EasyChair Preprint

№ 4771

How to Implement NoSQL Schemas with ModelDrivenGuide?

Jihane Mali, Faten Atigui, Ahmed Azough and Nicolas Travers

December 23, 2020

How to Implement NoSQL Schemas with ModelDrivenGuide?

Jihane Mali
jihane.mali@usmba.ac.ma
Université Sidi Mohamed Ben Abdellah
Fès, Morocco

Faten Atigui
faten.atigui@cnam.fr
CEDRIC, Conservatoire National des Arts et Métiers (CNAM)
Paris, France

Ahmed Azough
ahmed.azough@usmba.ac.ma
Université Sidi Mohamed Ben Abdellah
Fès, Morocco

Nicolas Travers
nicolas.travers@devinci.fr
Léonard de Vinci Pôle Universitaire, Research Center
Paris La Défense, France

ABSTRACT

With the evolution of data in terms of volume, variety and velocity, designing and developing an Information Systems (IS) requires studying the best solutions to store and manipulate data while respecting the user's requirements. In this demonstration, we show how to implement an IS using ModelDrivenGuide, which is a semi-automated approach based on transformation rules starting from a conceptual model, then going from one logical model to an other by refinement to finally the chosen physical model.

KEYWORDS

NoSQL, MDA, Meta-Model, Model Transformation, Model Refinement, Ecore, QVT

1 INTRODUCTION

For decades, the storage and the exploitation of data has mainly relied on relational databases. With the advent of Big Data, the volume of data has exploded, the heterogeneity has increased tenfold, causing problems of transformation from traditional databases to new storage on the Cloud, whether in terms of storage management, data query, cost or performance. To deal with these problems, *NoSQL* data management systems have appeared since 2009.

Several works have focused on storage and modeling problems of data using NoSQL systems. Most of the studies have proposed either (i) a comparative study between RDB (Relational DataBases) and NoSQL DB (DataBases) and/or how to transform relational data into dedicated NoSQL system [6, 9, 11] or (ii) how to transform a conceptual schema into a specific NoSQL DB [1, 3–5] (iii) while very few studies have proposed criteria for the choice of physical models and implementation platforms [8, 10].

Our ModelDrivenGuide [2, 7] approach offers logical modeling suitable for models refinement in order to generate all types of optimized physical models by relying on a common 5Families meta-model (4 NoSQL families & the Relational model). Based on transformation rules, it provides a functional process that integrates the use case to generate the different SQL and/or NoSQL solution(s) adapted to business requirements.

2 MODELDRIVENGUIDE: FROM CONCEPTUAL MODEL TO PHYSICAL MODELS

We suggest a model driven approach that offers steps to generate a logical model for each family. Our approach focuses on model transformation, starting from conceptual to logical then to physical models, and on model refinement to go from one logical model to another. The peculiarity of this approach is to allow the optimization of the data model directly during the transformation process instead of the last step. This helps to make a choice of implementation according to the context of use. In order to formalize the process, we adopted a Model-Driven Architecture (MDA[1]).

Our ModelDrivenGuide approach (Figure 1) starts from a UML class diagram and considers two Platform Independent levels corresponding to the conceptual (PIM1) and logical (PIM2) levels, as well as a Platform Specific level related to the different target platforms.

The **PIM1** (*Platform Independent Model*) of the first level is an UML class diagram that serves as a basis for modeling the user requirements and the context of use.

The **PIM2** is the second level independent model, common to the five families of models. It allows to carry out *refinement* rules by generating recursively all possible denormalized models by relying on merge and split rules. We mention that a heuristic is applied to guide the generation of data models in order to avoid cycles and useless models. It is mainly based on the use case. This heuristic is not detailed in this demonstration.

The **PSMx** (*Platform Specific Model*) are obtained by the transformation of PIM2 models into the compatible target data family (e.g. nesting for DO, rows for CO, edges for GO, etc.).

This demonstration focuses especially on the PIM2 5Families meta-model used to generate all possible data models, on transformation, and on refinement rules.

2.1 Experimental Environment.

Since our approach is based on (MDA[2]), we need an infrastructure suitable for meta-modeling, modeling and models transformations. We developed our approach using a model transformation environment called *Eclipse Modeling Framework* (EMF[3]). It is a set of Eclipse plug-ins which can be used to design a data model and to generate code or other output based on this model. EMF respects

[1] MDA: https://www.omg.org/mda/index.htm
[2] MDA: https://www.omg.org/mda/index.htm
[3] EMF: https://www.eclipse.org/modeling/emf/

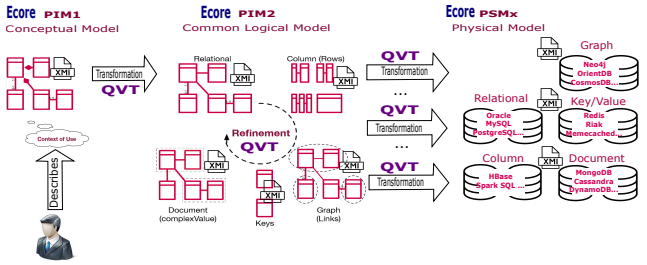

Figure 1: `ModelDrivenGuide` Approach

the known distinction between a meta-model and a model. A meta-model describes the structure of a model. A model is a concrete instance of this meta-model. To implement our approach, we have used the following tools proposed by EMF: 1) *Ecore* used for the implementation of all the `PIM1`, `PIM2`, and `PSMx` meta-models. Inspired by the object-oriented approach, the Ecore language is based on the notion of package (EPackage), class (EClass), attribute (EAttribute), reference link (EReference), data type (EDataType), and enumeration (EEnum); 2) *XMI*[4] a format in which instances of meta-models are created; and 3) *QVTO* (QVT Operational) a Model-To-Model (M2M) transformation tool that implements the QVT language. It is used to formalize both exogenous transformation rules (from conceptual to logical model, and from logical to physical model) and also refinement (endogenous transformation) rules.

3 DEMONSTRATION

Before starting the transformation process, it is necessary to define the three meta-models that we have in our `ModelDrivenGuide` approach since they are going to be needed in any further steps. We defined the conceptual (class diagram), logical (`5Families`) and physical (MongoDB) meta-models using `Ecore`. We also formalized the transformation/refinement rules in QVT.

3.1 TPC-C Benchmark Scenario

For this scenario, we have used the TPC-C[5] benchmark as an entry. It gives a full use case mixing at the same time transactions, joins and aggregations. The TPC-C benchmark simulates the behavior of a logistic DB on user orders with transaction-oriented stock management (OLTP). We will focus on the six classes (*Warehouse*, *District*, *Customer*, *Order* and *OrderLine*, *Item*) in the PIM1.

(1) Instantiate the PIM1 meta-model using TPC-C benchmark's class diagram,
(2) Typically, transform the PIM1 into the PIM2 as a normalized relational model conform to our `5Families` common meta-model,
(3) Apply applied semi-automatic refinement rules recursively,
(4) Transform the chosen model obtained from the refinement into a `MongoDB` database. The choice was led by the fact that `MongoDB` is one of the rare NoSQL solutions integrating ACID transactions in *sharding* (version 4.2) required by the TPC-C benchmark. However, our approach is extensible by defining a new PSM for each target database type (without ACID properties in that case). We can also visualize the JSON schema of the final output.

3.2 User Model Scenario

This scenario illustrates the whole process by integrating user's own data model. The user has to follow the undermentioned steps while using the `ModelDrivenGuide` Approach:

(1) Create its own instance of the source meta-model in the XMI format, in our case the source is the conceptual class diagram meta-model,
(2) Typically, transform the PIM1 into the PIM2 as a normalized relational model,
(3) Apply the heuristic to generate a tree of denormalized data models (using recursively refinement rules),
(4) Choose one or more generated PIM2, and apply its transformation into the target PSM.

4 CONCLUSION

Our `ModelDrivenGuide` approach is a MDA-based approach that aims to improve model transformation. `ModelDrivenGuide` is a global approach that generates optimized models based on the common `5Families` meta-model favoring the application of refinement rules to produce potential target models. This mix between data modeling and optimization rises to an approach that aims to find the efficient target model among the 5 families of data.

For future works, we seek to define a generic cost model allowing to compare the produced solutions and eventually suggest top-k logical models. This cost model will integrate different dimensions (storage, bandwidth, CPU/energy impact, etc.) and will help to make a decision among all produced data models.

[4] *XML Metadata Interchange*: https://www.omg.org/spec/XMI/About-XMI/
[5] TPC-C: http://www.tpc.org/TPCC/default5.asp

REFERENCES

[1] Fatma Abdelhedi, Amal Ait Brahim, Faten Atigui, and Gilles Zurfluh. 2017. MDA-based Approach for NoSQL Databases Modelling. In *International Conference on Big Data Analytics and Knowledge Discovery*. Springer, 88–102.
[2] Faten Atigui, Asma Mokrani, and Nicolas Travers. 2020. DataGuide : une approche pour l'implantation de schémas NoSQL. *Extraction et de Gestion des Connaissances (EGC'20)* RNTI-E-36 (2020), 407–408.
[3] G. Daniel, A. Gómez, and J. Cabot. 2019. UMLto[No]SQL: Mapping Conceptual Schemas to Heterogeneous Datastores. In *2019 13th International Conference on Research Challenges in Information Science (RCIS)*. 1–13.
[4] Gwendal Daniel, Gerson Sunyé, and Jordi Cabot. 2016. UMLtoGraphDB: mapping conceptual schemas to graph databases. In *International Conference on Conceptual Modeling*. Springer, 430–444.
[5] Shady Hamouda and Zurinahni Zainol. 2017. Document-oriented data schema for relational database migration to NoSQL. In *2017 International conference on big data innovations and applications (innovate-data)*. IEEE, 43–50.
[6] Chongxin Li. 2010. Transforming relational database into HBase: A case study. In *2010 IEEE international conference on software engineering and service sciences*. IEEE, 683–687.
[7] Jihane Mali, Ahmed Azgouh, Faten Atigui, and Nicolas Travers. 2020. DataGuide: An Approach for Implementing NoSQL Schemas. In *DEXA'20*. Bratislava, Slovakia, 1–10.
[8] AB Raut. 2017. NoSQL database and its comparison with RDBMS. *International Journal of Computational Intelligence Research* 13, 7 (2017), 1645–1651.
[9] Leonardo Rocha, Fernando Vale, Elder Cirilo, Dárlinton Barbosa, and Fernando Mourão. 2015. A framework for migrating relational datasets to NoSQL. *Procedia Computer Science* 51 (2015), 2593–2602.
[10] Clarence JM Tauro, Shreeharsha Aravindh, and AB Shreeharsha. 2012. Comparative study of the new generation, agile, scalable, high performance NOSQL databases. *International Journal of Computer Applications* 48, 20 (2012), 1–4.
[11] Tamás Vajk, Péter Fehér, Krisztián Fekete, and Hassan Charaf. 2013. Denormalizing data into schema-free databases. In *2013 IEEE 4th International Conference on Cognitive Infocommunications (CogInfoCom)*. IEEE, 747–752.