Efficient development of data migration transformations

Paulo Carreira
Oblog Consulting and FCUL
paulo.carreira@oblog.pt

Helena Galhardas
INESC-ID and IST
hig@inesc-id.pt

ABSTRACT

In this paper, we present a data migration tool named DATA FUSION. Its main features are: A domain specific language designed to conveniently model complex data transformations; an integrated development environment that assists users on managing complex data transformation projects and an auditing facility that provides relevant information to project managers and external auditors.

1. INTRODUCTION

Current data migration applications aim at converting legacy data stored in sources with a certain schema into target data sources whose schema is predefined. Organizations often buy applicational packages (like SAP, for instance) that replace existing ones (e.g., supplier management). This situation leads to data migration projects that must transform the data model underlying old applications into a new data model that supports new applications. The migration process is first exhaustively tested and then applied in a one-shot operation, usually during a weekend. The original data sources become obsolete once the migration is performed. The transformation step of the ETL (Extract-Transform-Load) process involved in large-scale data migration projects has two kinds of requirements. The first one concerns the specification of migration transformations. The second deals with the project development and management.

Several issues arise when specifying data migration transformations. First, migration programs require more powerful languages than those supported by most commercial ETL tools currently available. In fact, those languages are usually not expressive enough to represent the semantics of the transformation rules involved. Typically, complex transformations are handled by *ad-hoc* programs coded outside the tools. Second, data migration programs need more than simple programmers. People that write migration code are often business experts as well. They prefer to use high-level constructs that can be easily composed. Third, the cost involved in the production and maintainability of migration programs must be minimized. Migration code must be short, concise and easily modifiable.

Data migration projects deal with large amounts of data and potentially involve a considerable number of transformations. Therefore, data migration programs are iteratively developed. In real world projects, easy prototyping is thus an imperative requirement. Moreover, as in any other software development effort, code and data must be logically organized into distinct packages. Managing such information is crucial for the success of the initiative.

Finally, migration processes deal with critical data. This means that project auditing is frequent and strict. Auditors want to be sure that the entire set of source data is being migrated, i.e., that the migration transformations cover all source records. To ensure this, they need a tool that measures the progress of the migration, and reports which source fields have been migrated and which target fields have been populated.

DATA FUSION [1] is a data transformation platform developed and commercialized by Oblog Consulting that has been used in real data migration projects. Although the company plans to extend the tool with native support for information fusion, for the moment only the data migration component named DATA FUSION DM is fully operational. It is currently being applied by the Spanish software house INDRA [2] to migrate financial data and by Siemens [3] to integrate three databases storing Portuguese public administration information.

1.1 The DATA FUSION DM component

DATA FUSION offers a domain-specific language named *Data Transformation Language* (DTL) for writing concise and short programs. It also provides an Interactive Development Environment (IDE) for efficiently producing and maintaining code.

DTL provides a set of abstractions appropriate for expressing the semantics associated to data transformations. The basic concept is a *mapper* that may enclose several rules. A *rule* encloses transformations with similar logics, e.g., populate fields with the *null* value. The choice of providing such domain-specific language brings several advantages. First, migration solutions can be expressed in a language close to the problem domain. Second, programs are usually concise and easy to read and maintain. Due to these two fea-

SIGMOD 2004 June 13-18, 2004, Paris, France.
Copyright 2004 ACM 1-58113-859-8/04/06 . . . $5.00.

tures, DTL is appropriate for easy prototyping and testing, which are major requirements of data migration applications. Third, the compiler can check if the specific vocabulary is correctly used. In DTL, for example, a target attribute cannot be assigned twice. Since DTL embodies domain knowledge, a number of optimizations that could not be identified otherwise, can be introduced. Finally, a debugger facility can be developed for data migration programs. The debugger facility implementation of Data Fusion DM is in progress.

The Data Fusion DM IDE supports the development of data migration projects. It follows the trend of modern environments for software development (like e.g., Eclipse or Visual Studio). It includes a text editor that supports known functionalities such as syntax highlighting and code templates. Moreover, the DTL compiler is integrated within the IDE and provides helpful hints when compilation errors occur.

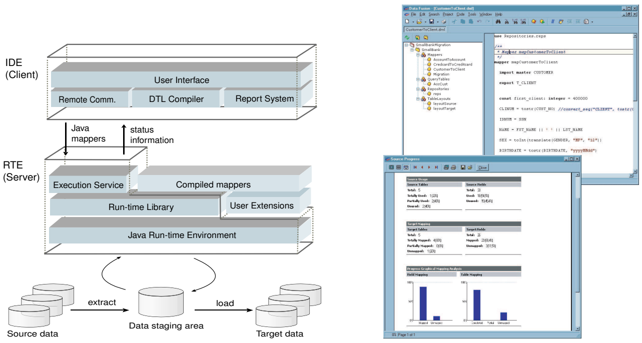
The IDE also supports project management. First, the code produced is organized into packages according to the functionality provided. This feature is extremely important in large-scale projects as is the case of data migration. Second, the IDE provides a project tracking facility that shows to be very useful in real data migration applications. The information to migrate is precious in the sense that every source record must be migrated and every slot of the target schema must be filled in. Auditing a data migration project is a very common activity. People owning data to be migrated frequently ask for periodically checking the progress of the data migration process. The IDE reports the state of all source and target fields, i.e., the association between all target and source fields, the percentage of source and target data already migrated, etc..

Data Fusion DM assumes that the source-target schema mappings are known. The tool does not offer any facility for discovering schema mappings as it is the case of other research tools.

2. ARCHITECTURE

The Data Fusion platform follows the client-server architecture depicted in Figure 1*a*. On the client side, the *Integrated Development Environment* (IDE) allows users to work in multiple data migration projects. On the server side, the *Run-Time Environment* (RTE) is responsible for compiling and parallelizing the data migration requests submitted from IDE instances. This client-server architecture attains scalability. An instance of the IDE may submit requests to multiple RTE instances and an instance of the RTE may run in parallel accepted submissions from multiple IDE instances.

The IDE is constituted by: *(i)* the graphical user interface, which is a development environment for DTL specifications, *(ii)* the remote communication subsystem in charge of submitting the compiled mappers and receiving the migration progress information, *(iii)* the DTL compiler that generates Java code from DTL mappers, and *(iv)* the report system that is responsible for displaying project tracking and auditing information.



```
use Repositories.reps

/**
 * Mapper mapCustomerToClient
 */
mapper mapCustomerToClient

  import master CUSTOMER

  export T_CLIENT


  const first_client: integer = 400000

  CLINUM = tostr(CUST_NO) //convert_seq("CLIENT", tostr(

  IDNUM = SSN

  NAME = FST_NAME || ' ' || LST_NAME

  SEX = toInt(translate(GENDER, "MF", "12"))

  BIRTHDATE = tostr(BIRTHDATE, "yyyyMMdd")
```

Source Usage

Source Tables
Total: 5
Totally Used: 1(20%)
Partially Used: 2(40%)
Unused: 2(40%)

Source Fields
Total: 33
Used: 18(54,5%)
Unused: 15(45,4%)

Target Mapping

Target Tables
Total: 5
Totally Mapped: 4(80%)
Partially Mapped: 0(0%)
Unmapped: 1(20%)

Target Fields
Total: 26
Mapped: 23(88,4%)
Unmapped: 3(11,5%)

Figure 1: ***(a)*** **On the left, the architecture of Data Fusion,** ***(b)*** **on the right, a snapshot of the IDE.**

The RTE is composed by: *(i)* an execution service responsible for processing submission requests by compiling, launching and monitoring the execution of mappers, *(ii)* a run-time library that implements the semantic concepts of DTL, and *(iii)* the Java Run-time Environment which is responsible for executing the Java code.

The transformations are executed by the RTE on a data staging area which can be supported by any RDBMS with a JDBC connection. Data extraction and loading are performed by third-party tools (e.g., Oracle SQL*Loader).

3. SCENARIO DEMONSTRATED

The scenario to demonstrate illustrates the generic characteristics of a data migration project. We present a constructed example of a banking migration. The Banking information system is composed of four applications: Clients, Accounts, Loans and Credit-cards. The data handled by these applications must be migrated into a pre-defined target schema. With this demonstration, we want to outline the following points:

1. Complex legacy data transformations – We illustrate a set of data migration transformations expressible in DTL that are either not tackled or are impractical in existing tools and frameworks.
2. IDE – We show a development environment for DTL specifications (see a snapshot in Figure 1*b*). In particular, we present how the IDE project management handles the migration of real world financial data systems with thousands of tables.
3. Project tracking and auditing – We show how coverage metrics indicate the progress of rule coding. We also show how auditors take advantage of the data dependency reports to gain insight and confidence about the migration specification.

4. REFERENCES

[1] Oblog Consulting. Data Fusion home page. http://datafusion.oblog.com/.

[2] INDRA. Indra home page. http://www.indra.es.

[3] Siemens. Siemens home page. http://www.siemens.pt.