Building CMS-based Web Applications Using a Model-driven Approach

Feliu Trias
Kybele Research Group
Rey Juan Carlos University
Tulipán S/N, 28933, Móstoles, Madrid, Spain.
feliu.trias@urjc.es

Abstract— **In recent years Content Management Systems (CMSs) have increased their presence in organizations and businesses thanks to their efficiency for managing digital content. Specifically, Web applications provided by these organizations have started to use CMS as the core system to support them. For this reason, the CMS-based Web applications have gained popularity rapidly. Despite this fact, current Model Driven Web Engineering (MDWE) methods do not fit well into the CMS domain. Accordingly, their modeling languages lack of expressiveness to represent and capture the key elements needed to develop this kind of Web applications. To address this issue we propose a CMS Common Metamodel that captures the key concerns required to model and implement CMS-based Web applications. This metamodel can be used to extend the modeling languages used in current MDWE methods as well as serving as the base of new modeling languages specified within the CMS domain.**

Keywords-web engineering; model driven engineering; content management system

I. MOTIVATION

Over the last years, the World Wide Web has evolved towards a platform for sophisticated enterprise applications and complex business processes [1]. Organizations rely on the Web to support their business processes and use Internet as a way to create competitive advantage, global collaboration and integration with external partners [2]. In accordance, the number of Web applications developed by industry has increased dramatically [3]. In parallel with the growth of the Internet, content volumes of digital content have grown rapidly. Organizations experienced the need of using strong management tools to maintain their large Web applications and manage all their content [4] [2]. One of the most popular adopted solutions has been the Content Management System (CMS)-based Web Applications [5]. These applications allow users to collect, manage and publish content in an integral way. Moreover, such systems help organizations stay organized so that their Web applications can grow and evolve quickly while maintaining high quality [6].

Since ten years ago, the Web Engineering community has been evolving to provide better proposals, methods and techniques to improve the efficiency of Web application development processes [7].

Most of these Web Engineering methods, such as WebML [8], OOH [9], UWE [10], OOHDM [11] or HM^3 [12], encourage the use of a Model-Driven Engineering (MDE) to improve the development. MDE use models as a primary artifact in the development process. Thus it is possible to address platform details using MDE technologies that combine: modeling language that reflects the structure, behavior and requirements of a given domain, with transformation mechanisms that transform models into different implementations respectively [13].

It is quite difficult to find a Model Driven Web Engineering (MDWE) method that fits well for every Web application domain. For this reason these methods need to be adapted, extended or redefined to different contexts. For instance, with the arrival of the Rich Internet Applications the existing methods did not have enough expressiveness to model this new domain [14]. Therefore, most of the MDWE methods, such as WebML, UWE, OOHDM or OOH have been redefined in order to address the development of this kind of Web applications [15].

Currently, MDWE methods are not suitable for the particular CMS-domain since they do not consider the specific aspects of this kind of Web applications. Some of these aspects are: strict separation between content, structure and graphical design, a content repository for the reuse of information, integrated workflow for structuring the process of creation and publication of information [2], extensibility and modularity, support several types of contents and dynamic manage of layout and visual appearance [5]. One of the most critical drawbacks is that their modeling languages lack expressiveness to define CMS-based Web applications [3].

To overcome this problem our work CMS Common Metamodel that captures the key concepts for the development of CMS-based Web application. It can be used as a common metamodel base to adapt or extend the modeling languages of current MDWE methods as well as to define new modeling languages for new methods specialized in the context of the development of CMS-based Web applications.

To analyze the CMS domain and define the CMS Common Metamodel we captured the principal elements of the three most popular open-source CMS platforms in the market currently [16]: Drupal [17], Joomla! [18] and Wordpress [19].

978-1-4577-1938-7/12/$26.00 ©2011 IEEE

These platforms provide the implementation of Web applications improving development time and stability [1].

The rest of the work is organized as follows: section II presents the problem statement. In section III we present our research methodology. In section IV we explain our progress, so far. In section V we consider the related works and in section VI we present the conclusions.

II. PROBLEM STATEMENT

This work is centered on the modeling and automatic generation of CMS-based Web applications from the MDE perspective.

In the late 1990s, web site complexity exploded along multiple dimensions: higher volumes of digital content, more dynamic content, higher numbers of visitors, and increasingly complex supporting hardware and software [4]. Therefore, Content Management Systems (CMS) became essential for all organizations with a significant Web presence and huge volumes of digital content [4].

Many organizations bet on CMS to base their Web applications. These CMS-based Web applications allow these organizations to create and deploy digital content to Web based audiences: partners and staff accessing Web content via intranet, extranet and Internet [4].

In this sense, it creates an interesting scenario that should be considered by the Web Engineering community. However, despite the widely use in CMS-based Web application, MDWE methods still do not consider this emerging CMS domain [3]. In order to solve this gap, in this work we consider necessary to analyze this CMS domain defining a metamodel that captures the key elements, their attributes and their relationships.

This metamodel could extend the existing MDWE methods and qualify them for modeling and generating this kind of Web applications. Moreover, it could be used as the base of new MDWE methods specialized in developing CMS-based Web applications.

In this work we present the CMS Common Metamodel that defines this CMS domain. To build it we have considered the most used CMS platforms currently on the market for the development of CMS-based Web applications [16]: Drupal, Joomla! and Wordpress.

The following sections explain the current state of MDWE Methods and present the specific features provided by CMS-based Web applications.

A. Model-Driven Web Engineering Methods

Since the development of software systems in Internet appeared, the research community has detected the necessity of proposing new methodologies, techniques and models to offer a suitable reference environment for the new and special characteristics of Internet [20]. For this aim, a new research line in the Software Engineering has been developed in the last years: Web Engineering [7].

The Web Engineering community provides with several Web Engineering methods, specifically conceived for the development of Web applications. Most of them, such as OOH, UWE, OOHDM and HM^3, are based on the MDE paradigm.

Most of these MDWE methods propose the use of different views of Web systems following a horizontal separation of concerns [21] to model and develop Web applications: the content, navigation, process and presentation views. As is stated in [15]: 1) content is the view used to represent the business and data objects, their properties and methods, and their relationships 2) navigation is the view which expresses the composition of the interface in terms of containers and content the navigation, that is, the mechanisms for user's interaction through links and form submission, and the business logics, expressing the update of the domain objects and the invocation of arbitrary operations 3) process, is the view that defines how the application reacts to the events raised by the user's navigation and 4) presentation is the view for specifying the layout and the look & feel of the interface, as well as the widgets that enable user's interaction.

All of these MDWE methods are provided with modeling languages that are specialized to model concepts within a given domain by means of notations and abstractions [1]. These required concepts are included in metamodels and grouped using a set of views. For instance, UWE provides a modeling language based on a metamodel that is a "profileable" extension of the UML 2.0 metamodel. It is structured into six views similar to those established by Web Engineering, i.e. requirements, content, navigation, process, presentation and adaptation [22].

B. CMS-based Web Applications

This kind of Web applications present a set of specific characteristics that differ from traditional Web applications. Some of these features according to [6] and [4] are:1) to assume a large, very well organized team producing content that will be moved onto pages in bulk. The responsibility of content deployment has moved from the webmaster to the actual authors within the business (customers, suppliers, partners and staff), 2) the identification and administration of user roles and identification of groups of users 3) components and content are the things to be managed not pages. Types of content include HTML, XML, images, videos, documents and dynamic content, e-mails, catalogues, technical documents, audio and video, databases reports and e-commerce transactions, 4) this content is hold in repository and data source and 5) this content is categorized by using metadata. Also it is interesting to consider the aspects gathered in [23]: 6) extensibility and modularity, 7) independence between content and presentation, 8) dynamic management of layout and visual appearance and 9) support workflow definition.

We consider that some aspects such as the configuration of user permissions, the taxonomy of the data or the inclusion of external services are not well tacked by the existing MDWE methods within the CMS-based Web application domain [3].

Therefore, the objective of this work is to identify the set of common elements required to be modeled for the development of CMS-based Web applications. We have captured these elements in a common metamodel that we call CMS Common Metamodel.

Thus, it would be possible to build a CMS model conformed to this CMS Common Metamodel, considering as inputs the traditional models proposed by the current MDWE methods. From this CMS model it would be possible to develop a CMS-based Web applications supported by one of the three most popular CMS platforms: Drupal, Joomla! and Wordpress.

The different elements of the presented CMS Common Metamodel can be presented using four concerns: navigation, content, user and CMS behavior.

In the next sections we explain the strategy followed to define and implement the CMS Common Metamodel that we present.

III. PROPOSED METHODOLOGY

Next we present the methodology proposed to carry out this work. Figure 1 presents a general perspective presenting the two main phases to be executed in this research.

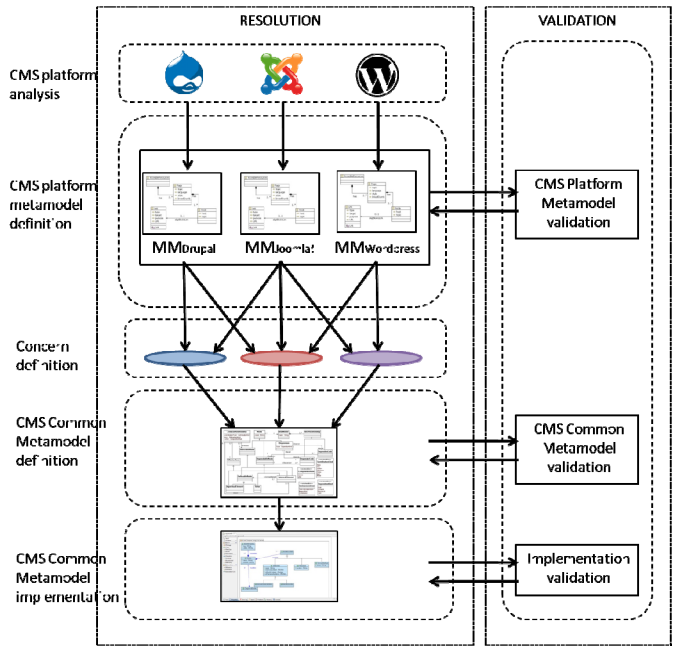

Figure 1. General view of the methodology

A. Resolution

The main aim of this phase is to define and to implement the CMS Common Metamodel that captures the key elements of the CMS domain. This phase is composed of five principal activities: CMS platform analysis, CMS platform metamodel definition, concern definition, CMS Common Metamodel definition and finally the CMS Common Metamodel implementation. Next, we explain each one of them in detail.

1) CMS platform analysis: this activity is centered in analysing the CMS market to obtain a better understanding of the subject. Thanks to this activity we conclude that currently the most popular and used CMS platforms are Drupal, Joomla! and Wordpress, all of them open-source platforms. After detecting them we analyzed each platform in detail.

2) CMS platform metamodel definition: after analyzing and getting familiar with these CMS platforms we defined a metamodel for each one of them which help us to understand and contol their domain. The three CMS platform metamodels identified the key elements of each domain and defined their attributes. Likewise, we determined the relationships between those elements.

3) Concern definition: after the identification and definition of the key elements and their relationships of each CMS platform domain, we defined a set of perspectives which allow us to group the elements of each CMS platform metamodel. In this work we call these perspectives as concerns.

4) CMS Common Metamodel definition: after defining the three CMS platform metamodels, we built the CMS Common Metamodel. To build it we decided to create the union of the three previous metamodels. Thus, the resulting CMS Common metamodel copes with the common elements and the specific elements of each platform to assure that our metamodel allow the modeling and the implementation of a CMS-based Web applicatioin based in any of these CMS platforms.

5) CMS Common Metamodel implementation: after defining the CMS Common Metamodel we implemented it in eXtensible Markup Language (XML) by using the Eclipse Modeling Framework (EMF)

B. Validation

In parallel to the activities of the resolution phase we carried out activities of validation. These activities are performed in the validation phase.

The three main activities of this phase are: CMS platform metamodel validation, CMS Common Metamodel validation and implementation validation. All these activities were performed in a cyclical way. All the errors and inconsistencies detected were corrected to create the next and improved version of all the metamodels as well as the implementation of the CMS Common Metamodel. Next, we explain briefly the activities carried out in this phase.

1) CMS platform metamodel validation: to validate each CMS platform metamodel we modeled a CMS-based Web application by using a model that captures the elements and relationships defined in each metamodel.

2) CMS Common Metamodel validation: just as we did with the validation of the CMS platfom metamodels we modeled another CMS-based Web application to validate the CMS Common Metamodel

3) Implementation validation: implementing the CMS Common Metamodel in EMF we could validate the implementation using the tool provided by this framework.

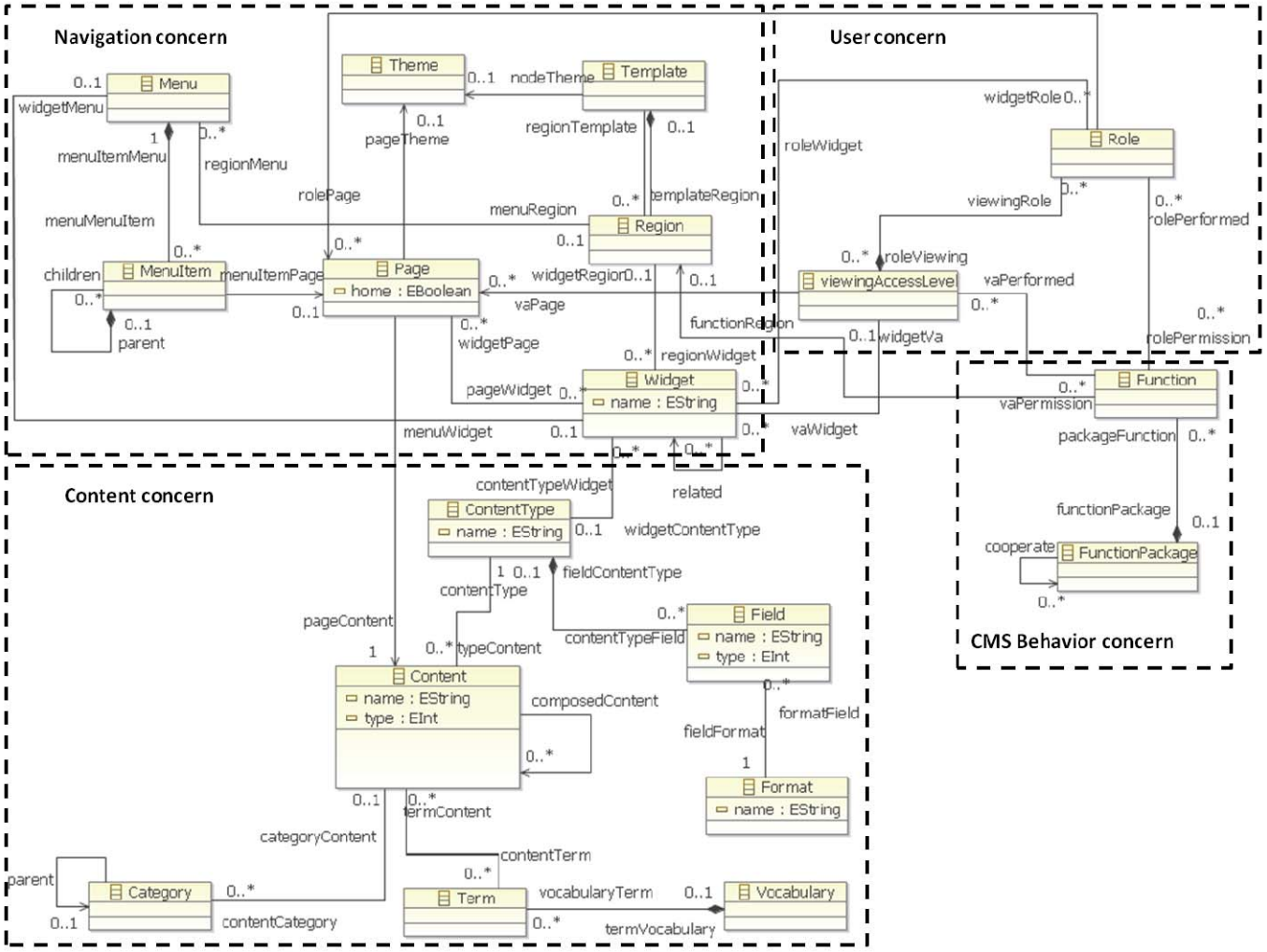

Figure 2. CMS Common Metamodel

To execute the three validation activities we used two case studies of different field and magnitude.

The first one was a Web application focused on the popularization of wellbeing. It allows user to consult information about health, sport and balanced diets. Moreover, they can participate in some polls about nutrition and health issues. Also, users can connect this Web application with their twitter account.

The second one is a real case study aimed at preventing fraud in the banking sector. It is divided in two parts: the former is a Web application that allows users to interact with a data base and to store a large amount of information about frauds (who committed the fraud, where it was committed, what type of fraud); the latter is a set of algorithms that using the information stored in the data base can to make a set of predictions to prevent future frauds.

IV. PROGRESS SO FAR

In this section we explain the CMS Common Metamodel presented in Figure 2. As we can see, the Metamodel is classified in four different concerns. These concerns help us to classify and manage the elements captured by the Metamodel. Next, we explain which kind of elements is contained within each defined concern. It is interesting to note that the proposed concerns match with the four layer hierarchy defined in [6]:

1) Navigation: this concern considers both the elements that define the navigational structure and those that allow the navigation among this navigational structure.

2) Content: it is the concern that captures the data and data type of the information managed by the CMS-based web application.

3) User: in this concern, all the elements related to roles of the users and their permissions are defined.

4) CMS Behavior: this concern contains the elements that allow the definition of the different functions performed by the CMS-based Web application.

It is important to remark that the CMS Common Metamodel results from the union of the three CMS platform metamodels. This union can be expressed as a formula:

$$(\bigcup_{i=1}^{n} MM_i) = MM_{CMS\ Common}$$

where *MMi* are the metamodels of the analyzed platforms and the $MM_{CMS\ Common}$ is the metamodel that we present.

Finally, we can conclude that each entity within the domain is defined in a metamodel. It is represented defining the following formula.

$$\forall d \in D(\exists m \in MM[map(m) = d])$$

where *map()* is the function that associates an element d of a domain *D* with the key element m of a metamodel *MM*.

V. RELATED WORKS

After witnessing the rise in popularity of CMS-based Web applications the MDE Community has expressed an increasing interest in the CMS domain. For this reason, in recent years MDWE methods specialized in the development of Web applications supported by CMS have emerged. Likewise, traditional MDWE methods have attempted to adapt their modeling languages to cope with the CMS-based Web applications development.

Next we explain some of the works related to the development of the CMS-based Web applications following the model-driven paradigm.

In the field of MDD there are several methods for the generation of Web applications. In contrary, model-driven methods specialized for generating CMS-based websites are not highly spread yet.

In [5] is presented a graphical language specialized in modeling CMS-based Web applications. This modeling language is structured in two levels of abstraction, CMS-ML (CMS- Modeling Language) and the CMS-IL (CMS-Intermediate Language). The former addresses the high-level specification of a typical Web application. It is simple enough to be used and learnt by non technical stakeholders (called business designers). The latter is a common low-level language for CMS platforms, deployed by the so-called system designer.

Both, captures the elements required to determine a set of models. For instance, the CMS-ML defines the key elements to build the template model and the toolkit model. The template model reflects the structure and behavior of the Web application and the toolkit model allows the addition of new modeling elements that can be used in the template model.

Their key elements are defined by means of an abstract syntax and a concrete syntax. Furthermore, they are classified in different views. For instance, the elements in the template model are established in eight views: structure view (macro and micro structure), navigation view, roles view, permission view, users view, language view, contents view and visual themes view. The two possible drawbacks we can consider in this approach are the lack of expressiveness to deal with concrete implementation details and the high number of views captured in a unique model.

In [3] is presented a research focused on the improvement and adaptation of the OOWS Method in the CMS Domain. The OOWS Method is evaluated to detect its limitations in the CMS domain. Following Situational Method Engineering (SME) [24] the OOWS metamodel is extended to integrate the CMS domain.

Finally, in [1] the Web Engineering Method (WEM), an approach for the automated configuration of CMS-based Web applications, is presented. As part of this approach, they also propose an abstract and concrete syntax to model this kind of websites.

VI. CONCLUSIONS

We conclude that many organizations bet on CMS-based Web applications because allow them to create and deploy effectively digital content to Web based audiences: partners and staff accessing Web content via intranet, extranet and Internet [4].

Despite the widely use of CMS-based Web applications, MDWE methods still do not consider this emerging CMS domain [3]. In order to solve this gap; in this work we have analyzed this CMS domain defining the CMS Common Metamodel that captures the key elements with their attributes and their relationships.

It is interesting to note that to define this Metamodel we considered the three most popular open-source CMS platforms in the market: Drupal, Joomla! and Wordpress. We built a metamodel to determine the domain of each platform and then we defined the CMS Common Metamodel as the union of those metamodels.

After defining the Metamodel, we specified four concerns (navigation, content, user and CMS behavior) to classify the elements captured in the CMS Common Metamodel. Although these concerns are similar to the views proposed by the Web Engineering community to develop and define the architecture of Web applications, they include specific elements of the CMS domain.

Finally, we propose to use the CMS Common Metamodel, to extend the existing MDWE methods and qualify them for modeling and generating CMS-based Web applications. Moreover, it could be used as the base of new MDWE methods specialized in developing CMS-based Web applications.

REFERENCES

[1] J. Souer, T. Kupers, R. Helms, and S. Brinkkemper. "Model-Driven Web Engineering for the Automated Configuration of Web Content Management Systems". in ICWE. 2009. Heidelberg.

[2] J. Souer, I. van de Weerd, J. Versendaal, and S. Brinkkemper, "Situational Requirements Engineering for the Development of Content Management System-Based Web Applications". International Journal of Web Engineering and Technology 2007. 3(4): p. 420 - 440

[3] K. Vlaanderen, F. Valverde, and O. Pastor, "Model-Driven Web Engineering in the Cms Domain: A Preliminary Research Applying Sme", in Enterprise Information Systems. 2009: Heidelberg. p. 226-237.

[4] S. McKeever, "Understanding Web Content Management Systems: Evolution, Lifecycle and Market". Industrial management + data systems, 2003. 103(9): p. 686 - 692.

[5] J. de Sousa and A. Rodrigues, "Web Application Modeling with the CMS-ML Language", in Inforum. 2010. p. 461-472.

[6] B. Boiko, "Understanding Content Management". Bulletin of the American Society for Information Science and Technology, 2001. 28(1): p. 8-13.

[7] Y. Deshpande, et al., "Web Engineering". Journal of Web Engineering, 2002. 1: p. 3-17.

[8] S. Ceri, P. Fraternali, and A. Bongio, "Web Modeling Language (WebML): A Modeling Language for Designing Web Sites.". Computer Networks, 2000. 33: p. 137-157.

[9] J. Gómez, C. Cachero, and O. Pastor. "Extending a Conceptual Modelling Approach to Web Application Design". in International Conference on Advanced Information Systems Engineering. 2000.

[10] N. Koch and A. Kraus. "The Expressive Power of UML-Based Web Engineering". in International Workshop on Web Oriented Software Techonology. 2002.

[11] D. Schwabe and G. Rossi, "The Object Oriented Hypermedia Design Model". Communications of ACM, 1995. 38: p. 45-46.

[12] P. Cáceres, D.C. V., V. J.M., and E. Marcos, "Model Transformations for Hypertext Modeling on Web Information Systems". SAC, 2006: p. 1232-1239.

[13] D. Schmidt, "Model-Driven Engineering". IEEE Computer, 2006(39).

[14] J.C. Preciado, M. Linaje, S. Comai, and F. Sanchez-Figueroa, "Necessity of Methodologies to Model Rich Internet Applications", in 6th International Symposium on Web Site Evolution. 2005.

[15] P. Fraternali, S. Comai, A. Bozzon, and T. G., "Engineering Rich Internet Applications with a Model-Driven Approach". ACM Transactions on the web, 2010. 4: p. 47.

[16] R. Shreves, "Open Source CMS Market Share", W.S. White paper, Editor. 2011.

[17] Drupal Cms. [cited 2011 25th july]; Available from: http://drupal.org/.

[18] Joomla! Cms. [cited 2011 5th june]; Available from: http://www.joomla.org/.

[19] Wordpress Cms. [cited 2011 10th june]; Available from: http://wordpress.org/.

[20] M.J. Escalona, J. Torres, M. Mejias, J.J. Gutierrez, and D. Villadiego, "The Treatment of Navigation in Web Engineering". Advances in Engineering Software, 2007. 38(4): p. 267-282

[21] A. Kraus, A. Knapp, and N. Koch, "Model-Driven Generation of Web Applications in Uwe". 2008.

[22] C. Kroiss and N. Koch, "The UWE Metamodel and Profile - User Guide and Reference". 2008, Ludwig-Maximilians-Universität München (LMU), Institute for Informatics: München.

[23] J. de Sousa and A. Rodrigues. "CMS-Based Web-Application Development Using Model-Driven Languages". in Fourth International Conference on Software Engineering Advances. 2009. Porto, Portugal: IEEE Computer Society.

[24] J. Ralyté, R. Deneckère, and C. Rolland. "Towards a Generic Model for Situational Method Engineering". in CAiSe 2003. 2003. Heidelberg: Springer.