A Model-Driven Engineering Technique for Developing Composite Content Applications*

Moharram Challenger[1], Ferhat Erata[2], Mehmet Onat[3], Hale Gezgen[4], and Geylani Kardas[5]

1 **R&D Department, UNIT IT R&D Ltd., Izmir, Turkey, and International Computer Institute, Ege University, Izmir, Turkey**
moharram.challenger@unitbilisim.com
2 **R&D Department, UNIT IT R&D Ltd., Izmir, Turkey, and International Computer Institute, Ege University, Izmir, Turkey**
ferhat.erata@unitbilisim.com
3 **R&D Center, Koçsistem Information and Communication Services Inc., Üsküdar/Istanbul, Turkey**
mehmet.onat@kocsistem.com.tr
4 **R&D Center, Koçsistem Information and Communication Services Inc., Üsküdar/Istanbul, Turkey**
hale.gezgen@kocsistem.com.tr
5 **International Computer Institute, Ege University, Izmir, Turkey; and R&D Center, Koçsistem Information and Communication Services Inc., Üsküdar/Istanbul, Turkey**
geylani.kardas@ege.edu.tr

Abstract

Composite Content Applications (CCA) are cross-functional process solutions built on top of Enterprise Content Management systems assembled from pre-built components. Considering the complexity of CCAs, their analysis and development need higher level of abstraction. Model-driven engineering techniques covering the use of Domain-specific Modeling Languages (DSMLs), can provide the abstraction in question by moving software development from code to models which may increase productivity and reduce development costs. Hence, in this paper, we present MDD4CCA, a DSML for developing CCAs. The DSML presents an abstract syntax, a concrete syntax, and an operational semantics, including model-to-model and model-to-code transformations for CCA implementations. Use of the proposed language is evaluated within an industrial case study.

1998 ACM Subject Classification D.1.7 Visual Programming, D.2.6 Programming Environments, Graphical environments, D.2.11 Software Architectures, Domain-specific Architectures

Keywords and phrases Domain-specific modelling languages, composite content applications, model transformation, code generation

Digital Object Identifier 10.4230/OASIcs.SLATE.2016.11

1 Introduction

An enterprise content management system (ECM) organizes documents, contacts and records related to the processes of a commercial organization [3]. ECM aims to make the management

* This work is financially supported by the Scientific and Technological Research Council of Turkey (TUBITAK) Technology and Innovation Funding Programs Directorate (TEYDEB) under grant no. 3110712.

of corporate information easier through simplifying storage, security, version control, process routing, and retention. However, considering all capabilities and components of ECM, its architecture is rather huge and complex. Composite Content Applications[1] (CCA) are cross-functional process solutions (built on top of an ECM system) assembled from pre-built components, e.g. an integration of a Customer Relation Management (CRM), Forms, ECM, and Business Process Modelling (BPM). This integration adds more structural complexity to the system architecture.

Considering this complexity, CCA analysis and development needs new domain engineering and software development techniques. One possible approach to cope with this complexity is to increase the abstraction level using models [6] [8], in other words applying Model Driven Engineering (MDE), which moves software development from code to models [11] and may increase productivity [5] and reduce development costs [13]. One of the approaches to realize MDE is developing a Domain-specific Modelling Language (DSML) [7]. A DSML allows end-user programmers (domain experts) to describe the essence of a problem with abstractions related to a domain specific problem space.

In this paper, we present MDD4CCA, a domain specific modelling language for composite content applications. The DSML covers abstract syntax, concrete syntax, and operational semantics (including model-to-model and model-to-code transformations). In this study, the target platform is Microsoft Sharepoint. Using MDD4CCA tool, a user can model the system from various aspects, such as Form, Navigation, Content, and Workflow. As result the functional architectural code will be generated by the tools which is fully functional in target platform. Furthermore, the proposed language is evaluated using a real industrial use case and the results are reported in this paper.

There are some frameworks in industry to address the difficulty of CCA development, such as Nintex[2], AgilePoint[3], and K2[4] Frameworks. Nintex is a workflow software to automate business processes. AgilePoint enables business users to manage their processes. K2 is a platform for creating business applications and provides forms and workflows. However, there is no language behind any of these frameworks. So, they can only convert the model to code in a idiosyncratic way and they offer no semantics check and constraint control possibilities. A DSML, not only provides the concrete syntax for elements in the language, but also it presents abstract syntax including meta elements and their relations. Moreover, DSML allows providing controls both in modelling and interpretation time. Having these capabilities was our main motivations for developing MDD4CCA. In addition, the above-mentioned frameworks mainly stress on Forms and Workflow views. However, in our approach MDD4CCA uses various views in the form of different packages namely, Content, Navigation, Workflow, Form, and User packages.

Rest of the paper is organized as follows: In Section 2, the methodology applied in developing MDD4CCA is presented. In the next section, development of MDD4CCA is elaborated. Section 4 presents the industrial use case in which MDD4CCA is used to develop the software. Finally, the paper is concluded in Section 5.

[1] http://www.gartner.com/it-glossary/composite-content-applications-ccas/
[2] http://www.nintex.com/
[3] http://agilepoint.com/
[4] http://www.k2.com/

2 Methodology

To perform this study, we fulfilled three main steps: Requirement analysis, development of the language and finally its application in a case study. The overall procedure of steps one and two are depicted in Figure 1 and the case study is discussed in Section 4. The DSML development procedure includes the problem space (PS) modelling, solution space (SS) modelling, and solution space code generation [10]. PS covers issues for the modelling the problem and related tools independent of the details of the underlying frameworks, while SS deals with the modelling and code generation for the target framework.

Considering the requirement engineering for MDD4CCA, initially a concept dictionary is provided including the terms of a CCA. We have used a feature model [9] as a formal method to represent the specifications. As a tool to realize this, we have used Clafer [1] with which it is possible to perform model checking. After checking the feature models, the feature model is transformed into a meta-model in EMF [12] automatically by transformation rules given in Xtend [2] and Java. The meta-model is divided into four viewpoints including: Content, Form, Navigation, and Workflow.

The provided PS meta-model in EMF is the main input by which problem space modelling and its tool are provided. We developed a graphical editor in GMF[5] with which an end-user can model a problem according to the related business domain. Also, we provided Form model with an endogenous model to model transformation which is required for a composite content model. Using the Form model, we generate entity data models using bidirectional transformations which provides round-trip functionality and any modification in each side can be converted to the other side automatically. In the solution space modelling, the entity data model is generated.

Finally, from entity data model, entity classes are generated by using entity data model tool. On the other hand, from solution space models, the codes for server side pages, and other required files such as XML files are generated by model to text transformations via Xpand[6], Xtend, and Java. With a similar transformation, the required library code is configured based on the solution space model.

3 Development of MDD4CCA

As depicted in Figure 1, MDD4CCA development started with the requirement engineering providing a concept dictionary including terms (from YAWL[7], UWE[8], BPMN[9], and so on) and their relations with other concepts. The dictionary includes 537 concepts which are used as CCA requirements. To present these specifications in a formal way, we mapped the requirements into feature model. Also, to cope with the complexity and size of the resulted feature model (considering large number of concepts and their inter-relations), the models are divided into several viewpoints. It is worth noting that the User view is integrated in Content model. We used Clafer to implement the feature model with which we can also use Alloy[10] to do model checking. In this way, a clean room software engineering is realized with preventing some defects in the requirement engineering level using Alloy model checking

[5] http://www.eclipse.org/gmf-tooling/
[6] http://wiki.eclipse.org/Xpand
[7] http://www.yawlfoundation.org/
[8] https://en.wikipedia.org/wiki/UML-based_Web_Engineering
[9] http://www.bpmn.org/
[10] http://alloy.mit.edu/alloy/

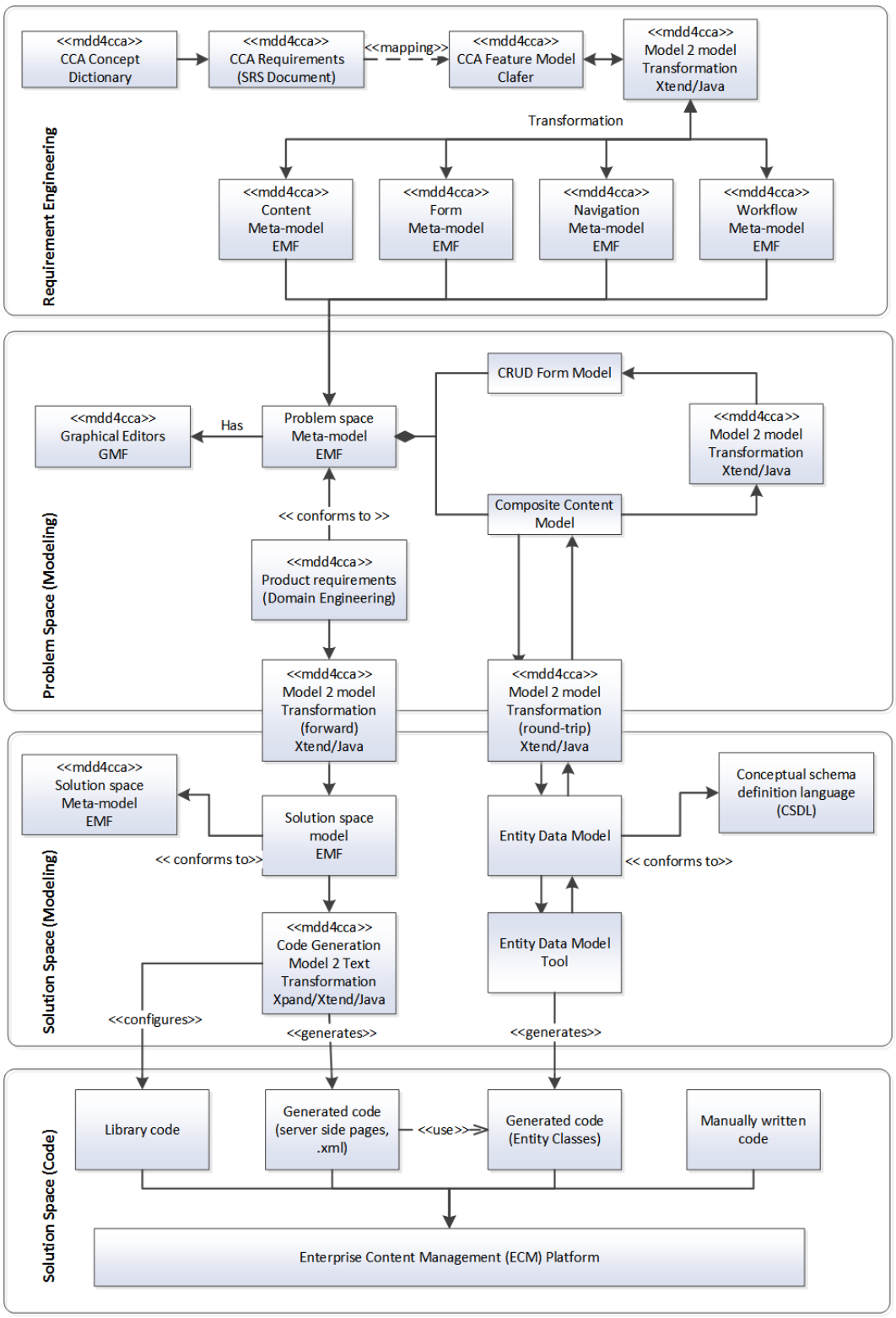


Figure 1 The process of analysing and developing MDD4CCA.

on feature model. Then, we transform feature models in Clafer format (XML-based) to meta-model [4].

3.1 Problem Space Modelling

Modelling of PS addresses the problem independent of details inside the artefacts of target framework. Based on the requirement analysis, the problem space modelling covers the domain with four viewpoints discussed in the previous section. Considering the space limitations of this paper, we focus on Content viewpoint which includes User viewpoint and also can generate main forms of Form viewpoint. Of course, an end-user's custom forms will be added to the Form viewpoint to complete the modelling.

To develop a DSML for modelling the PS, i.e. MDD4CCA, there is need for an abstract syntax. We generated the required meta-model automatically from the PS feature model. Part of this meta-model, related to Content Viewpoint, is depicted in Figure 2. This meta-model is used to provide the graphical editor for the Content viewpoint in MDD4CCA.

As can be seen in this figure, the features and the hierarchy structure in the feature model is transformed to the meta-elements and the relations between them in the meta-model. For example, a Web item can have several sub Web items and each of them can have Pages, Lists, SiteCollections and so on.

In addition to the abstract syntax, the graphical concrete syntax is provided for the MDD4CCA's graphical editor. To this end, we mapped the abstract syntax elements of MDD4CCA to the graphical notations. In this study, we have used Eclipse GMF to provide the graphical editor of the DSML. However, the complexity of configuring GMF forced us to use a higher level tool on top of GMF called Eclipse Epsilon[11] with which we could generate GMF components from an Ecore like language called EMF and a tool called Emfatic[12]. As result, we have provided a fully functional tool for MDD4CCA.

MDD4CCA's syntax tools can impose some restrictions/controls during the user's modelling. One part of these controls comes from the PS meta-model and the remaining originates from the graphical tool itself. These constraint controls help the end-user to design an accurate model by presenting mistakes such as wrong element connection, avoiding empty attribute, controlling number of required relations for a specific element, and so on. These constraints are implemented in Epsilon Validation Language (EVL)[13]. Part of the contraint control for Content Type is given in Listing 1.

3.2 Solution Space Modelling

There is a difference between modelling a CCA independent of the underlying frameworks (PS) and its modelling based on the platforms (SS). Therefore, we needed to separate level of modelling by providing two different meta-models for problem and solution spaces. To prepare a meta-model for SS, we analysed the commonality and variability. The result was a meta-model including elements from different related technologies, e.g. Entity Framework, Server Side pages and Forms, Cache layer, and object oriented language concepts.

Finally, a (instance) model of solution space meta-model is transformed to a platform specific code (or API) which is discussed in the next section.

[11] http://www.eclipse.org/epsilon/
[12] http://www.eclipse.org/epsilon/doc/eugenia/
[13] http://www.eclipse.org/epsilon/doc/evl/

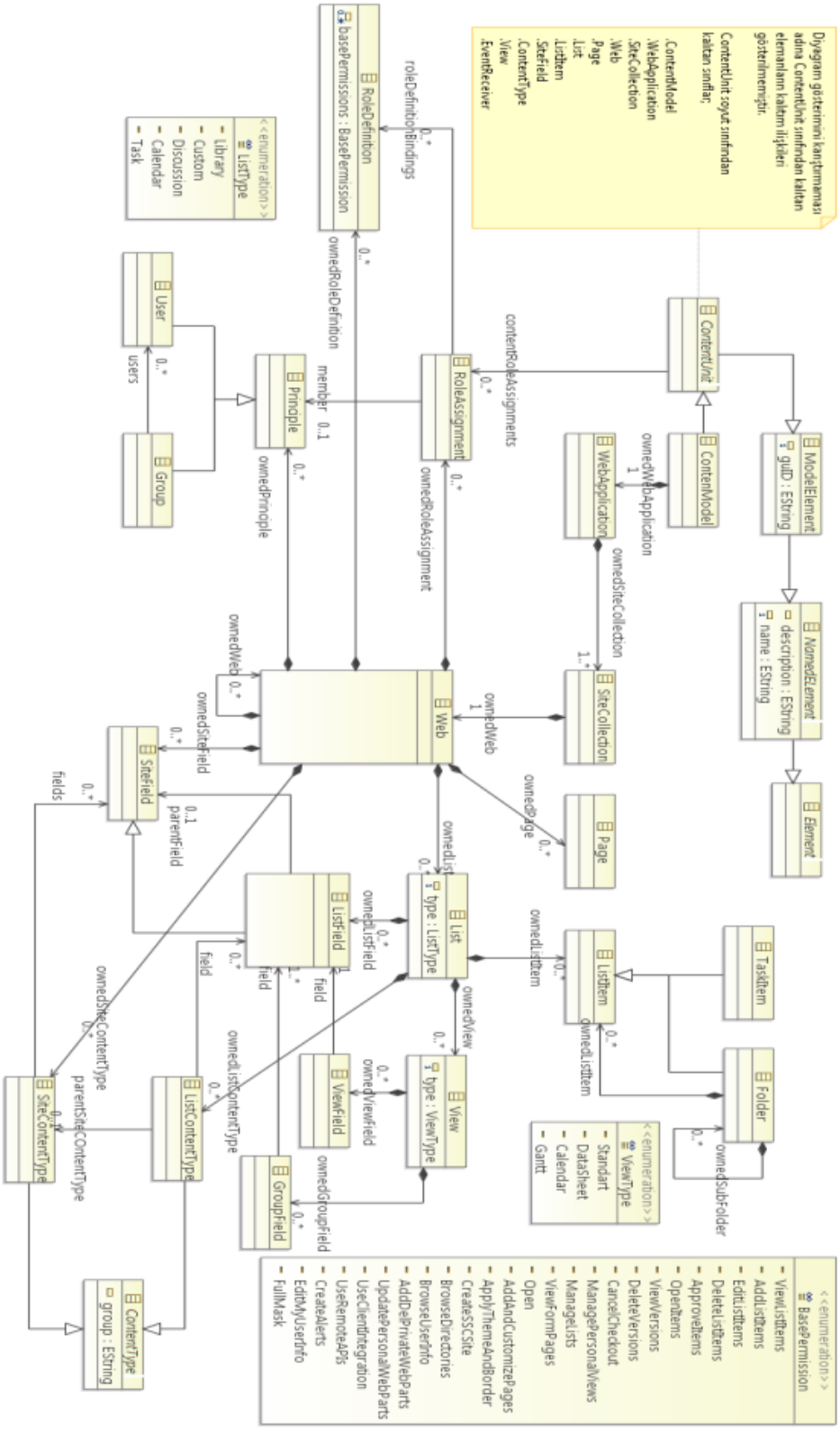


Figure 2 The Abstract Syntax of the Content Viewpoint.

Listing 1 Constraint control for Content Type element in EVL.

```
import ecore : 'http://www.eclipse.org/emf/2002/Ecore#/';
package modelgen : modelgen =
 'http://www.mdd4cca.com/msf/modelgen/Modelgen'{
 ...
 class File{
  attribute name : String[?];
  attribute extension : String[?] = '.cs';
  attribute folderPath : String[?];
 }
 class EnumMProperty extends MProperty{
  property enumType : EnumClass[?];
 }
 class MNavigationProperty{
  attribute name : String[?];
  property type : Cache[?];
  attribute multi : Boolean[?];
 }
 class MClass extends File{
  attribute usings : String[*] { ordered };
  attribute namespace : String[?];
  attribute constructorBody : String[*] { ordered };
 }
 ...
}
```

Table 1 Model Transformations in MDD4CCA.

Direction	*Source MM*	*Target MM*	*Type*	*# of Rules*
Forward	Problem Space	Problem Space	M2M	38
Forward	Problem Space	Solution Space	M2M	71
Forward/Backward	Problem Space	Microsoft CSDL[14]	M2M	24
Forward	Solution Space	code and xml files	M2C	204 Xpand templates

3.3 Model Transformations

It is not sufficient to complete a DSML definition by only specifying the notions and their representations. The complete definition requires that one provides semantics of language concepts in terms of other concepts whose meanings are already established. In this study, four types of transformations are fulfilled to realize the mentioned methodology, see Table 1. These transformations are Clafer to Ecore transformation (in requirement engineering phase), Content viewpoint to Form viewpoint transformation (in problem space modelling phase), problem space model to solution space model transformation, and finally solution space model to target platforms' code transformation. These transformations are used to respectively generate CRUD forms from content models; weave and translate content and form models into a solution space model; transform MDD4CCA content model (EDM) from/to EDMX (Entity Framework); and code generation from the SS models. The transformations are implemented using Java and Xpand in an integrated way.

4 Industrial Use Case: TUPRAS TPY Project

Taking into account the high cost of developing a DSML, there is a need to have the Return On Investment (ROI), balancing the expected benefits and productivity improvement against the cost of development and future maintenance of the tools before moving to a new development environment. Therefore, in this section we present one of the industrial projects which is implemented using MDD4CCA for one of the corporate customers of UNIT Company. The purpose of this section is to briefly report our experiences of developing a DSML based on the use case.

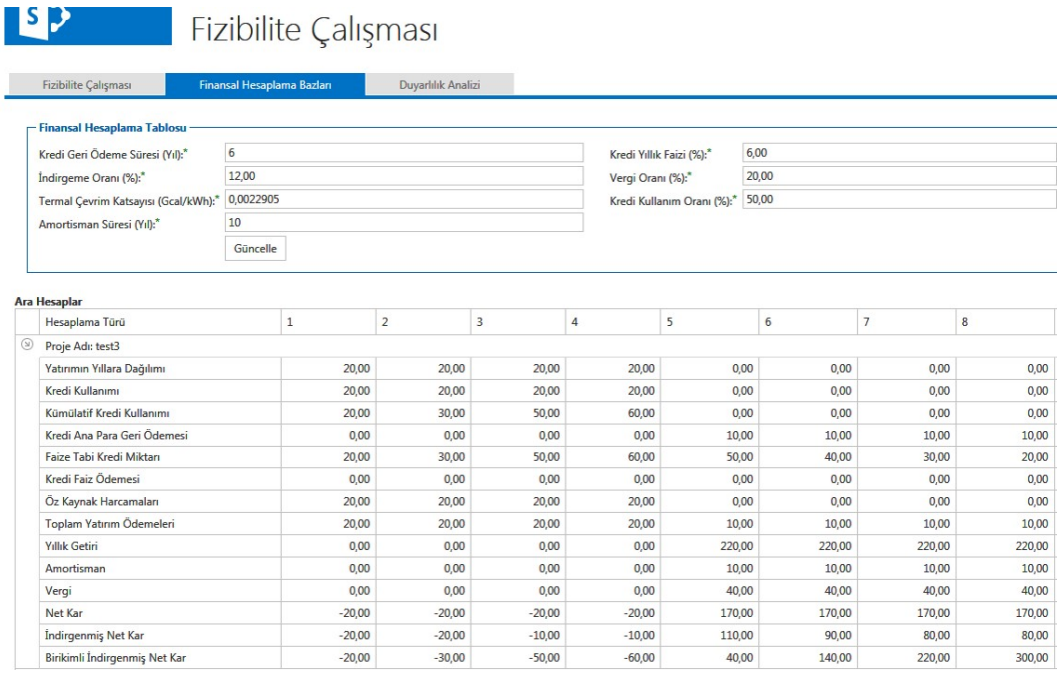
The industrial project discussed here is called TPY (acronym for the Turkish translation of "Tupras Project Management"). Turkish Petroleum Refineries Corporation (TUPRAS)[15] is currently the biggest enterprise in Turkey according to Turkey's Fortune 500 list[16]. TPY project is for managing the newly defined projects such as designing a new refinery in Tupras. The use case is modelled in MDD4CCA in 4 viewpoints (Content, Form, Fork-flow, and Navigation) with many diagrams. For example, there are several diagrams modelled for Form viewpoint considering the forms in different parts of the project, such as Activity form, Feasibility form, and so on.

After modelling the project in different viewpoints with diagrams for each part of the project using MDD4CCA, the transformations are applied using transformation engine of MDD4CCA and other models are generated from which the architectural code is generated. By adding delta code with developers, the system is fully functional. Altogether, this project has 79 work-flows, 195 transitions in work-flows, 43 roles, 137 tasks, 107 tables of databases, and 403 web pages for forms. The project has several important parts such as pre-feasibility, feasibility, pre-discovery, discovery, yearly investment planing, and Progress. For instance the screenshot of automatically generated form of the TPY's "Feasibility Page" is shown in Figure 3. Due to the confidentially issues, only the general parts are demonstrated in the figure. Besides, both the interface and the content of the form are in Turkish since Tupras aimed at using TPY only nation-wide at this stage. In the figure, feasibility study of a new project inside the refinery is considered. In the active tab given in the figure, some of the financial calculation values such as yearly interest for the credit, tax ratio, yearly distribution of the new project investments are given inside the form which is achieved by the execution of the MDD4CCA transformations.

In this case study, totally 96 model to model transformation rules and 151 model to code transformation templates are used. The model transformation from the problem space to the solution space is fully transformed automatically. In the scope of this case study, most of the code also is generated automatically. The resulting architectural code is functional in the target platform. Considering the underlying web-based content management platform, the language generated around 65% of the code. The other 35% of the code is in fact the part which has high variability among the projects.

The resulting software product has been used in TUPRAS since mid 2014 (about 2 years until the time this paper was prepared). The software is used by the staff from different TUPRAS departments. 120 workers from the project management departments of four TUPRAS refineries are using the software. In addition, 80 engineers are benefiting from the modeling environment of the given tool during project proposal preparation.

[15] http://www.tupras.com.tr/masterpage.en.php
[16] http://aa.com.tr/en/economy/tupras-tops-turkey-s-fortune-500-list/30989



Hesaplama Türü	1	2	3	4	5	6	7	8
Proje Adı: test3								
Yatırımın Yıllara Dağılımı	20,00	20,00	20,00	20,00	0,00	0,00	0,00	0,00
Kredi Kullanımı	20,00	20,00	20,00	20,00	0,00	0,00	0,00	0,00
Kümülatif Kredi Kullanımı	20,00	30,00	50,00	60,00	0,00	0,00	0,00	0,00
Kredi Ana Para Geri Ödemesi	0,00	0,00	0,00	0,00	10,00	10,00	10,00	10,00
Faize Tabi Kredi Miktarı	20,00	30,00	50,00	60,00	50,00	40,00	30,00	20,00
Kredi Faiz Ödemesi	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Öz Kaynak Harcamaları	20,00	20,00	20,00	20,00	0,00	0,00	0,00	0,00
Toplam Yatırım Ödemeleri	20,00	20,00	20,00	20,00	10,00	10,00	10,00	10,00
Yıllık Getiri	0,00	0,00	0,00	0,00	220,00	220,00	220,00	220,00
Amortisman	0,00	0,00	0,00	0,00	10,00	10,00	10,00	10,00
Vergi	0,00	0,00	0,00	0,00	40,00	40,00	40,00	40,00
Net Kar	-20,00	-20,00	-20,00	-20,00	170,00	170,00	170,00	170,00
İndirgenmiş Net Kar	-20,00	-20,00	-10,00	-10,00	110,00	90,00	80,00	80,00
Birikimli İndirgenmiş Net Kar	-20,00	-30,00	-50,00	-60,00	40,00	140,00	220,00	300,00

Figure 3 The running application of TPY use case (Feasibility Page).

5 Conclusion and Future Work

In this paper, a domain specific modelling language, called MDD4CCA is developed for Composite Content Applications. The language covers all model driven components including the abstract syntax, the concrete syntax, model to model and model to code generation. The language is used in the development of an industrial project which is reported as a case study again in this paper. The result shows that utilization of the proposed model-driven technique and MDD4CCA, the 65% of the application code is generated automatically in average. Our next work is to continue the evaluation of the DSML by using it for the development of new real industrial CCAs.

References

1 Kacper Bąk, Zinovy Diskin, Michał Antkiewicz, Krzysztof Czarnecki, and Andrzej Wąsowski. Clafer: unifying class and feature modeling. *Software & Systems Modeling*, pages 1–35, 2014.

2 Lorenzo Bettini. *Implementing Domain-Specific Languages with Xtext and Xtend*. Packt Publishing, 2013.

3 Bob Boiko. *Content Management Bible*. John Wiley & Sons, 2005.

4 Ferhat Erata, Moharram Challenger, Serhat Gezgin, Argün Demirbaş, Mehmet Önat, and Geylani Kardas. A methodology for supporting the synchronization between capability models and metamodels in software product lines. In *8th Turkish National Software Engineering Symposium*, volume 1221, pages 2–13, 2014.

5 Tomaž Kos, Tomaž Kosar, Jure Knez, and Marjan Mernik. From DCOM interfaces to domain-specific modelling language: A case study. *Computer Science and Information Systems*, 8(2):361–378, 2011.

6 Ivan Lukovic, Vladimir Ivancevic, Milan Celikovic, and Slavica Aleksic. DSLs in action with model based approaches to information system development. In Marjan Mernik, editor, *Formal and Practical Aspects of Domain-Specific Languages: Recent Developments*, pages 502–532. IGI Global, 2013.

7 Marjan Mernik, Jan Heering, and Anthony M. Sloane. When and how to develop domain-specific languages. *ACM Computing Surveys*, 37(4):316–344, December 2005.

8 Aleksandar Popovic, Ivan Lukovic, Vladimir Dimitrieski, and Verislav Djukic. A DSL for modeling application-specific functionalities of business applications. *Computer Languages, Systems and Structures*, 43(C):69–95, October 2015. doi:10.1016/j.cl.2015.03.003.

9 Roger Pressman. *Software Engineering: A Practitioner's Approach.* McGraw Hill, 2000.

10 Awais Rashid, Jean-Claude Royer, and Andreas Rummler. *Aspect-Oriented, Model-Driven Software Product Lines - The AMPLE Way.* Cambridge publications, 2011.

11 Douglas C. Schmidt. Guest Editor's Introduction: Model-Driven Engineering. *Computer*, 39(2):25–31, February 2006.

12 Dave Steinberg, Frank Budinsky, Marcelo Paternostro, and Ed Merks. *EMF: eclipse modeling framework.* Pearson Education, 2008.

13 Antonio Vallecillo. A journey through the secret life of models. In Uwe Aßmann, Jean Bézivin, Richard Paige, Bernhard Rumpe, and Douglas C. Schmidt, editors, *Perspectives Workshop: Model Engineering of Complex Systems (MECS)*, number 08331 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2008. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany.