Model Transformations in the MOF Meta-Modeling Architecture: From UML to CodeIgniter PHP Framework

Oualid Betari, Mohammed Erramdani, Sarra Roubi, Karim Arrhioui and Samir Mbarki

Abstract Over the last few years, with the increased importance of the internet in many domains, web development industry has seen ground breaking changes. To solve the challenge of business and technology change, models have become increasingly important in constructing application systems. For example, OMG's Model Driven Architecture (MDA) uses models as building blocks to support application development. In this paper, we present the application of the MDA approach to model the CodeIgniter PHP framework. We developed the models used for transforming Platform Independent Model (PIM) to Platform Specific Model (PSM), using a UML class diagram as a source model to generate an XML file containing the core components of a CodeIgniter PHP framework.

Keywords Model driven architecture • Unified modeling language • Platform independent model • Platform specific model • Codeigniter • PHP • Framework

O. Betari (✉) · M. Erramdani · S. Roubi
MATSI Laboratory, Superior School of Technology,
Mohammed First University, Oujda, Morocco
e-mail: beta.oualid@gmail.com

M. Erramdani
e-mail: m.erramdani@gmail.com

S. Roubi
e-mail: roubi.sarra@gmail.com

K. Arrhioui · S. Mbarki
MISC Laboratory, Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco
e-mail: arr.karim@gmail.com

S. Mbarki
e-mail: mbarkisamir@gmail.com

1 Introduction

The coding methodology, affected by the rapid development of internet for web based application, indicates a higher need of sustainability and maintainability. PHP, a scripting tool for web that enable dynamic interactive web development, provides an intuitive, compiled fast, cross platform, open source, flexibility as well as required minimal setup [1]. Furthermore, PHP has become one of the most powerful programing languages for developing web applications. In order to solve the problems created by the growing complexity of the projects, Several methods for writing PHP codes such as Object Oriented Programming (OOP), Procedural PHP coding and Model View Controller (MVC) pattern have been proposed. This rapid development gave birth to several frameworks such as CodeIgniter PHP framework. As the time has gone popular PHP frameworks have just got bigger and better and become handy tool for developers to build giant application effortlessly.

Computing infrastructures are expanding their reach in every dimension. New platforms and applications must interoperate with legacy systems while virtual enterprises span multiple companies. New implementation platforms are continually coming down the road, each claiming to be “the next big thing”.

To handle these changes, the Object Management Group (OMG) proposes the Model Driven Architecture as a solution. This approach supports evolving standards in application domains. MDA provides an open, vendor-neutral approach to the challenge of interoperability, building upon and leveraging the value of OMG’s established modeling standards: Unified Modeling Language (UML); Meta-Object Facility (MOF) [2].

In this paper, we consider the unison of the solutions presented by the introduction of the PHP frameworks and the use of models in developing application, as we will apply the MDA approach to model the CodeIgniter PHP framework.

This paper is organized as follows: in the second section, we present a preview to the MDA approach and the CodeIgniter PHP framework. UML and CodeIgniter meta-models and the transformation rules of our approach are introduced in the third section. In the fourth section, we discuss related work. Last section concludes this paper and presents the future directions of our work.

2 MDA and CodeIgniter

2.1 Model Driven Architecture

In late 2000, OMG members first reviewed the document entitled “Model Driven Architecture” and decided to form an architecture team to produce a more formal statement of the MDA [2].

MDA addresses the challenges of today’s highly networked, constantly changing systems, providing an architecture that assures portability, cross-platform

Interoperability, platform independence, domain specificity and productivity [2]. Its architecture [2, 3] is divided into three layers:

- In the first layer, we find the standard UML (Unified Modelling Language), MOF (Meta-Object Facility) and CWM (Common Warehouse Meta-model).
- In the second layer, we find a standard XMI (XML Metadata Interchange), which enables the dialogue between middleware.
- The third layer contains the services that manage events, security, directories and transactions. The last layer provides frameworks which are adaptable to different types of applications namely Finance, Telecommunications, Transport, medicine, E-commerce and Manufacture, etc.).

Note that the term "architecture" in Model-driven architecture does not refer to the architecture of the system being modeled, but rather to the architecture of the various standards and model forms that serve as the technology basis for MDA.

MDA consists of three general types of models, structured into three basic layers, as shown in Fig. 1 [4].

These model types are briefly described as follows [5].

- Computation Independent Model (CIM): It represents a high-level specification of the system's functionalities. It shows exactly what the system is supposed to do, but hides all the technology specifications.
- Platform Independent Model (PIM): It allows the extraction of the common concept of the application independently from the platform target.

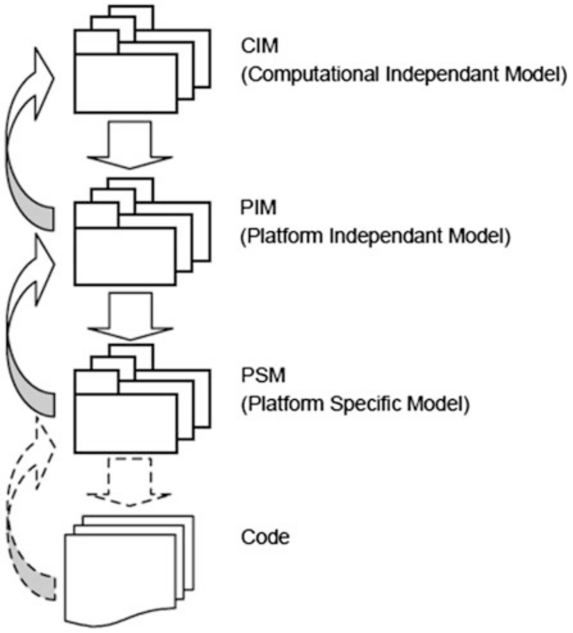

CIM
(Computational Independant Model)
PIM
(Platform Independant Model)
PSM
(Platform Specific Model)
Code

Fig. 1 Model driven architecture layers

- Platform Specific Model (PSM): It combines the specifications in the PIM with the details required of the platform to stipulate how a system uses a particular type of platform which leads to include platform specific details.

2.2 CodeIgniter MVC (Model View Controller)

CodeIgniter is an Application Development Framework—a toolkit—for people who build web applications using PHP. Its goal is to enable to develop projects much faster than writing code from scratch, by providing a rich set of libraries for commonly needed tasks, as well as a simple interface and logical structure to access these libraries.

CodeIgniter is based on the Model-View-Controller development pattern. MVC (Model View Controller) is a software approach that separates application logic from presentation. In practice, it permits your web pages to contain minimal scripting since the presentation is separate from the PHP scripting [6].

- The Model represents the data structures. Typically, the model classes will contain functions that help retrieve, insert, and update information.
- The View is the information that is being presented to a user. A View will normally be a web page, but in CodeIgniter, a view can also be a page fragment like a header or footer. It can also be an RSS page, or any other type of "page".
- The Controller serves as an intermediary between the Model, the View, and any other resources needed to process the HTTP request and generate a web page [7].

3 Proposed Approach

The required steps during the model-driven development with the UML approach can basically be divided into three phases:

- First, a model, independent of any implementation technology, is built with a high level of abstraction. This is called a Platform Independent Model (PIM).
- Next is, the transformation of the proposed PIM into one or more Platform Specific Models (PSMs).
- The final step is to transform the generated model respecting the PSM into the code of the chosen platform. Because a PSM fits its technology very closely, this transformation is rather trivial.

In this paper, we focused on the PIM to PSM transformation using an approach by modeling. We will present next our PIM model which consists of a UML Meta-model, and the CodeIgniter Meta-model as the PSM model.

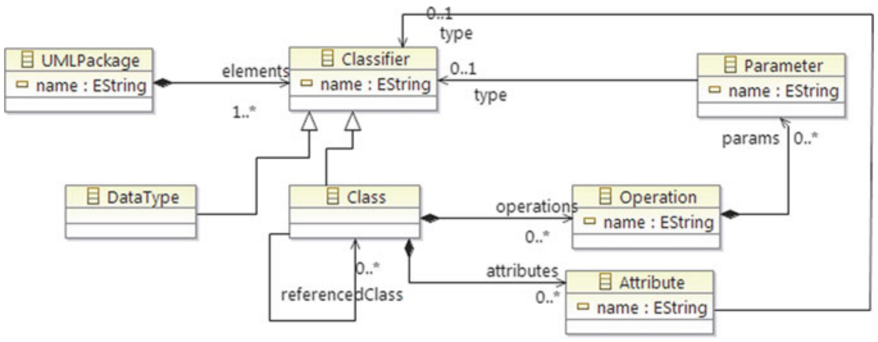


Fig. 2 Simplified UML meta-model

3.1 UML Meta-Model

The UML specification has the Kernel package which provides the core modeling concepts of the UML, our source Meta-model structures a simplified UML model based on a package containing the data types and classes. The classes contain structural features represented by attributes, and behavioral features represented by operations. The Fig. 2 presents the source meta-model.

- UmlPackage: is an abstract element of UML used to group together elements that are semantically related. This meta-class is connected to the meta-class Classifier.
- Classifier: This is an abstract meta-class which describes set of instances having common features. This meta-class represents both the concept of class and the concept of data type.
- Parameter: describes the order, type and direction of arguments that can be given when the operation is invoked. It explains the link between Parameter meta-class and Classifier meta-class.

3.2 CodeIgniter Meta-Model

The developed Meta-model target structure contains the core of the CodeIgniter framework, represented by the model, view and controller packages. This last package is composed of helpers, libraries and the loader which connects it to the database. The modeling of CodeIgniter PHP framework model is shown in the Fig. 3.

- Router recovers the URL and decomposes it into actions, its role is to find the controller to call.

- Controller is used as the parent of the overall controllers created by the user. This controller will be extended to all controllers created by the web site developer.
- Lastly is the Model. It functions as the parent of the models created by the user.

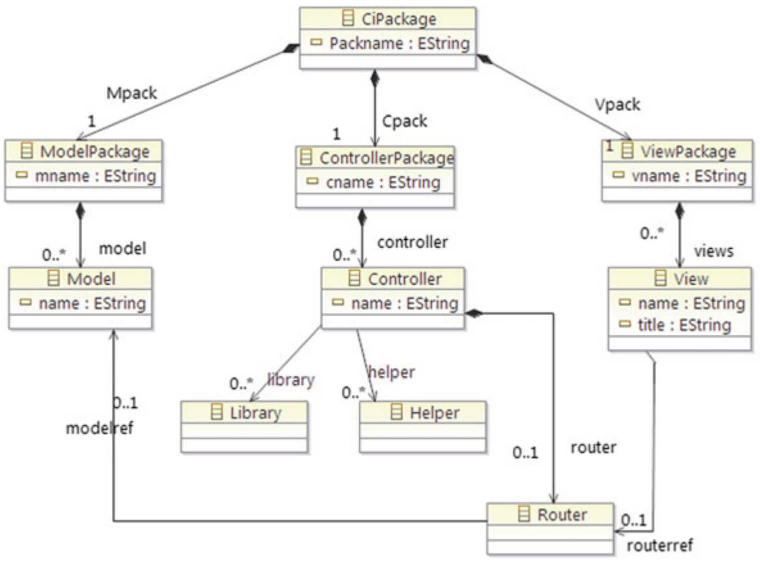
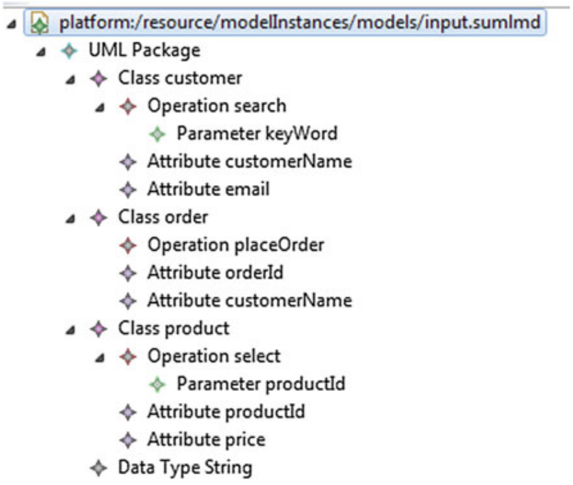
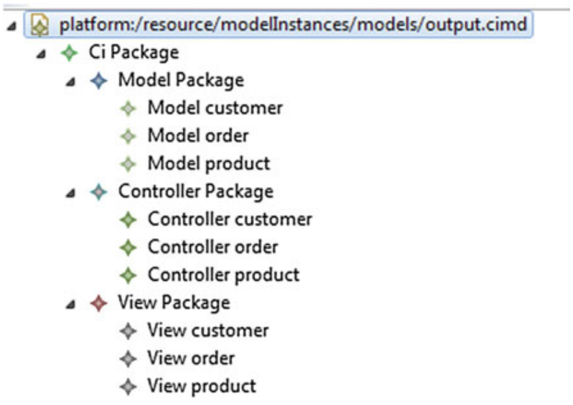


Fig. 3 CodeIgniter meta-model

Fig. 4 UML meta-model instance



```
platform:/resource/modelInstances/models/input.sumlmd
  UML Package
    Class customer
      Operation search
        Parameter keyWord
      Attribute customerName
      Attribute email
    Class order
      Operation placeOrder
      Attribute orderId
      Attribute customerName
    Class product
      Operation select
        Parameter productId
      Attribute productId
      Attribute price
    Data Type String
```



```
platform:/resource/modelInstances/models/output.cimd
  Ci Package
    Model Package
      Model customer
      Model order
      Model product
    Controller Package
      Controller customer
      Controller order
      Controller product
    View Package
      View customer
      View order
      View product
```

Fig. 5 CodeIgniter meta-model instance

3.3 *Models Instances*

In our case study, as an input, we will use an UML model that represents part of the information system of a commercial company. This file, shown in the Fig. 4, describes the elements needed for the transformation.

The output of the model to model transformation should be generated as shown in the Fig. 5, this file contains the major elements of the CodeIgniter Framework.

4 Related Work

There are several works related to the presented approach. Meta-modeling architectures based on the MDA approach can be found in other domains of computer science.

A lot of ideas for this work have been inspired by the paper [8] which applies of the MDA approach to generate, from the UML model, the Code following the MVC2 pattern using the standard MOF 2.0 QVT (Meta-Object Facility 2.0 Query-View-Transformation) as a transformation language.

In our approach, that uses as well the UML model as a source for transforming PIM to PSM, we went for a PHP Framework based on MVC design pattern as our target model.

In a different work, an MDA process have been used to automatically generate the multidimensional schema of data warehouse in [9], This process uses model transformation using several standards such as Unified Modeling Language, Meta-Object Facility, Query View Transformation and Object Constraint Language, from the UML model.

An overview of the MDA has been treated in the European MDA Workshops [10]. The goal of the workshops was to understand the foundations of MDA, to share experience in applying MDA techniques and tools, and to outline future research directions.

An interesting idea [11] consists of introducing an empirical study of the evolution of PHP MVC framework, this paper discusses the MVC based most famous PHP frameworks, evaluate their performance.

As the presented approach, these works dealt with modeling different systems using the MDA approach, but none of the above considered the use of MDA with web applications based on PHP frameworks. The developed models will be the first step into modeling web based application with the MDA approach.

5 Conclusions

The solution presented in this paper allows modeling web applications based on the CodeIgniter PHP. For this, we used the MDA approach, which provides the added advantage of improving application quality and portability, while significantly reducing costs and time-to-market. This approach has demonstrated its efficiency through enabling to build a complete model describing sufficiently a MVC CodeIgniter application.

The transformation from the UML model to the CodeIgniter model will be the subject of a study thereafter.

Afterwards, we will consider applying the solution of the MDA approach to other PHP frameworks in order to propose a comparative study.

References

1. Bergmann, S., Kniesel, G.: GAP: generic aspects for PHP. In: EWAS’06 (2006)
2. Object Management Group, Model Driven Architecture (MDA), MDA Guide rev. 2.0. http://www.omg.org/cgi-bin/doc?ormsc/14-06-01/
3. Mbarki, S., Erramdani, M.: Towards automatic generation of MVC2 web applications. INFOCOMP **7**, 88–91 (2008)
4. Blanc, X.: MDA en action: Ingénierie logicielle guidée par les modèles. Eyrolles, Paris (2005)
5. Roubi, S., Erramdani, M., Mbarki, S.: Model-Driven Transformation for GWT with Approach by Modeling: from UML Model to MVP Web Applications. In: MEDICT (2015)
6. CodeIgniter Documentation Website. https://www.codeigniter.com/docs/
7. Pitt, C.: Pro PHP MVC. Apress (2012)
8. Esbai, R., Erramdani, M., Mbarki, S., Arrassen, I., Meziane, A., Moussaoui, M.: Transformation by Modeling MOF 2.0 QVT: from UML to MVC2 Web model. INFOCOMP **10**, 01–11 (2011)
9. Arrassen, I., Meziane, A., Esbai, R., Erramdani, M.: QVT transformation by modeling. IJACSA **2**, 07–14 (2011)
10. Assmann, U., Aksit, M., Rensink, A.: Model Driven Architecture: European MDA Workshops: Foundations and Applications, Linkoping. Springer, Sweden (2005)
11. Olanrewaju, R., Islam, T., Ali, N.: An Empirical Study of the Evolution of PHP MVC Framework. In: Advanced Computer and Communication Engineering Technology (2015)