

INTEROPERABILITY ISSUES OF MDWE METHODOLOGIES

ATTILA ADAMKÓ AND LAJOS KOLLÁR

ABSTRACT. Due to the evolution of Web technologies experienced in the past 10–15 years, the Web has become a primary platform for developing applications. However, as these technologies evolve very fast, they might become obsolete soon. Developers of Web applications need sophisticated solutions that support the whole product lifetime of an application that is able to cope with the skyrocketing changes of the underlying technologies.

Model-driven Web Engineering (MDWE) is a still emerging field aiming at providing sound model-based solutions for building Web applications that try to separate the abstract design (PIM) from the concrete technological platforms (PSMs). However, current MDWE approaches cannot provide solutions for all kinds of the requirements against a software system therefore a lightweight, extensible, loosely coupled set of models for designing applications are needed.

This paper introduces an approach for the interoperability of (some) existing methodologies based on metamodeling, model transformations and model weaving which allows the MDWE methodologies to be extended in a consistent manner where new model kinds are separated and weaved together with the classical models that each approach supports.

1. INTRODUCTION

Existing model-based Web Engineering approaches provide different methods and tools for both the design and the development of various kinds of Web applications. In order to reduce complexity, most of the methodologies propose the separation of different views (i.e., models) of the application into 3 levels: structural (or content), navigational (or hypertext) and presentational models. For more information see [9]. Figure 1 shows the most common design dimensions of the currently existing methodologies.

Received by the editors: April 10, 2011.

2010 *Mathematics Subject Classification*. 68U35, 68M11, 68N99.

1998 *CR Categories and Descriptors*. D.2.2 [**Software**]: Software Engineering – *Design Tools and Techniques*; D.2.10 [**Software**]: Software Engineering – *Design*.

Key words and phrases. MDWE, Web Applications, Metamodeling, Model transformations, Model weaving.

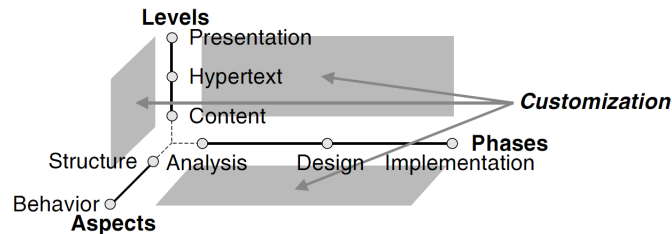


FIGURE 1. Design dimensions of Web applications [9].

In addition, some methodologies add some new models (or refine existing ones) to obtain a more fine-grained solution when modeling the application. Despite the separation, the levels should be interconnected in order to be able to capture the semantics behind the elements of the different models, e.g., the navigational objects are based on certain elements of the content model.

Beyond the creation of the models for the corresponding levels, Web application designers need to be aware of the various aspects of the systems to be modeled. Some applications are providing access to more or less static information hence they require much less behaviour modeling compared to systems that need to perform several complex business processes like e-commerce applications. Both structure and behaviour need to be modeled using a uniform notation that has to cope with the specific characteristic of each of the levels.

Current design methods offer some possibilities for modeling the levels and aspects mentioned above but they all has a unique approach (e.g., offering some model kinds that the others not) so this field is not standardized.

2. RESEARCH BACKGROUND

2.1. Domain-specific modeling, Metamodeling. The main goal of domain-specific modeling is to raise the level of abstraction by specifying the solution directly using domain concepts. The final product (and maybe several intermediate artifacts, as well) are generated based upon these high-level specifications. It also allows the stakeholders and domain experts to concentrate to the domain only. Domain-specific languages (DSLs) are built in order to capture domain semantics. A very common (but not the only) way of defining DSLs is metamodeling. The previously mentioned Web application design methods contain notations that can be used for describing a model of a Web application so they can be considered as DSLs for Web applications hence.

Some of the existing Web application design methods (e.g., UWE, WebML) offer a metamodel, as well [?, ?]. This allows model-based development since one need to build models conforming to the appropriate metamodel in order

to capture the structural, navigational or presentational structure of the application to be developed. However, in the most of the cases, these models mix the different levels of Web applications that results in a solution that might be appropriate for the given application domain but makes the reuse of models or model parts almost impossible.

2.2. Model transformation, Model weaving. Model transformations are the most important operations in model engineering, describing how elements in the source model are converted into elements in the target model. This is achieved by relating the corresponding metamodel elements in the source and the target metamodels. Transformations can be classified into two categories: vertical transformations (a.k.a. refinements) are defined between models of different abstraction levels (e.g., PIM—PSM mappings), while horizontal transformations are mappings between models of the same level of abstraction (e.g., for improving or correcting a model).

Weaving models are used to explicitly describe fine-grained relationships between models and metamodels (that are models themselves, as well) and execute operations based on them. With the help of applying weaving models, large metamodels that capture all aspects of a system can be avoided and a lattice of metamodels can be constructed instead where each metamodel that focuses on its own domain is maintained independently from the others. The links defined by the weaving model have some associated semantics about the linked elements.

3. PROBLEM STATEMENT

Most of the methods mentioned in Section 1 are using different notations for these models, hence the interoperability between them is very hard to achieve. This also decreases reuse as one cannot import, for example, a conceptual model or a part of it when developing an application for a similar domain.

The idea of complete integration of the existing languages and methodologies, i.e., developing a common metamodel and unified phases of development that everyone will use in the future is utopian and (in our opinion) it must not be the goal of any integration or interoperability efforts. The main reason behind it is that different domains and various flavours of Web applications may require different styles of modeling and it is almost impossible to achieve such a common modeling notation which is easy to understand and work with while being flexible enough to solve the uprising issues. Therefore we should work on bridging the different models together that allows (or promises, at least) the interchangeability of models and/or model pieces instead.

New models, processes and transformations should be included into the existing design methods when new aspects arise. However, these changes to a methodology are very risky and can cause several problems. In [6], three categories of concerns were identified:

- dependent concern, that depend on some other (earlier defined) concern(s), e.g., navigation (which depends on the conceptual model);
- replacement concern, that fully replaces a previously defined concern, e.g., presentation;
- orthogonal concern, that is a brand new concern which is completely independent of all the others, e.g. business process models.

However, we are not against the creation of subsequent metamodels and/or methodologies as they can result in better description of system parts or improved development processes. We only claim that a common metamodel is not the Holy Grail of MDWE as each and every “common” one will most probably fail as being a universal solution because the diversity of Web applications will require new answers for such questions that probably had not been asked by the time of developing the common metamodel.

4. PROPOSED SOLUTION

Our goal is to establish an extensible model-based framework which can provide interoperability among the existing Web modeling languages. This task has to be achieved by separating the different concerns (i.e., levels, phases and aspects) of Web applications in order to be able to either reuse relevant model parts or “transfer” a model into another notation (e.g., after a structural model is created conforming the metamodel of language A we decide to build the navigational model in language B since it might be more appropriate for our goals).

Hence, it is extremely important that the metamodels defining the languages for describing the various aspects of a Web application need to be separated from each other as much as possible. So we suggest of decomposing the various methods into a combination of models, each of which conforms to a well-defined part of the whole application domain regardless of the language used for the notation. For example, that allows of describing the structural model either in relational model, Entity Relationship (ER), UML or by using any custom DSL but it requires the separation of the structural model from any other models (e.g., navigational or requirements model). Besides, we suppose that no method uses a notation that does not conform to the MOF metapyramid (in fact, this is not a heavy constraint).

In our proposed solution, model weaving should appear on two levels:

- (1) On *intra-method* level, the relationships existed before the decomposition of the concerns need to be defined in a weaving model in order to be able to produce the same level of expressiveness. Let us consider the well-known conference management system as an example! In UWE, for instance, we would have a UML class called *Paper* in the structural model while its derived (and stereotyped) versions would appear in the navigational and presentational model, as well. Instead of the given method's built-in notation for this derivation, weaving links should be established in a weaving model that comprises statements about the relationship between the models in question. This weaving model can also be used later on when the starting point of the design is the building of the structural model as it captures the semantics that structural model elements also become (stereotyped) elements of the navigational model under given circumstances so a transformation might be applied to the structural model in order to create an initial version of the navigational one.
- (2) On *inter-method* level, when the relationships described by the weaving model define which model elements of a given model M_a conforms to which model elements in M_b . M_a and M_b here typically have the same level of abstraction (e.g., they both are structural models described by different methodologies) and the weaving model is defined between their corresponding MM_a and MM_b metamodels. For example, if one of the methods uses ER for describing the structural model while the other one applies UML for the same purpose, then the weaving model should contain that the strong entity type of the ER corresponds to a class in a UML class diagram, etc. This approach allows not only the generation of such a model transformation based on the weaving model that can transform a model in a notation into another model of another notation but model traceability is also supported.

ACKNOWLEDGEMENT

The work is supported by TÁMOP 4.2.1./B-09/1/KONV-2010-0007/IK/IT project. The project is implemented through the New Hungary Development Plan co-financed by the European Social Fund, and the European Regional Development Fund.

REFERENCES

- [1] S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, and M. Matera. *Designing Data-Intensive Web Applications*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

- [2] M. D. D. Fabro, J. Bézin, F. Jouault, E. Breton, and G. Gueltas. AMW: a generic model weaver. In *Proc. of the 1cre Journe sur l'Ingnierie Dirige par les Modcles (IDM05)*, 2005.
- [3] M. D. D. Fabro and F. Jouault. Model Transformation and Weaving in the AMMA Platform. In *Pre-proc. of the Generative and Transformational Techniques in Software Engineering Workshop*, pages 71–77, 2005.
- [4] J. Gómez and C. Cachero. OO-H method: extending UML to model web interfaces. pages 144–173, 2003.
- [5] N. Koch and A. Kraus. Towards a common metamodel for the development of web applications. Cueva Lovelle, Juan Manuel (ed.) et al., Web engineering. International conference, ICWE 2003, Oviedo, Spain, July 14-18, 2003. Proceedings. Berlin: Springer. Lect. Notes Comput. Sci. 2722, 497-506 (2003)., 2003.
- [6] N. Moreno, S. Meliá, N. Koch, and A. Vallecillo. Addressing new concerns in model-driven web engineering approaches. In *Proceedings of the 9th international conference on Web Information Systems Engineering, WISE '08*, pages 426–442, Berlin, Heidelberg, 2008. Springer-Verlag.
- [7] D. Schwabe. A conference review system. www.dsic.upv.es/~west/iwmost01/files/ConferenceReviewSystem.doc, 2001. [Online; accessed 24-April-2011].
- [8] D. Schwabe and G. Rossi. An object oriented approach to web-based applications design. *Theor. Pract. Object Syst.*, 4(4):207–225, 1998.
- [9] W. Schwinger and N. Koch. *Modeling Web Applications*, chapter 3, pages 39–64. John Wiley & Sons, 2006.

H-4032 DEBRECEN, EGYETEM TÉR 1.
E-mail address: adamkoa@inf.unideb.hu
E-mail address: kollarl@inf.unideb.hu