World Applied Sciences Journal 31 (5): 979-987, 2014
ISSN 1818-4952

DOI: 10.5829/idosi.wasj.2014.31.05.268

An Agent Based Etl System: Towards an Automatic Code Generation

Abderrahmane Sadiq, Abdelaziz El Fazziki and Mohamed Sadgal

Computer Systems Engineering Laboratory, University of Cadi Ayyad Marrakesh, Morocco

Abstract: The key of an effective management of enterprises is applying intelligent software tools to define decision analysis process, from data acquisition to recommended action, including an adaptive feedback. Decision support systems (DSS) have multiple data sources that contain valuable information that must be integrated and processed to allow the key business processes to have adaptive business intelligence. Conceptual modeling of decision support systems (DSS) should take into account some relevant aspects, such as the integration of external data, the behavior of the system environment, the available data sources and the emerging paradigm related to the intelligent systems. The aim of this work is to propose an approach based on model driven architectures, multi-agent systems (MAS) for DSS development. We are particularly interested in this work in developing a multi-agent based extraction, transformation and loading process (ETL), retrieving, extracting and integrating external data into Data Warehouses (DW) and generating automatically the DW code.

Key words: Data Warehouse · Model Driven Architecture (MDA) · Code generation · Multi-Agent System · Decision Support Systems · External data integration

INTRODUCTION

Actually, business intelligence adopts two main pillars, the strategic foresight and decision-making, widespread in the business area. These two pillars can be integrated in a global manner, to, understand, predict, control, simulate or seize new opportunities based on relevant information – internal or external, structured or unstructured data [1]. The most common functions of business intelligence technologies are: editing reports, online analytical processing, data mining, business performance management, text-mining and predictive analytics and knowledge discovering.

Many studies propose the use of data warehouses based systems supplied with operational data, for decision making. These solutions are focused on the automation and the adaptation of the data ETL process according to the existing systems in an enterprise [2]. But these systems present enough problems according to the agility and adaptability to face the continuing change of the enterprise environment [1]. Therefore, it is clear that most of the DSS's depend on the building of the DW that involves the ETL process [3]. Thus, the success of DW-based system correlates with the performance of the ETL processes.

The objective of this article is to propose a development process based on model driven architecture (MDA) [4], MAS and the incorporation of external data, while taking into account the evolution of the enterprise environment.

The remainder of this paper is structured as follows. Section 2 will give a brief literature review of Multi-Agent Systems, the MDA and DSS. This will be followed by an introduction of the basic concepts and an overview of the proposed approach. After presenting the system architecture, the agent system structuring will be presented in section 4. Section 5 is devoted to the system modeling and implementing. The DW design and code generation process will be given in section 6. Finally, Section 7 presents a conclusion describing the contributions of this work and its perspectives.

Related Work: Many studies have proposed the use of data warehouses fed by transactional databases for

Corresponding Author: Abderrahmane Sadiq, Computer Systems Engineering Laboratory, University of Cadi Ayyad Marrakesh, Morocco. Tel: +212(0)600553367.

decision support systems. These works focused on automating and adapting the ETL process according to the existing systems, but these systems do not take into account the problems of adaptation to the continuing evolution of the enterprise environment, which musts answer the question of decision support tool's flexibility [5-6].

The outstanding advances in intelligent systems have made a powerful impact on most of the information system development and business intelligence [5]. Actually, many improvements have been notified in the DSS field by integrating the paradigm of artificial intelligence techniques, such as data mining tools and multi-agent systems. The emergent DSSs are now intelligent decision support systems (IDSS) [7].

Several kinds of IDSS exist, some of them replace the model base management system (MBMS) with the Expert Systems or other intelligent decision making tool, those where other functionalities are added to enhance MBMS to make it intelligent and improved user interfaces can be achieved using other parts of intelligent tools, including natural language processing or similar techniques [8].

McGregor [9] uses an Agent-based DSS that have the intelligence component attached to the model base as opposed to replacing it. In this work, an IDSS is DSS that has intelligent components, which either replaces or enhances the different DSS component models.

Integrating Decision Artificial Intelligence (DAI) technologies in DSS is an endeavor to develop a DSS that takes into account human qualities, such as reasoning, intuition, negotiation and natural common behavior. The use of intelligent DSS is planned to improve the ability of decision-actors to better perform their tasks [7]. Some works propose the use of Multi-agent systems in the DW ETL Process, like [10] which proposed the integration of several information resources for Decision Support in Enterprises.

Other works have addressed the issue of automatic DWs design based on the model driven architecture (MDA) and automatic transformations between models such as [11] that present a rigorous framework for modeling relationships between the source and the target models and defining mappings between them. For example, in [12] the authors describe an automatic generation of a multidimensional schema from UML class diagram.

The Proposed Approach: The proposed approach is mainly based on the concepts of model-driven architecture, multi-agent systems and automated model transformations using the Atlas Transformation Languages (ATL) [13].

The Basic Concepts

Model Driven Architecture: Model Driven Architecture, is a software development approach, proposed and supported by the OMG [4]. This is a particular variant of the model driven engineering (MDE). In the MDA principle, the process of software development is driven by hierarchical models in four levels according to a set of meta-models. The key feature of the MDA is the concept of model mapping. A model transformation is a set of transformation rules and techniques of exploitation on a source model to attain a target model. According to OMG, there are three types of model: the CIM (Computational Independent Model), the PIM (Platform Independent Model) and PSM (Platform Specific Model). Figure 1 shows a conceptual view of the MDA development process.

The MDA software development life cycle is a four-step process [4]:

- Create a CIM; CIM of a system describes domain activities, information and requirements with the help of various UML Diagrams.
- Develop a PIM; the PIM does not include any details of particular platform; it captures the domain's key concepts technically.
- Transformation of PIM into PSM.
- Generate the code from PSM; and deploy the system in a specific environment.

ATLAS Transformation Language (ATL): Following the publication of MDA in 2001 by the Object Management Group (OMG), many software companies and research laboratories have developed many tools to claim this architecture. Among these, the ENRIA laboratory of Nantes has developed the ATL (ATLAS Transformation Language) model transformation language in the context of the ATLAS project [13].

Multi-Agent Systems: The MAS consists of an autonomous agent's collection that can define their own goals and actions and can interact and collaborate among each other. In a MAS environment, agents work collectively to solve specific problems. It provides an effective platform for coordination and cooperation among multiple functional units in an organization [14].

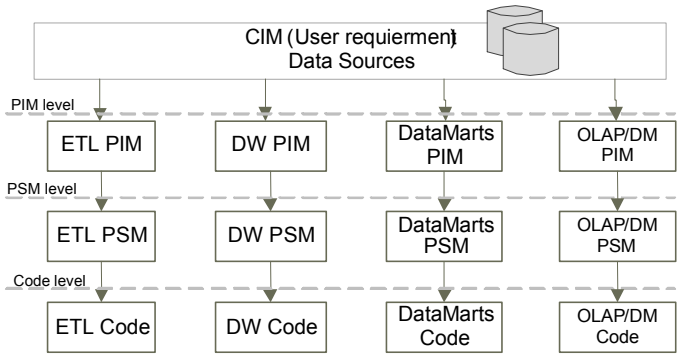

Fig. 1: The MDA oriented DW development framework

While there is no universally agreed definition of an agent, the following one is the most widely accepted: "an agent is a computer system that is situated in some environment and that is capable of autonomous actions in this environment in order to meet its design" [15].

There exist a lot of similarities between the agent in the MAS paradigm and the human actor in business organizations, in terms of their characteristics and coordination. This lead us to a conceptualization where intelligent agents are necessary to represent human behavior in organizations [14].

The research stream on DSS makes the MAS paradigm a very appropriate for integrative decision support within business information systems. This, lead to develop systems, where intelligent agents are used to represent human and organizational dimensions and will react to the DSS environment [15].

The proposed approach considers the decision-making process as an entity that is able of acting, reacting, collaborating and especially establishing decisions. In order to do this, we propose the use of an MDA agent oriented architecture. In this work, first a CIM must be developed in order to acquire user needs. Secondly, each PIM is designed by using the appropriate model. These PIMs are conceptual models of each part of the DW without taking into account any specific implementing technology. PIMs are manually or automatically generated from CIM [4]. Then, the resulting PIM can be automatically transformed into PSMs. These transformations are formally implemented by using the ATL language [13]. Finally the necessary code to implement the PSM model is generated. The main advantage is that once every required PIM has been developed, we automatically can obtain the PSM and code by using a set of automatic transformation rules.

An MDA Based DSS Architecture: The approach consists of applying the MDA principle which is four levels architecture:

- Defining the, CIM, PIM and PSM models;
- Defining and implementing transformation rules;
- Generating the code.

Figure 1 presents the proposed architecture.

The Multi Agent Decision Support System Framework: There are three fundamental components of a DSS [5], ETL component; OLAP (Online Analytical Processing) component and user interface component. In order to provide an adequate solution in terms of robustness, smartness and agility, we use a Multi-Agent framework to represent Decision Support System. The proposed Multi-Agent framework is issued from the previous DSS components. The derived MADSS is composed of three subsystems:

- Multi-agent ETL Subsystem (MAETLSS)
- Multi-agent OLAP Subsystem (MAOLAPSS)
- Multi-agent User Interface Subsystem (MAUISS)

The following figure illustrates this structuring.

MAETLSS: One of the main requirements of DSS is the need for aggregated information from various, internal (i.e. Relational databases) and external structured or semi-structured data sources, these data must be retrieved, normalized and automatically integrated afterwards in a DW [16-2] (Figure 2).

MAOLAPSS: The role of the OLAP subsystem (OSS) is analogous to that of Database management systems. It includes the basic models that provide DSS with analytical capabilities. The purpose of an OLAP subsystem is to convert the data into information by applying analysis. Since many problems that the user of a DSS will cope with may be unstructured data, the OSS should also be able of assisting the user in model building.

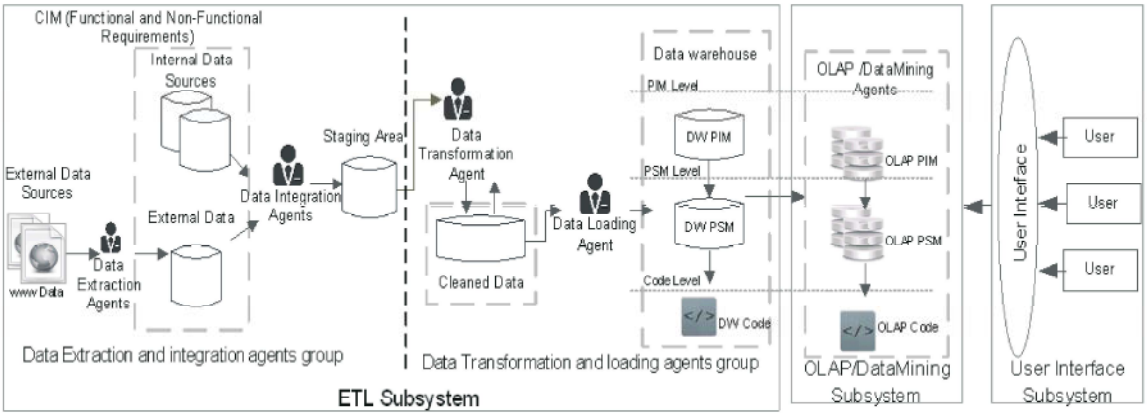

Fig. 2: The multi-agent DSS architecture

MAUISS: Have in charge interactions between a user and the DSS. As the DSS users are often not computer trained managers, this requires being prepared with insightful and easy use of interfaces.

These interfaces assist in model building as well as interaction with the model, such as gaining insight and recommendations from it. The primary responsibility of a user interface is to enhance the ability of the system user to use and entirely benefit from the DSS.

In this paper we focus on the MAETLSS description and modeling.

Mads Subsystems Description

The ETL Subsystem: This subsystem is structured into two agent groups:

- Extraction and integration agents group
- Transformation and loading agents group.

Extraction and Integration Agents Group: These agents are used in order to retrieve data mainly from the World Wide Web and internal databases; this includes several agent roles as follows:

- The source identifier agent: searches any new pertinent data in the web.
- The extractor agent: sets connection with the data source and extracts data.
- The Integrator agent: installs the extracted data in the data staging area (DSA).
- Data sources monitor agent: identify if there is any probable change in the existing data.

The extracted data are loaded into the data warehouse staging area. Internal data are directly extracted from the source databases, transformed and purified (cleaning) in order to load them into the target DW.

Figure 3 presents the proposed data extraction and integration agents group.

Transformation and Loading Agents Group: Responsible of data validation, data accuracy, data type conversion and business rule application. It is the most complicated ETL element. This group is composed of the following agents:

- Data Filtering and Accuracy agent ensures the consistency of data, verifies that fields contain appropriate values and guarantee that all values for a specified field are stored in the same way in the target DW regardless of how they were stored in the data source.
- The data validation agent verifies that all rows in the fact table match rows in dimension tables to ensure data integrity.
- The Integrator agent that installs the transformed data in the cleaned data area (CDA).
- The loading agent is responsible for the continuous loading of the prepared data into the target DW. This agent must ensure the efficiency and performance to minimize the data warehouse offline time during update operations.

Maetlss Modeling and Implementing

The Development Process: The development process is an iterative process allowing incremental development and provides the rollback possibility to a previous phase due to the use of the MDA [5]. This development framework is based on the following criteria:

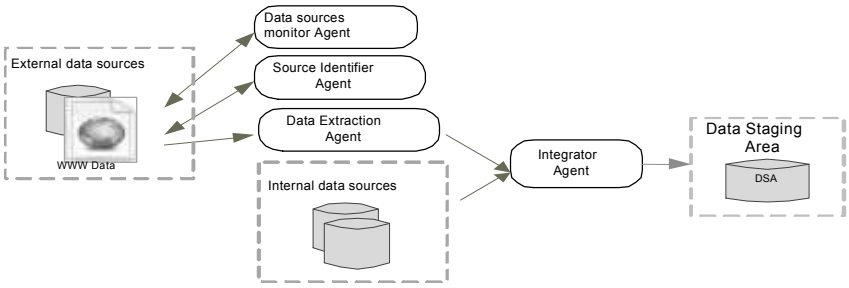

Fig. 3: The multi-agent data extraction and integration group

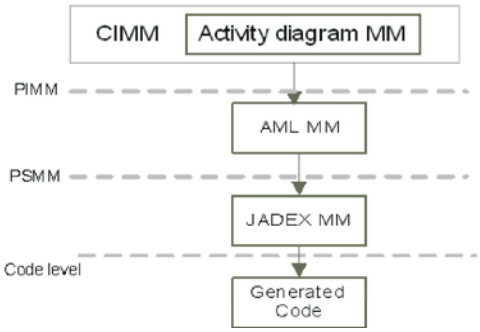

Fig. 4: Development approach

- Concepts and specific properties of the domain,
- Concepts and specific properties of the MAETLSS;
- The transformation rules between domain concepts and agent concepts;
- Concepts and properties which are based on the specific platform;
- The transformation rules between the domain concepts, the activities concepts and agent concepts to the target platform concepts. Figure 4 shows the different phases of the process.

In our approach, we propose several meta-models. We start by the analysis meta-model described by UML use cases and Activity diagrams (AD), the agent meta- model based on the AML concepts and finally the Jadex meta-model based on Jadex concepts [17].

CIM Level: Before starting ADs modeling, firstly, it is necessary to define the business requirements for the review of each domain. To do this, we use the use case diagram (Figure 5).

Use Case to ADs Mapping: Frequently, analysis of business processes will result in a set of activity diagrams which illustrate how work is performed within an organization. This may include dividing the diagram up into logical streams using swim lanes. After this initial analysis work using the UML use case diagrams, it can be convenient to group activities into logical Use Cases. Mapping Use Cases onto Activity Diagrams provides a good means of visualizing the overlay of system behavior onto business process. The result AD is shown in the Figure 6.

Table 1: UML Activity diagram and AML mapping table

UML Activity diagram	AML
Partition	Group of agentType
Partition	AgentType
Node	Action (Extention)
Structured Activity Node	Plan
Initial Node	Input
Final Node	DecidableGoal
Accept Time Action	--
Send Event Node	Communication Message Payload
Receive Event Node	InternalTask

PIM Level:Agents Modeling: For the agent modeling, we use the AML language (Agent Modeling Language) as a PIM. The PIM is directly defined from the system AD's. Each agent is derived from the source activity diagram partitions [17-18]. These agents cover all the system activities.

UML Activity Diagram to AML Mapping: Generally the UML AD can be used as a basis for the development of agents' behavior. This, by identifying for each agent: its plans, goals, interactions, statements and resources. Normally, the derivation of the agents' behavior from an AD is highly dependent on the type of agents and how their behaviors are implemented in a specific platform.

This derivation is essentially deduced from a set of rules based on the concepts of the agent meta- model and AD meta-model.It can be performed automatically or manually. Generally there are two types of pools: simple and structured (composed with lanes). Each agent behavior is derived from a simple pool or from a lane of a structured pool. These behaviors cover all the agent activities.

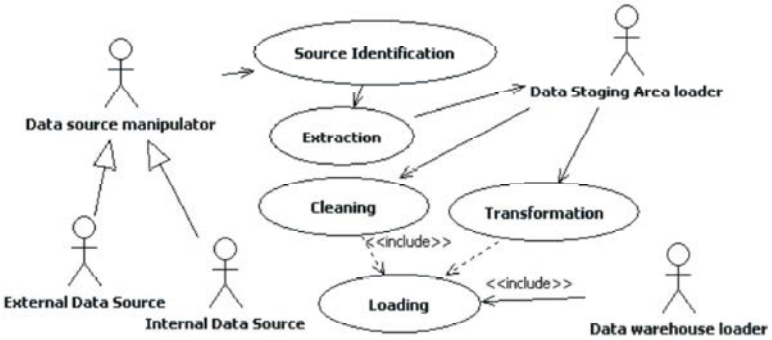

Fig. 5: The ETL use case diagram

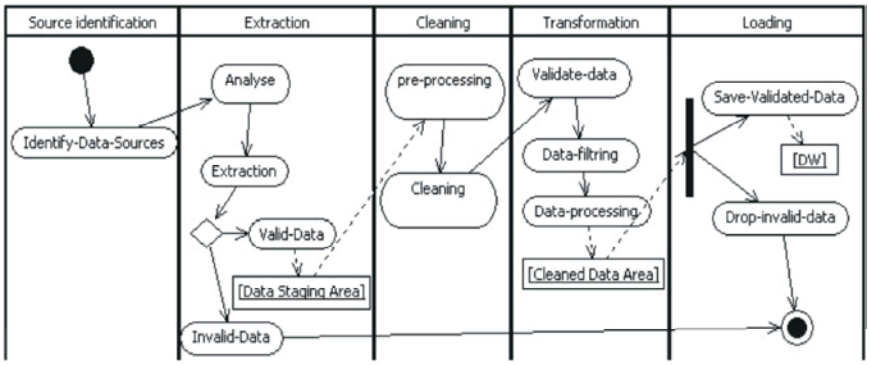

Fig. 6: ETL process activity diagram

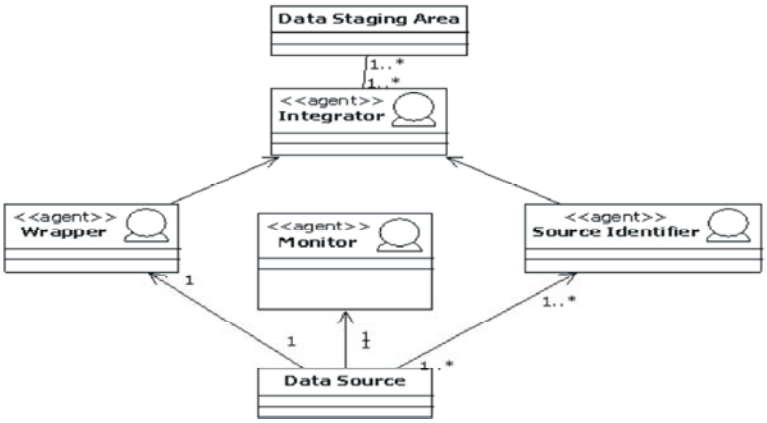

Fig. 7: Class diagram for data extraction and integration

In addition, use cases and ADs are used to identify agents with their types and their interactions. The mapping that we adopt is specified by a set of transformation rules, ensuring the gap from ADs concepts to agent concepts. These rules are not exhaustive and are not detailed. The aim of this phase is the specification of a mapping from the UML activity diagram to the agent behaviors. The main transformation rules are presented in the following table [17]:

The result AML class and sequence diagrams for data extraction and integration agents are given in the Figures 7 and 8:

PSM Level: Agents Implementing: For the PSM development we use automatic transformations between the AML models and Jadex platform in order to generate the related multi-agent system code. A Jadex application consists of a set of XML-based configuration files and Java-based behaviors.

AML to Jadex Mapping: In this section we specify a mapping between AML and Jadex platform. This corresponds to the relation between mental package of AML and Agent Definition Files (ADF) in jadex [19]. As it can be seen in the overview of Jadex and AML

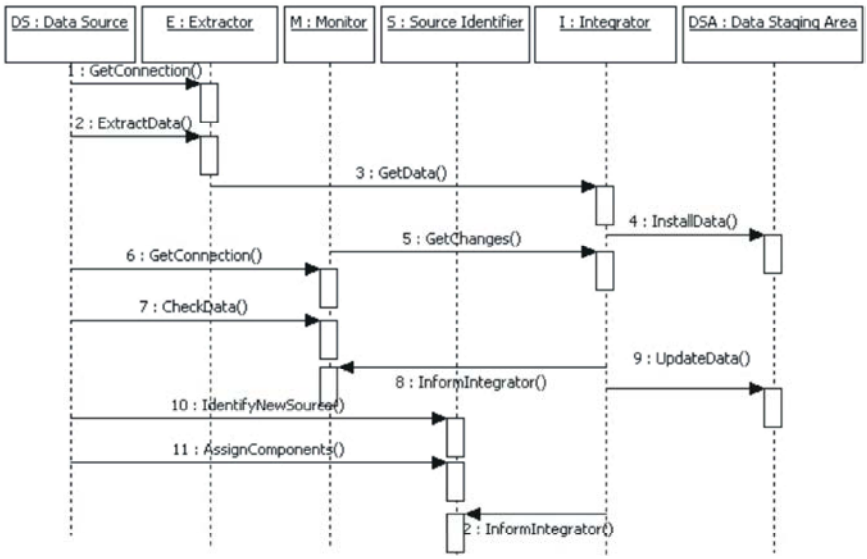

Fig. 8: Sequence diagram for data extraction and integration

Table 2: AML to Jadex mapping

AML Elements	JADEX Elements
AgentType	Name of the ADF file
Belief of AgentType	Belief
DecidableGoal of AgentType	Achievegoal
Plan of AgentType	Plan
CommunicationMessagePayl	Action (stereotype) of AgentType
StartEventMessage	MessageEvent
First plan of AgentType	InitialPlan sub-element of the configuration
Action of AgentType	Function implemented in the plan Java class

frameworks, the semantics of their elements are close to each other, since both of them follow the BDI (belief–desire–intention) architecture [16]. Table 2 shows an overview of the AML and Jadex Corresponding elements.

Jadex plans are other components used to develop agents within the system and implemented in Java programming languages, using libraries provided in Jadex. On the other hand Jadex has only a very general specification of these files, thus no additional strict implementation constraints are defined. Therefore the detailed generation of such files is very close to the problem of code generation from UML models to Java language [19].

Code Generation: The code generation from an XMI file is a flexible approach that uses Eclipse Modeling Framework (EMF) which is a very prominent modeling framework and a code generation facility that uses model specification in XMI 2.1 format [17]. Figure 9 illustrates the code generation steps.

The Data Warehouses Code Generation: In this section we describe briefly the MDA-based development process for the the DW code generation, for more detail refer to [16]. Figure 10 illustrates the transformation chain to be applied.

After the ETL extraction phase, to define the multidimensional schema from internal and external data sources, we proposed the two stage process:

- The definition of an elementary functional dependency graph (FDG) between dimensions, without cycles nor transitivity, the graph root is the selected fact [16].
- Transforming the graph into the relational model [16].

After the FDG construction, each FDG branch corresponding to a dimension is explored for the target transformation. Transformations are represented in accordance with the ATL language [13], thus relations between the meta-models elements are used for building transformations between PIM and PSM models [11].

The code is generated from the PSM to create data structures according to a specific platform. In this case the platform chosen for the implementation of the data warehouse is a relational one, then we need to create tables, primary keys, foreign keys and so on [16].

Generally, the code is not generated directly from the UML class diagram and it is preferable to transform the class diagram into a schema that represents the data through their relational modeling elements. The code is then deducted from the relational schema.

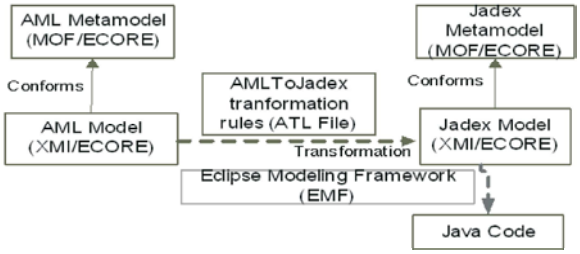

Fig. 9: Code generation steps

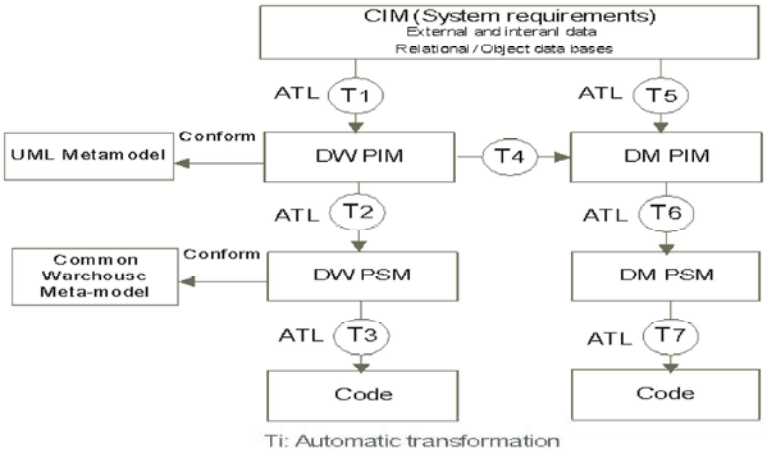

Fig. 10: Development process/Transformation chain

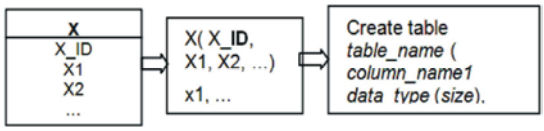

Fig. 11: Code generation process

One of the advantages is to represent the relational structure of data in a graphical form that gives an overview of data, free from any constraint language syntax to generate (eg SQL) (Figure 11).

CONCLUSION AND PERSPECTIVES

In this work we have addressed the issue of the decision support systems development, while taking into account the environment constraints. We were particularly interested in the DSS architecture and the building of the ETL process in term of agents. For this, we adopted an MDA based approach in order to generate the DW code and a large part of the ETL system code. The perspectives of this work are the agent detailed implementing of the system.

ACKNOWLEDGEMENTS

The authors wish to acknowledge the contributions of other members of the engineering laboratory of computer systems for their helpful discussions and the availability of all resources that have helped make this work in the best conditions. They would also like to thank the reviewers for their remarks and suggestions.

REFERENCES

1. Rud, O., 2009. Business Intelligence Success Factors: Tools for Aligning Your Business in the Global Economy. Hoboken, N.J: Wiley & Sons.
2. Shaker, H., E. Ali, M. Abdeltawab, H. Ahmad and H.E. Ali, 2011. A proposed model for data warehouse ETL processes, Journal of King Saud University C Computer and Information Sciences, 23(2): 91-104.
3. Kimball, R. and J. Caserta, 2004. The Data Warehouse ETL Toolkit. Practical Technique for Extracting, Cleaning, Conforming and Delivering Data. Wiley Publishing. Inc. Indianapolis.
4. Mellor, S., K. Scott, A. Uhl and D. Weise, 2004. MDA distilled: principles of Model-Driven Architecture, Addison-Wesley.
5. Tariq, A. and R. Khan, 2012. Intelligent Decision Support Systems- A Framework. Information and Knowledge Management, 2(6): 12-19.
6. Kimball, R. and M. Ross, 2002. The Data Warehouse Toolkit. second edition, John Wiley & Sons.

7. Gloria, P.W. and L. Jain, 2007. Recent Advances in Intelligent Decision Technologies. Lecture Notes in Computer Science, 4692: 567-571.
8. Foster, D., C. McGregor and S. El-Masri, A Survey of Agent-Based Intelligent Decision Support Systems to Support Clinical Management and Research. In the Proceedings of the Workshop on Multi-Agent Systems for Medicine, Computational Biology and Bioinformatics in association with the 4th International Joint conference on Autonomous Agents and Multi-Agent Systems, pp: 16-34.
9. McGregor, C. and S. Kumaran, 2002. An Agent-Based System for Trading Partner Management in B2B e-Commerce. In the Proceedings of the 12th International Workshop on Research Issues on Data Engineering, pp: 84-89.
10. Boussaid, O., F. Bentayeb ans J. Darmont, 2003. An MAS-Based ETL Approach for Complex Data. In the Proceedings of the 10th ISPE International Conference on Concurrent Engineering: Research and Applications, pp: 49-52.
11. Mazón, J.N. and J. Trujillo, 2008. An MDA approach for the development of data warehouses. Decision Support Systems, 45(1): 41-58.
12. Arrassen, I., A. Meziane, R. Sbai and M. Erramdani, 2011. QVT transformation by modeling From UML Model to MD Model. International Journal of Advanced Computer Science and Applications, 2(5): 7-14.
13. Jouault, F., F. Allilairea, J. Bézivin and I. Kurtev, 2008. ATL: A model transformation tool. Science of Computer Programming, 72(1-2): 31-39.
14. Lavbic, D. and Rok Rupnik, 2009. Multi-Agent System for Decision Support in Enterprises, Journal of Information and Organizational Sciences, 33(2): 269-284.
15. Kishore, R., H. Zhang and R. Ramesh, 2006. Enterprise integration using the agent paradigm: foundations of multi-agent-based integrative business information systems. Decision Support Systems, 42(1): 48-78.
16. Elfazziki, A., A. SADIQ and M. Sadgal, 2013. A model driven architecture for the development of decision support systems. In the proceeding the International Conference on Web and Information Technologies.
17. Elfazziki, A., Nouzri and S. Sadgal, 2012. M. An agent oriented information system: an MDA based development. International Journal of Computer Applications, 47(4): 1-8.
18. Trencansky, I. and R. Cervenka, 2005. Agent Modeling Language (AML): A Comprehensive Approach to Modeling MAS. Informatica, 29(4): 391-400.
19. Mészáros, A., 2010. Code Generation from AML to Jadex, Master's thesis, Comenius University in Bratislava.