A Case Study on Model Driven Data Integration for Data Centric Software Development

Hyeonsook Kim
Thames Valley University
W5 5RF St Mary's Road
London, United Kingdom
+44 0208 579 5000
hyeonsook.kim@tvu.ac.uk

Ying Zhang
Thames Valley University
W5 5RF St Mary's Road
London, United Kingdom
+44 0208 579 5000
ying.zhang@tvu.ac.uk

Samia Oussena
Thames Valley University
W5 5RF St Mary's Road
London, United Kingdom
+44 0208 579 5000
samia.oussena@tvu.ac.uk

Tony Clark
Thames Valley University
W5 5RF St Mary's Road
London, United Kingdom
+44 0208 579 5000
tony.clark@tvu.ac.uk

ABSTRACT
Model Driven Data Integration is a data integration approach that proactively incorporates and utilizes metadata across the data integration process. By decoupling data and metadata, MDDI drastically reduces complexity of data integration; whilst also providing an integrated standard development method, which is associated with Model Driven Architecture. This paper introduces a case study to adopt MDA technology as an MDDI framework for data centric software development; including data merging and data customization for data mining. A data merging model is also proposed to define relationships between different models at a conceptual level which is then transformed into a physical model. In this case study we collect and integrate historical data from various universities into the Data Warehouse system in order to develop student intervention services through data mining.

Categories and Subject Descriptors
H.2.0 [**Database Management General**]: Modeling of Data Integration – *conceptual data integration model, metadata model and interchange, model transformation.*

General Terms
Design, Management, Experimentation.

Keywords
Model Driven Data Integration, Data Warehouse, Model Driven Architecture, Data Merging.

1. INTRODUCTION
Data integration involves combining data from different sources and providing users with a view of this data combined together. Data integration is a complex, time consuming, and unreliable method due to the fact that it involves various distributed data sources and many numerous people working from various parties. Data sources which are generally designed for different purpose make the built of a united data repository difficult. Model Driven Data Integration (MDDI) has been proposed to resolve the problem of data integration, by reducing the complexity of data integration. This is achieved by decoupling data and metadata.

MDDI is a data integration approach that actively incorporates and utilizes metadata across the data integration process. In applying the model transformation framework of Model Driven Architecture (MDA) [2] it provides an integrated development method as well as support for system evolution, integration, interoperability, portability, adaptability and reusability [6].

For MDDI approach, Common Warehouse Meta-model (CWM) [4] has been leading the industrial standards, which support metadata modeling and metadata interchange between different platforms and tools. CWM uses UML [3], MOF [20], and XMI [17] to model, manipulate, and interchange warehouse metadata including both technical and business metadata. UML provides a language for modeling metadata, whilst MOF presents APIs for manipulating metadata, and XMI guides mechanisms for interchanging metadata in XML. Despite this, CWM is insufficient to represent all peculiarities of Data Warehouse (DW) modeling in a conceptual level and is too complex to be handled by both end users and designers [18].

There have been several research works that solve the problem by providing a conceptual ETL (Extraction, Transformation, and Load) mapping model with UML extension [5] [12] and their own graphical notation [13]. This also applies for the process based data integration [7] and conceptual data integration framework, which [6] are proposed in a similar manner. However, some of these works do not properly address how to compromise industrial standards in their approaches, failing to show adaptation of their model into real system. The details of previous research works are described in section 2.

In this paper, we present a case study on an MDDI associating with MDA for data centric software development that has been applied in the MCMS (Mining Course Management Systems) project. This case study addresses modeling aspects and issues of each data integration process and shows how to utilize industrial standards and tools in the context of MDDI. A logical data merging modeling which is not properly supported by the existing standards is proposed in the project. The merging of models at the logical level includes a data merging PIM in UML and its transformation into PSM in CWM.

This paper is organized as follows. Section 2, discusses related research works; Section 3, introduces the MCMS project which experimentally applies MDA on data integration and discusses the data integration processes; Section 4, displays the data merging model for MDDI and its transformation; Section 5 and 6

DSMM'09, November 6, 2009, Hong Kong, China.

summarizes the entire paper and provide bibliography respectively.

2. Related Works

Several researches have been suggested in order to overcome the challenges in the design of data integration, when in context of MDDI. In this section, we present a brief discussion about some relevant approaches.

In [6], MD2A (Multi Dimensional Model Driven Architecture) is suggested as an approach for applying the MDA framework to one of the stages of the DW development: multidimensional (MD) modeling. The authors defined MD PIM, MD PSM and necessary transformations. Although the suggested framework and models covers formalized MDDI, the designed models do not properly address data merging.

For conceptual modeling of data mapping, [13] suggests an ETL mapping model with their own graphic notation. However, [12] conversely extends UML to model inter-attribute mapping at the attribute level. A conceptual model can be identified with a PIM in the context of MDA as it describes the necessary aspects of the application independently of the platform on which it will be implemented and executed [2]. Although both works present the mapping between a data source and target in different levels of granularity, they do not cover linking to PSM, which is usually transformed from PIM.

[7] Proposes the model-driven generation and optimization of integration tasks using a process-based approach. The approach models data integration process in a high abstraction level in order to raise portability and lower maintenance effort. Although it provides modeling whole integration process rapidly, it does not consider details of each integration process modeling such as data mapping.

Furthermore, several automated data merging approaches are also researched in order to reduce human intervention for data merging through extraction of combined meta data from source data or source meta data in [14] and [15]. Particularly, [10] and [11] describes semi-automated model transformation using matching transformations and weaving models which can be applied to the generation of merged models.

3. Mining Course Management Systems (MCMS) Project

MCMS project [9] funded by JISC (Joint Information Systems Committee) proposes to build a knowledge management system based on data mining in Thames Valley University. Different data sources from current university systems (such as the library system, student administration, or e-learning) are integrated as Data Warehouse (DW) [1] through ETL processes. Text mining has applied to data extraction since a number of data sources are combined together in different formats, including documents as well as database files.

3.1 Data Sources

In this project, we have collected 3 years institutional historical data to build a DW and to predict individual student performance and dropout as well as the suitability of the course or module for student intervention. A brief explanation about each data source and their usage for MCMS is stated below:

(1)Student Record System holds information about student records for example student background, examination results and course enrollment. It is the most important data source for our project.

(2)Online Learning System allows tracing students' activities in virtual class, which includes downloading of class materials and joining online group discussion.

(3)Library System provides information student loan history with details book borrowed.

(4)Reading list system is hosted on the library system server, but has separated database. It can help us track how often the students borrow books from the recommendation list.

(5)Online Resource System holds logs of students' access to online resources such as e-book and online journal.

(6)Programme Specification is a document which provides course information. Text Ming was applied to find out course title, learning and teaching method etc.

(7)Module Study Guide is a text data source which provides module information. It contains details of a module including student assessment strategy, learning outcome and reading list.

(8)Course Marketing System is developed for the purpose of marketing usage, which gives more course information.

(9)Online Test System enables all to take online entrance skills check. It includes language, math and ICT ability test.

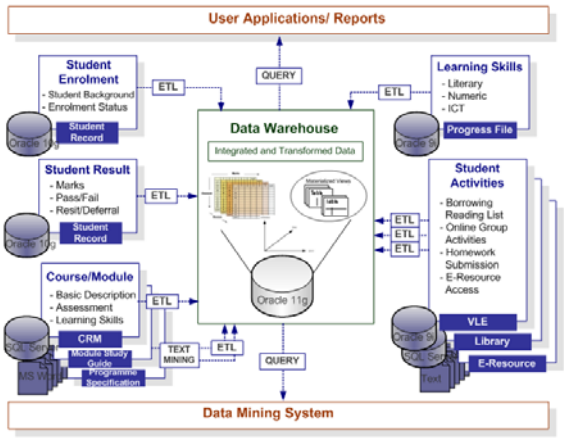

Figure 1. MCMS system overall architecture.

Figure 1 shows the system architecture of MCMS: all data sources are integrated and transformed into data warehouse.

Currently we have three years worth of data to work with: For Course marketing system, we have 5,458 records, which include 1,881 courses; 5,352 Course Offering; 7 Schools and 7 Faculties. For Student Record system, there are 5,800 Students, 5,352 Course_Enrolments. For Library system, there are 144,604 Borrowers, 3,150,816 Loans, 630,190 Items, 435,113 Works and 45,900 Classification. For Reading-List system, there are 552 Course, 1,540 List and 7,084 List_Entry. For online Learning systems, there are 2,460 module offering, and 2,021,334 online activities.

3.2 Data Integration Process

The MCMS project has adopted MDDI associating with MDA approach to build a DW. Every data model for each DW building phase has been designed as Platform Independent Models (PIM) first, and then automatically or manually transformed into individual Platform Specific Models (PSM) that are then transformed into real codes for actual execution. This case study mainly concerns data merging and customization rather than Multi-Dimensional (MD) design since the DW is not referenced from OLAP application but from data mining application in this project.

Oracle 11g enterprise data base and oracle DW developer have been utilized as data base server and ETL tool respectively, and IBM rational architect and data architect for data modeling has been used to activate the MDDI.

As Figure 2 shows, data integration processes can be divided into four phases; (1) analyzing and understanding data in the different data sources, (2) preparing and collecting data into staging area; usually one physical platform, (3) combining data through data cleansing, merging and, transformation, which is usually called ETL. Then finally, (4) customizing data according to application purpose [19].

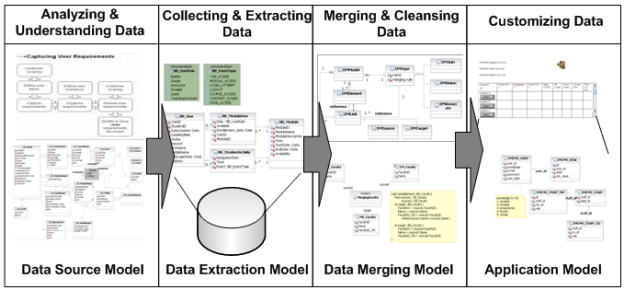

Figure 2. Four modeling phases for the MDDI.

(1)Phase of analyzing and understanding data: In this phase, it is crucial to gather business requirements, determine data quality requirements and, understand data and its associated quality both in the source system and across multiple source systems. Most data models were generated from each data source systems directly using reverse engineering. However some data source models could not be derived directly from the data instances due to different formats of data sources and security policy that led to involvement of domain experts or data specialist.

(2)Phase of collecting and extracting data: Defining the gap between available data and its quality on business requirements were followed as the next step. To remove the gap, the data extraction models for each data sources were defined using the UML class diagram. For example, in student record system, a UML model is designed with the entities of the student, module, course, faculty, staff etc and, theirs relationship. Based on the extraction model, data preparation that includes: gathering, reformatting, consolidating, transforming, cleansing, and storing data was processed both in staging areas and the DW.

(3)Phase of merging and cleansing data: We designed our data merging models to combine each extraction data models into a united model, considering data cleansing as well. Several ETL tools support these processes including data cleansing and data mapping. However, the tools were not helpful in this design stage because these tools tended to be tightly bound to physical models. As a result a new merging model has been developed in order to describe the merged model using UML and rule description. For example, a student-mapping model describes the integration of student information that are stored in of the student record system, library system, online learning system and, online test system. The details of this merging model are provided in section 4.

(4) Phase of customizing data: As a final step, the data customization model was designed according to an application scenario. The modeling implied data aggregation and denormalization for data query performance and efficiency of the data mining application.

3.3 Transformation

It is well known that logical models (PIM) provide not only guidance on how to integrate actual data but also automated generation of real code which is ready for execution according to MDA viewpoints. In this context, PIMs, transformation for PSMs and, transformation for real code are necessary for each modeling phase.

Most data source PIMs were derived from real data source through automated reverse transformation, with the support of Rational Data Architect (RDA) in this project. Analyzing the data source PIMs, we designed Extraction PIMs with UML that were transformed to PSMs with ERD and SQL codes in turn later. The transformations were also supported by RDA. Data merging PIMs were designed after building data cleansing strategy and then manually transformed into merging PSMs in Oracle Warehouse Builder (OWB) [16]. PL/SQL packages were generated from merging PSMs by OWB. The PIMs, PSMs, real codes and their transformations are shown briefly in Figure 3.

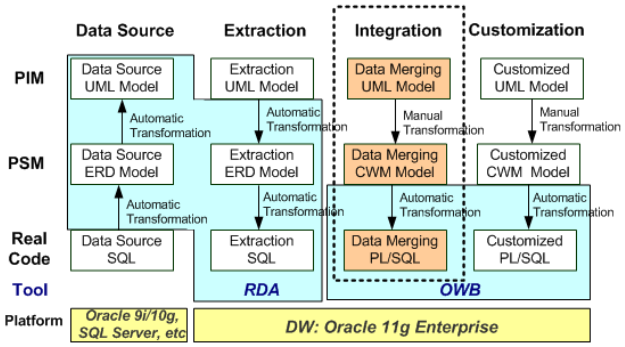

Figure 3. Models and transformations for MDA based MDDI

Nowadays most data modeling tools support reverse engineering which automatically transforms physical data schema into its physical ERD model and its logical UML model, as well as forward engineering for automated transformation of PIM to PSM and to real code as RDA and OWB does. Most ETL tools provide automated generation of source code for metadata generation and data processing by designing PSM based on CWM.

Transformation between UML and CWM is the only form not supported by ETL tools, as the tools do not provide conceptual data modeling. UML as itself has shortages in expressing data mapping that requires defining relationships between attributes [13]. Therefore we proposed a conceptual data merging model

and rules to support transformation of merging PIM into merging PSM. It will alleviate the issues a database designer has to keep consistent data model by deriving the physical model and source code from PIM during implementation. Details of the new data-merging model are shown in the next section.

4. Data Merging

4.1 Data Merging Model (PIM)

Our data merging model defines relationships between model elements from different models at the conceptual level. This relationship of meta data realize data mapping and shows how to move each source data to the target.

Our data merging model uses UML to express merging type and relationship of merging entities. This is illustrated in Figure 4.

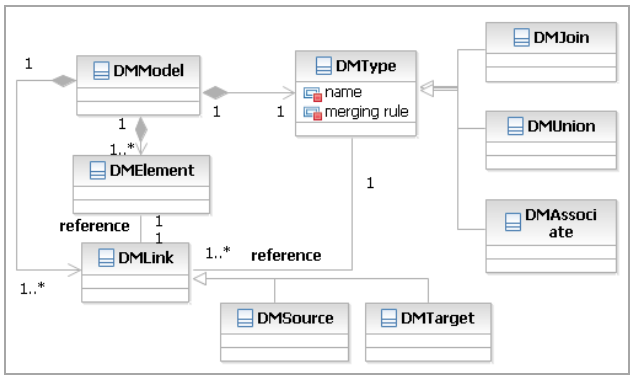

Figure 4. The merging meta model.

The root element, *DMModel*, is composed of DMType, DMElement and, DMLink elements. The description of each element is as follows:

- *DMType* is a base model of *DMJoin*, *DMUnion* and *DMAssociation* , which determines the merging method. A data mapping rule script attached on the DMType specified details of data mapping and their order.
- *DMJoin* is a type of merging which finds a joint data set of all linked source elements and moves the data into a target element.
- *DMUnion* moves all data from each source elements to a target element according to their order.
- *DMAssociate* replaces association of source elements to the one of the target.
- *DMElement* represents model elements including both source and target.
- *DMLink* shows relationship and directions of data mapping.
- *DMSource* inherits *DMLink* to identify source elements.
- *DMTarget* inherits *DMLink* to identify a target element.

Figure 5 depicts a simple example of data merging model using the merging meta model. In this example, two model elements from different systems have exactly the same data structure, however they differentiate through the reference of the different data objects for faculty. This ensures that when you merge the two elements together it changes the reference as well as data.

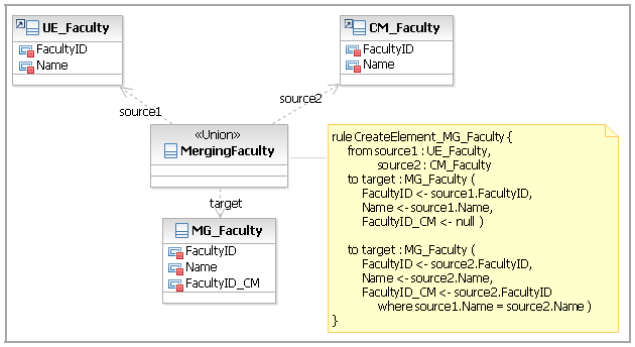

Figure 5. An example of data merging PIM.

The *ruleCreateElment_MG_Faculty* describes how to map the attributes of source elements, *UE_Faculty* and *CM_Faculty*, to the target element, *MG_Faculty*. The rule has a set of {*sources*, *targets*} and *targets* has a set of {target *element name*, *a set of attributes mapping*}. *Attribute mapping* is expressed with arrow directing from source attribute to target attribute. As an example, *DMUnion* moves the first source element into a target element on 'insert' basis and the others on 'update and insert' basis.

4.2 Model Transformation (into PSM)

The PIM merging model can be changed into OWB PSM model by model transformation rule. A *DMElements* is mapped into a table, especially *DMSource* references' existing data table and a *DMTarget* creates new data schema to merge *DMSource* data. The full description of transformation rule is listed in Table 1.

Table 1. Models Transformation Rule

DM Model	Transformation Rule
DMElement	-If *DMElement* is connected with *DMSource* link, generate a reference to an existing table. - If *DMElement* is connected with *DMTarget* link: create new table schema including primary key and foreign key constraints. -If an attribute of *DMElement* is not a primitive type, change table constraints on foreign key to reference a proper element.
DMUnion	-Create data mappings as much as the number of *DMSource* links. -According to the mapping order in rule script, each data mapping from a source to a target is transformed into each attribute connection between source and target elements in turn. - If attributes of source and target are not same type, insert data type change function before mapping data.
DMJoin	-Create a data mapping using *joiner* entity to merge source elements -From rule script, joining condition and mapping sequence are determined.
DMAssociation	-Change target table schema -Update target table schema to reference a source table with foreign key constraint.

DMLSource/ DMTarget	- No correspondent transformation. Just indicate whether a linked *DMElement* is a source element or a target one.

As an example, the transformed PSM of Figure 5 is shown in Figure 6. Since the *DMUnion* is transformed into several OWB data mappings depending on number of source elements, two data mappings are created in this example. According to the *ruleCreateElment_MG_Faculty*, *UE_FACULTY* element is mapped into *MG_FACULTY* then *CM_FACULTY*. *CM_FACULTY.FACULTY_NAME* is converted to string data type before it is mapped since the target *MG_FACULTY.NAME* is of different data type.

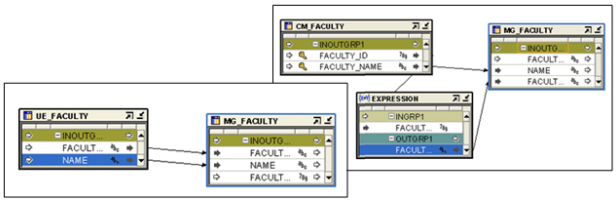

Figure 6. An example of data merging PSM.

The following script shows a part of generated PL/SQL package code from the PSM of Figure 6.

```
.... IF NOT "MG_FACULTY_St" THEN
  batch_action := 'BATCH INSERT';
  batch_selected := SQL%ROWCOUNT;
  INSERT
     INTO
          "MG_FACULTY" ("FACULTYID", "NAME")
          (SELECT "UE_FACULTY"."FACULTYID" "FACULTYID",
          "UE_FACULTY"."NAME" "NAME"
     FROM "UE_FACULTY"  "UE_FACULTY"......
```

5. CONCLUSION

In this paper, we have presented a case study that shows detailed process of an MDDI with specific models for each individual modeling phases. In this case study, our aims were to maximize the utilization of current industrial standards for an MDDI project and to propose a logical data-merging model that the standards have not referenced. We demonstrated the possibility of our data-merging model and how efficiently and rapidly a DW can be developed in the MDDI approach through the MCMS project in Thames Valley University. We modeled each data source with UML to extract designated data from different data source systems. We then designed merging models to integrate all data into one united model as well as customizing the data models for data mining.

We believe these models and the transformations can be reused as a DW modeling framework for any universities that need to build a united data repository for enhanced data utilization and decision-making. Currently we are formalizing our data transformation using MOF QVT (Query/View/Transformation) for automated PSM generation. Automated data merging and cleansing using data mining is also considered as future works.

6. REFERENCES

[1] Kimball, R. and Ross, M. 2002. The Data Warehouse Toolkit, second edition, John Wiley & Sons.

[2] Kleppe, A.,Warmer, J. and Bast,W. 2003. MDA Explained. The Model Driven Architecture: Practice and Promise. Addison-Wesley, Reading.

[3] Object Management Group (OMG). 2009. Unified Modeling Language (UML), Specification 2.2

[4] Object Management Group (OMG). 2003. Common Warehouse Metamodel (CWM), Specification 1.1

[5] Vassiliadis, P., Simitsis, A., and Skiadopoulos, S. Conceptual Modeling for ETL Processes, OLAP'02. 2002.

[6] Mazon, J., Trujillo, J., Serrano, M., and Piattini, M. Applying MDA to the development of Data Warehouses, DOLAP'05, 2005.

[7] Bohm, M., Habich, D., Lehner, W., and Wloka, U. Model driven development of complex and data intensive integration processes, MBSDI 2008, CCIS 8, pp.31-42

[8] Chaudhuri, S. and Dayal, U. 1997. An overview of data warehousing and OLAP technology. SIGMOD Rec. 26, 1 (Mar. 1997), 65-74.

[9] Oussena, S. 2008. Mining Courses Management Systems. Thames Valley University. http://samsa.tvu.ac.uk/mcms

[10] Fabro, D.D.M. and Valduriez, P. Towards the efficient development of model transformations using model weaving and matching transformations, Conference of Software and Systems Modeling 2008.

[11] Marcos, D.D.F., Jean B. and Patrick V., Weaving Models with the Eclipse AMW plugin, Eclipse Modeling Symposium 2006.

[12] Mora1, L.S., Vassiliadis, P., and Trujillo, J., Data Mapping Diagrams for Data Warehouse Design with UML, volume 3288 of Lecture Notes in Computer Science, pp 191-204.

[13] Vassiliadis, P., Simitsis, A,. and Skiadopoulos, S. Conceptual Modeling for ETL Process, ACM Fifth International Workshop on Data Warehousing and OLAP 2002.

[14] Konigs, A. Model Transformation with Triple Graph Grammars. Model Transformations in Practice Satellite Workshop of MODELS 2005. Montego Bay, Jamaica.

[15] Embley, D.W., Xu, L., and Ding, Y. Automatic Direct and Indirect Schema Mapping: Experiences and Lessons Learned, SIGMOD Record, Vol. 33, No. 4, December 2004.

[16] Oracle Warehouse Builder (OWB) 11g Documentation, http://www.oracle.com/technology/products/warehouse

[17] Object Management Group (OMG). 2007. XML Metadata Interchange (XMI), Specification 2.1.

[18] Medina, E. and Trujillo, J. A Standard for Representing Multidimensional Properties: The Common Warehouse Metamodel (CWM). Lecture Notes In Computer Science, vol. 2435. Springer-Verlag, London, 232-247.

[19] Rahm, E., and Do, H. H. 2000. Data Cleaning: Problems and Current Approaches, Journal of IEEE Data Engineering Bulletin, volume 23.

[20] Object Management Group (OMG). 2005. Meta Object Faciality (MOF) Specification 2.