1 INTRODUCTION

With the evolution of computer science and its increasing use in the diverse areas of knowledge, the information systems are becoming bigger and more complex. The faster development of hardware at lower prices has been changing the methodologies used for software development. There was, also, an alteration in the organization of the world-wide markets, which turned the information into a precious item for a nation.

In this scene of increasing technological evolution, the search for knowledge became a necessity of governmental institutions and private sectors. However, there is a certain difficulty in the acquisition of information and knowledge since the volume of data accumulated by institutions, during much time, is enormous, and these data are not integrated.

Researches have been developed in the area concerning search of information and knowledge in great databases, involving, mainly, Data Warehousing and Data Mining (DM).

Data Warehousing technology defines activities for search of great volume of data in heterogeneous operational environments, collecting and organizing the data in a more homogeneous and optimized environment for queries. DM is the second stage in the process of knowledge discovery in database (KDD). In the first stage (pre-processing), a DW can be constructed. In this case, it is necessary to develop software to support this process.

In a software design, the definition of architecture is an essential activity (Jacobson et al., 1999). So, this paper presents a software architecture specified for KDD systems. In order to facilitate the definition of such architecture, it was necessary to construct a reference model and reference architecture. The reference model represents the significant relationships between the entities of a determined environment. The reference architecture defines the common infrastructure for the systems and the interfaces of the main structural components that will be enclosed in those systems.

In the next section, some related works are presented. Following up in Sections 2, 3 and 4, respectively, the reference model, the reference architecture and software architecture for KDD systems are described. In Section 5, it is shown the conclusions about the research presented in this paper.

2 RELATED WORKS

As related works, the following ones can be listed: Oracle Warehouse Builder (Oracle, 2005); I-MIN (Gupta et al., 2004); GridMiner (Brezany et al., 2003); KDB2000 (Appice et al., 2006).

Madalena Dias M., Gerônimo Valentim L. and Rafael Carvalho J. (2008).
A SOFTWARE ARCHITECTURE FOR KNOWLEDGE DISCOVERY IN DATABASE.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - DISI*, pages 433-436
DOI: 10.5220/0001686304330436

Oracle Warehouse Builder is a tool that is part of the set of solutions for database of Oracle Company. That environment allows things such as, the access to databases, definition of transformations, creation of new databases and multidimensional databases operation.

Gupta et al. (2004) present an architecture for knowledge discovery and management called I-MIN. For the execution of activities of KDD process a language is used. That language is analyzed by a specific compiler present in the proposed architecture. All the architecture is divided into small modules.

GridMiner tool was constructed on a framework developed by the same group. That research team includes works about parallel computation in the development of systems for knowledge discovery. KDB2000 integrates, in only one tool, the database access, the preprocessing and transformation of data, techniques, algorithms of data mining and tools for visualization. However, that tool does not provide support for the repetition of the same process executed by the user.

3 REFERENCE MODEL

The reference model proposed in the present study enables the integration of the necessary functions for carrying out the construction of a DW, of OLAP tools (On-line Analytical Processing) and of DM in only one system. Such integration provides a high degree of reuse of the system components and, consequently, enables its construction, if compared with proposals that isolate the construction of a DW from other parts of KDD systems.

The functions of a KDD process are defined in the reference model in modules: *source definition*, *ET processing*, *start definition*, *load processing*, *DW definition, OLAP, DM, results definition, view tools*.

OLAP and DM modules are essential for the architecture proposed in the present study. They integrate OLAP and DM processes in just one environment, reusing operations, such as extraction, transformation and load, and sharing data repositories.

The reference model defines three different data sources to be used in OLAP or DM processing: Source bases, Staging Area and Data Warehouse. Moreover, beyond those databases, the model presents a storage space for the results obtained by using OLAP or DM.

The reference model provided the base for both definitions, the definition of the reference architecture and the software architecture, as well.

4 THE REFERENCE ARCHITECTURE

The reference architecture developed attends requirements of KDD system. For the definition of reference architecture it is necessary to choose and adopt an architectural style. Among the architectural styles presented by Shaw and Garlan (1996), the 'in layer' style offers abstraction levels that allow dividing a complex problem into a sequence of incremental steps and permits reuse. Such characteristics turn this type of style adequate to complex and interactive systems, as it is the case of KDD system.

The architecture is divided in four layers (Figure 1). Managers were defined for each layer in order to control both, the moment of creation of objects that they represent and the access to the layer components. The components of each layer have access to its manager and through the access reference other managers, thus making possible the communication with the components located on other layers. Thus, the communication between the components occurs only through the manager responsible for the layer they belong to.

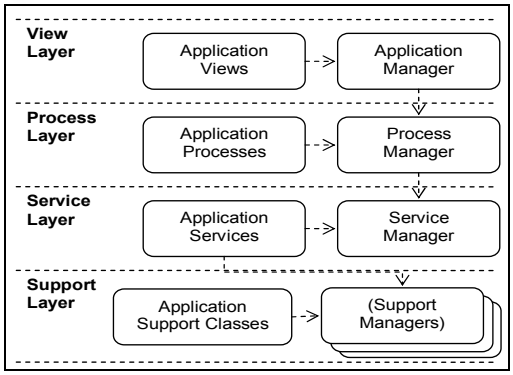

Figure 1: Layers that compose the reference architecture.

The **view layer** is responsible for accommodating the components that show the graphical interfaces (screens) and control the interaction between the user and the system.

The **process layer** has a manager that knows all the processes instanced by users. Thus, to access any process, the superior layer must ask the manager the instance of the process desired. Two layer

characteristics must be pointed out: 1) the processes are authenticated by the layer and are available only to the users with access rights for its; 2) one instance of the process is created for each user.

The **service layer** also has a manager that knows all the services that are instanced by the architecture. There is an interface that allows the upper layer to execute a service. That layer is not authenticated and all the users share the same instance of a service.

The **support layer** was defined to organize the system functions, which give support to the upper layers.

For the documentation regarding the reference architecture, it was made an adaptation of methodology presented by Merson (2006). Such methodology defines the documentation of software architecture in terms of views. Each view points out either static or dynamic aspects of the architecture. But, not all the views are necessary for all the projects.

To illustrate the developed reference architecture, the view of modules and its relationship are shown (Figure 2). It is important to point out in the figure that the **view module** can be extended to other packages that implement diverse visualization technologies, turning the business processes completely independent from interface technologies.

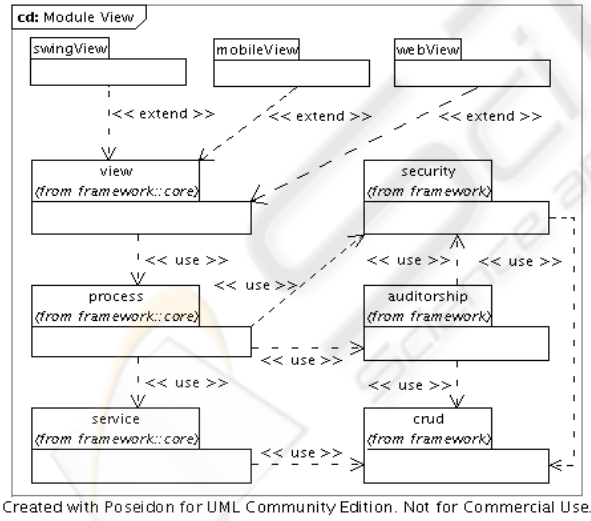

Figure 2: View of modules that shows the packages of reference architecture.

The **process module** is authenticated, due to that it uses the *security* and *auditorship* modules. The information about the process executed is registered in the auditorship. It depends on the *service* module, because the processes use the business services during the activities.

In the **service module** the services are executed inside of a transactional environment. The services can access and manipulate metadata (business entity/ persistent data) by using the *crud* module.

In the **security module** the security entities, as users, groups and rights are included. It has services and processes for user authentication, verification and definition of rights to access.

On the other hand, the **auditorship module** provides services to register the operations executed in processes and entities.

The **crud module** comprises a manager of entities, services and processes for the persistence of objects. However, the main function of that module is to provide an abstraction of business entity and data persistence. Such abstraction consists of joining the object instances with the metadata structures that contains additional information about the entity. Thus, when an entity is recovered from the object repository, it comes with its basic data, and also, with a set of additional information useful for validation and exhibition of the entity data. The **crud module** is an advanced mechanism to persist and control the internal data entities of KDD system.

5 SOFTWARE ARCHITECTURE

The software architecture is defined extending the functions found in the reference architecture, bearing in mind that, in the reference architecture some basic components, which serve as integration points between external components and the internal architecture, as well as its components have been defined. Following the reference architecture, each module in software architecture defines its own views, processes, services and support components.

The software architecture is a specific architecture for KDD domain (Figure 3). As the reference architecture solves the majority of the structural problems, the definition of software architecture is facilitated and its components are abstracted from the concepts and structures defined by the reference architecture.

A diagram, regarding use cases, was constructed to assist both, the learning of functional requirements and the conception of software architecture. The use cases are: *Source Definition, ET Definition, Run ET, Staging Definition, Load Definition, Run Load, DW Definition, Run OLAP, Run DM, Store Results, and View Results*. More details can be found in (Valentin, 2006).

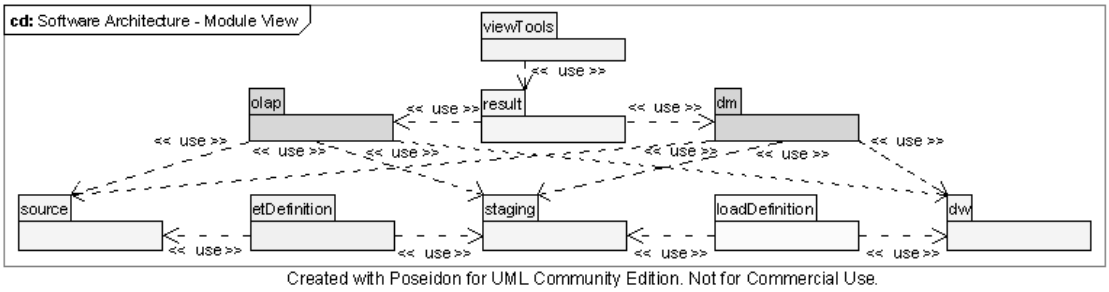

Figure 3: View of Modules of the Software Architecture of KDD system.

5 CONCLUSIONS

Following the architecture construction standard presented by Bass et al. (2003), it was defined a reference model, and a reference architecture was developed, in order to propose a software architecture for KDD systems.

The software architecture proposed was based on the reference model, and on use cases identified for a KDD system, thus being validated through a prototype, being its development facilitated by reference architecture.

The documentation using views, defined by Merson (2006), facilitate the learning and allow presenting both, the architecture general organization, as well as the internal functions.

As it could be seen, the definition of the software architecture components was based on concepts defined in reference architecture, having as main focus the application domain. Questions related to the communication between components, security and other matters remain in the definition of reference architecture, thus making possible a better definition of both, the application domain and the system functions in the software architecture.

Finally, the reference model defined represents the main functions found in the construction of a DW, in OLAP tools and data mining, integrated in a unique system. Thus, the reuse and integration of components make possible the cheaper construction of KDD systems.

The proposal architecture searched to congregate the main characteristics of some related works to the area of knowledge discovery in database. A comparison between the architecture proposed and related works was realized using the following characteristics: 1) descriptive language; 2) interactive activities; 3) software architecture reusable; 4) sources, targets and processes metadata, 5) DW activities, and 6) DM activities. The architecture proposed unsatisfied only the characteristic 1, while the others architectures unsatisfied two or more characteristics.

REFERENCES

Appice, A., Ceci, M., Malerba, D., 2007. *KDB2000: An integrated knowledge discovery tool*. Dipartimento di Informatica,University of Bari, Italy. Retrieved November 30, 2007, from http: //www.di.uniba.it/~malerba/software/kdb2000

Bass, L., Clements, P., Kazman, R., 2003. *Software Architecture in Practice. Second Edition.* Boston, MA: Addison-Wesley, 2003.

Brezany, P., Hofer, J., Tjoa, A., Wöhrer, A., 2003. Gridminer: An Infrastructure for Data Mining on Computational Grids. In: *APAC'03 Conference*, Gold Coast, Australia.

Gupta, S. K.; Bhatnagar, V.; Wasan, S.K., 2004. Architecture for knowledge discovery and knowledge management. *Knowledge and Information Systems*.

Jacobson, I.; Booch, G.; Rumbaugh, J., 1999. **The unified software development process**. 2.ed. Canadá: Addison-Wesley, 463p., (Object Technology Series).

Merson, P., 2007. *How to Represent the Architecture of Your Enterprise Application Using UML 2.0 and More*. Retrieved November 30, 2007, from http://developers.sun.com/learning/javaoneonline/2006/coreenterprise/TS-4619.pdf.

Oracle Warehouse Builder Documentation, 2007. Retrieved November 30, 2007, from http: //www.oracle.com/technology/documentation/warehouse.html.

Shaw, M.; Garlan, D., 1996. *Software Architecture – Perspectives on an Emerging Discipline*. Prentice Hall, Upper Saddle River, New Jersey.

Valentin, L. G., 2006. Uma arquitetura de software para sistemas de descoberta de conhecimento m banco de dados**.** Programa de Pós-Graduação em Ciência da Computação. *Dissertação de Mestrado*. Departamento de Informática. Universidade Estadual de Maringá.