DOI: 10.2298/CSIS110823010A

Specification of Data Schema Mappings using Weaving Models

Nenad Aničić[1], Siniša Nešković[1], Milica Vučković[1] and Radovan Cvetković[2]

[1]Faculty of Organizational Sciences,
Jove Ilića 154, 11000 Belgrade, Serbia
{nenad.anicic, sinisa.neskovic, milica.vuckovic}@fon.bg.ac.rs
[2]Telekom Srbije A.D., Technical Affairs Division
Bulevar umetnosti 16a, 11000 Belgrade, Serbia
radovan.cvetkovic@telekom.rs

Abstract. Weaving models are used in the model driven engineering (MDE) community for various application scenarios related to model mappings. However, an analysis of its suitability for specification of heterogeneous schema mappings reveals that weaving models lack support for mapping rules and, therefore, cannot prevent mapping specifications which are semantically meaningless, wrong or disallowed. This paper proposes a solution which overcomes the identified open issue by providing the explicit support for semantic mapping rules. It is based on introduction of weaving metamodels augmented with constraints written in OCL. The role of OCL constraints is to restrict mapping specifications to only those which are semantically meaningful. Using well known MDE technologies, such as EMF and QVT, an existing tool is used to validate the presented solution. This solution is also successfully evaluated in practice.

Keywords: schema mappings, weaving models, model transformations

1. Introduction

Specification of mappings among heterogeneous schemas[1] has been studied in many different research areas, such as distributed databases [1], data warehouses [2], ontologies [3], model driven development [4], [5], [6], etc. According to specific needs and characteristics of a particular problem domain, researchers have proposed different approaches and techniques that can be used to specify schema mappings.

Without diving into details of each particular approach, it can be generally concluded that most of them rely on a mapping specification formalism,

[1] The term schema is used in a broader sense and includes database schemas, ontologies, or generic models.

embodied in the form of either a special language or metamodel, which enables expressing and capturing the semantics of correspondences among schema concepts. In this paper we deal with the problem of finding a suitable formalism for the specification of schema mappings.

In accordance to the motto that “models are everywhere”, the corresponding mapping specification formalism in the context of the model driven engineering (MDE) utilizes (meta)models. One particular solution which has been proposed is based on so called weaving models [5], [6], [7]. A weaving model is a separate model on its own consisting of elements which represent individual links (i.e. correspondences) among elements of other models (called woven models). A weaving model conforms to a weaving metamodel, which provides the semantics of links specified in a weaving model. A weaving metamodel defines types of links that can occur among elements of woven models, i.e. links specified in a weaving model can be instances only of types defined in the weaving metamodel. A special core weaving metamodel with generic link types suitable for a range of different application scenarios is also proposed. For each application scenario, the core weaving metamodel is extended with specialized link types that are more suitable for the particular application. Supported by the ATLAS Model Weaver (AMW) toolkit [8] within the Eclipse Modeling Framework (EMF) environment, the proposed approach has gained a lot of attention in the MDE community lately. It has been reported that weaving models are successfully applied to several MDE related problems, including schema and data mapping problems [5].

However, our experience in the application of weaving models to the problem of schema mappings reveals that there exist some open issues. Namely, the definition of a weaving metamodel is based only on concepts of metameta model (i.e. Ecore metameta model in EMF). It does not rely on concepts of corresponding metamodels of models intended to be woven. Hence, it is completely unaware of any semantic rules regarding mappings among concepts of the metamodels in question. As a consequence, link types defined in a weaving metamodel cannot prevent links between elements of woven models which are semantically meaningless, wrong or disallowed. For example, when mapping concepts between an entity-relationship (ER) data schema S1 and a relational schema S2, it is possible to link an entity from the S1 schema with a column from the S2 schema. In other words, the weaving model lacks the semantics of mapping rules between ER and relational schemas, i.e. that entities can be mapped to relational tables only.

This paper proposes a possible solution which overcomes the identified open issues by providing explicit support for mapping rules. The solution is based on a weaving model which serves for definition of mapping rules between schema metamodels. This weaving model is then transformed to a weaving metamodel augmented with Object Constraint Language (OCL) constraints. The role of OCL constraints is to restrict links in weaving models to establish only those relationships between schema concepts which are meaningful.

The proposed solution is successfully tested in practice on several different schema mapping problems. This is supported by the set of software tools consisting of open-source plug-ins for model weaving (AMW [8]), model transformation (M2M QVT Operational) and model validation (Eclipse OCL).

The paper is organized as follows. The next section briefly presents the related work. Section 3 introduces a general conceptual framework, which is used to identify types of models that occur in the context of data schema mappings and to define their roles and mutual mappings. Section 4 analyses the existing practice in utilizing weaving models and discusses shortcomings and open issues. In Section 5 the proposed solution is explained and discussed. In the same chapter, a technical realization of the suggested approach is provided as well as an illustrative example. Conclusion is given in Section 6.

2. Related Work

Schema mappings are high-level specifications which express correspondences between elements in heterogeneous schemas. Although schema mappings have been studied independently in different contexts, there are two main issues involved:

– The first one is to discover the correspondences between schema elements that are semantically related in the source and target representations. This process of discovering correspondences is called schema matching;
– After the correspondences have been established, the second issue is to produce operational mappings that can be executed to perform the data exchange between source and target.

Many techniques have been proposed to discover the correspondences between schemas. One particular approach is implemented in Clio [9], which generates semi-automatic mappings between schemas based on value correspondences obtained from the user or by a machine learning technique. Furthermore, Clio supports the definition of nested mappings, but this definition cannot be extended. This makes it difficult to create complex mappings. Another method, presented in [10], [11] is called Similarity Flooding (SF). Schemas are presented as directed labeled graphs and this structural method propagates the similarity of a pair of nodes through their neighbors. However, this approach cannot be used to define a correspondence between schemas (models) conforming to different schema languages (metamodels). There are a number of other different approaches for automatic schema matching. Their thorough survey is given in [12], but they are not discussed here as our focus in this paper is on manual schema specifications.

Schema operational mappings typically incorporate the data transformation semantics required to specify how data from one source representation (e.g. a relational schema) can be translated to a target representation (e.g. an XML schema). They are also used for virtual information unification where users can pose queries over distributed heterogeneous data sources in a uniform and transparent manner.

In the case of information integration with heterogeneous schema languages, several approaches have been proposed [13], [14], [15]. These approaches are mostly based on a generic metamodel that abstracts concrete schema metamodels. Schema mappings in this case are specified in a language which depends on the chosen generic metamodel. For instance, in [13] a universal metamodel based on the supermodel is used as a generic metamodel and DATALOG is used for representing schema mappings. In [15], the GeRoMe model and corresponding specification language are used.

In the context of MDE, specifications of schema mappings and data mappings can be viewed as a special case of model mappings. Another special case of model mappings are model transformations, which represent a crucial notion in MDE. The MDE community has proposed several model transformation specification languages. For instance, OMG has defined the QVT language [16], ATL is used in the EMF environment [17], etc. Although transformation specification languages could be used for data schema mappings, they are not designed for such a task. Transformation specification languages are used to specify mappings between metamodels (M2 level models) and, consequently, are inconvenient for the specification of schema mappings, which are M1 level models.

Model weaving is an approach used for establishing fine-grained correspondences between model elements. It is supported by AMW, a component of the larger Atlas Model Management Architecture toolkit [8]. This approach is conceived with a goal to facilitate a range of different application scenarios, such as tool interoperability, model composition operations, traceability, model alignment, etc [8]. One group of supported application scenarios is related to schema mappings. The work from [5] presents the application of weaving model to discover model mappings and production of executable operational mappings (including model transformations) which translate from source models to targets. These results are extended by the work in [7], which utilize successive schema matching transformations to generate and refine a sequence of weaving models until a final one is generated, out of which data transformations are produced. Weaving models are created using different methods. In this paper we not deal with matching heuristics or algorithms which can be used to automatically create weaving models. We created weaving models manually by graphical user interfaces. The created weaving model may be later used by a model transformation language to translate source data model(s) into target data model(s).

3. Conceptual Data Integration Framework

In order to analyze suitability of the weaving modeling approach for the specification of schema mappings, we introduce a conceptual data integration framework, shown in Figure 1. The main purpose of the introduced framework is to identify kinds of models that occur in the context of data schema mappings and to precisely define their roles and mutual mappings.

The framework has 4 abstract levels, which correspond to the levels of OMG MDA standard [4], but are named here in the manner that is more appropriate for data integration purposes. As it is typical for metamodeling architectures, each level accommodates models which serve as metamodels for other models from the lower abstract level, whilst they must conform to their metamodels from the upper level. The exception is the model at the most abstract level which conforms to itself.

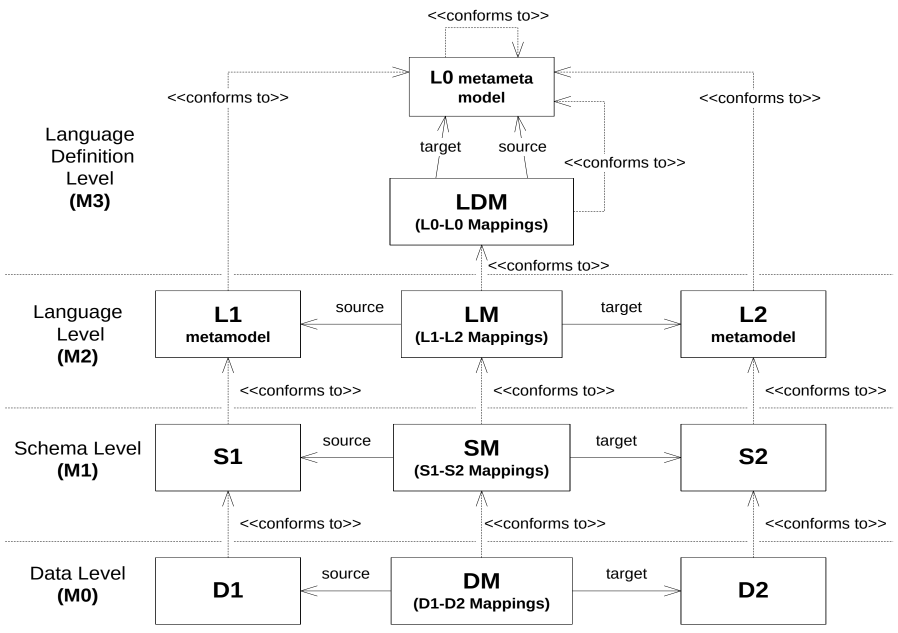


Fig. 1. Conceptual data integration framework

The framework identifies two types of models: (1) ordinary models which are used to describe domain concepts, and (2) mapping models which links elements from other ordinary models[2]. Depending on the abstract level where they reside, the following four kinds of mapping models can be identified:

– *Data Mapping Models* (DM) which specify links at the Data Level, i.e. between data instances stored in different possibly heterogeneous data sources. This level coresponds to M0 level of MDA.

[2] Mappings between different mapping models are also possible, but their consideration is beyond the scope of this paper.

– *Schema Mapping Models* (SM) which specify links at the Schema Level, i.e. between schema concepts that are possibly expressed in different schema languages. This level coresponds to M1 level of MDA.
– *Language Mapping Models* (LM) which specify links at the Language Level, i.e. specify mappings between concepts of different schema languages. This level coresponds to M2 level of MDA.
– *Language Definition Mapping Models* (LDM) which specify links at the Language Definition Level, i.e. between concepts of a metameta model used to describe schema languages. This level coresponds to M3 level of MDA.

It is important to note that mapping models must conform to their corresponding metamodels, which are also mapping models. This means that links specified in one mapping model must be instances of links specified in its mapping metamodel. In other words, links specified in the upper abstract level constrain links in the lower level to relate only certain types of model elements. Thus, links serve as mapping rules for the lower level allowing only meaningful links and preventing invalid ones.

The example shown in Figure 2 illustrates this for the case of mappings between a relational schema and an XML schema. As it is depicted in the figure, table Publication is mapped to XML element Book by a link which is an instance of the rule mapping relational tables to XML elements, whereas column ISDN is mapped to XML element BookID by a link which is an instance of the rule mapping columns to XML elements.

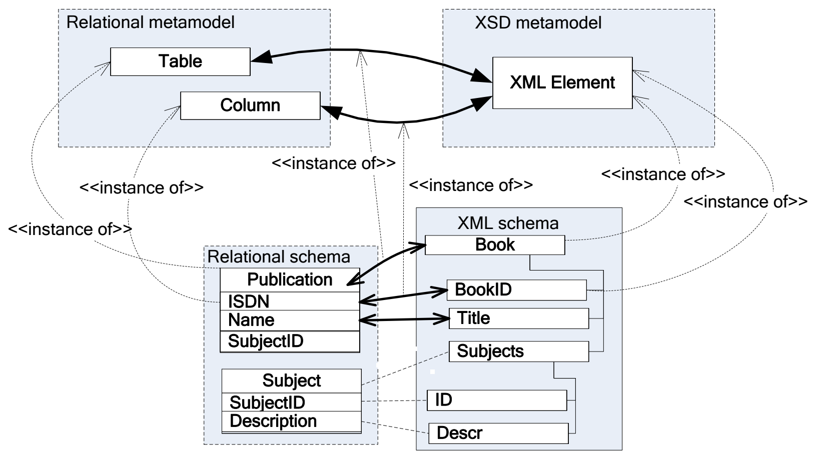


Fig. 2. Links at two different abstract levels

A special case is LDM, which is used to define rules for mappings between schema languages. Since it represents the most abstract mapping model in the framework, it is defined in terms of concepts of a corresponding metameta model *L0*.

4. Open Issues

Model mappings based on weaving models represent an approach in MDE which is supported by the AMW toolkit [17], [18]. AMW is a tool for establishing relationships (i.e., links) between model or metamodel elements and it is aimed to support a range of application scenarios where model mappings are involved. Here, we will discuss the approach from the schema mappings perspective only.

AMW supports an extensional mechanism based on the core weaving metamodel that encompasses a set of features (i.e. generic concepts) common in majority of application scenarios. Using extensions one defines a new weaving metamodel based on the core weaving metamodel. A new weaving metamodel typically defines new link types which are specific to a particular application scenario. Defined link types are used in weaving models for relating elements of two woven models.

Using the conceptual data integration framework given in Section 3 as a reference model, the typical application scenario employed by the AMW approach is shown in Figure 3. Here, the Ecore metamodel of EMF plays its usual role of the most abstract models *L0*. The role of LDM model in the conceptual framework is played by the core weaving metamodel, which is defined in terms of Ecore concepts. The role of LM, used to define mapping rules between different schema languages, is played by a weaving metamodel. It is defined as an extension of the core weaving metamodel. Note that this is different in comparison to our conceptual framework where LM is defined as an instance of a metamodel. In addition, a weaving metamodel in AMW does not specify mappings between concepts of a language, but simply defines a new link type. In other words, the semantics is provided only by giving a new name, without specifying schema concept types allowed to be related by this link type.

Specification of the mapping rules is given as an extension of the core weaving metamodel (Figure 3), which is an abstract metamodel. Therefore, the specification of the mapping rules is given at the language level and schema level separately, using LDM1 and LDM2. LDM1 specifies mapping rules (types of links) between elements of metamodels (i.e. language concepts) which are usualy simple equality links (e.g. *ElementEqual, AttributeEqual*). LDM2 specifies mapping rules for links typically expressing equality or set-inclusion relationships (e.g. *EqualLink, NestedLink, ChildLink*) which disregard types of linked schema elements. In this case, SM mappings between two concrete schemas conform to LDM2 mapping rules and do not conform to the defined LM mapping rules between the corresponding metamodels.

Of course, one can use the same weaving metamodel for specification of both LM and SM mappings. In this case LDM1=LDM2, i.e. they are the same weaving metamodel describing common link types (such as equality links) which are generally applicable to elements of different model types and at several abstract levels of our conceptual framework.

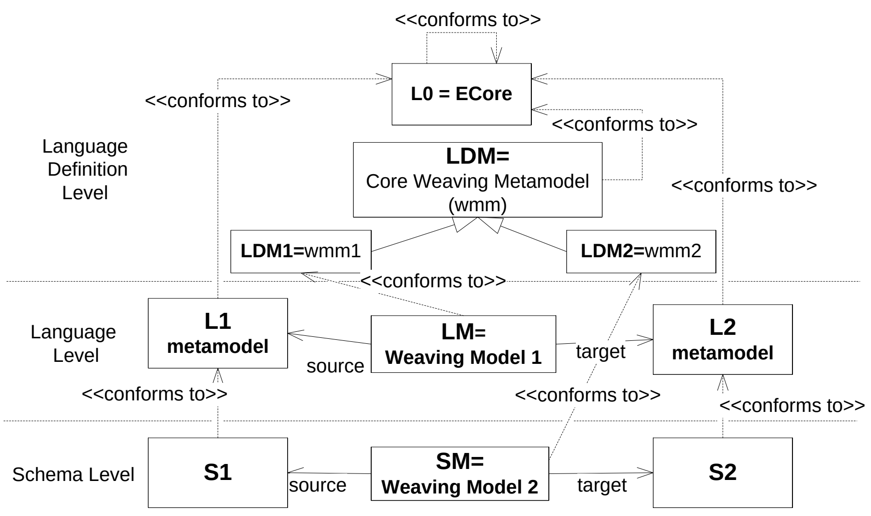


Fig. 3. The AMW approach

The role of SM, used to specify schema mappings, is played in AMW by a weaving model. It is defined as an instance of its corresponding weaving metamodel, which is in accord with the conceptual framework. However, links specified by a weaving metamodel can relate any elements from woven models. It is up to a modeler to take care whether such links are meaningful.

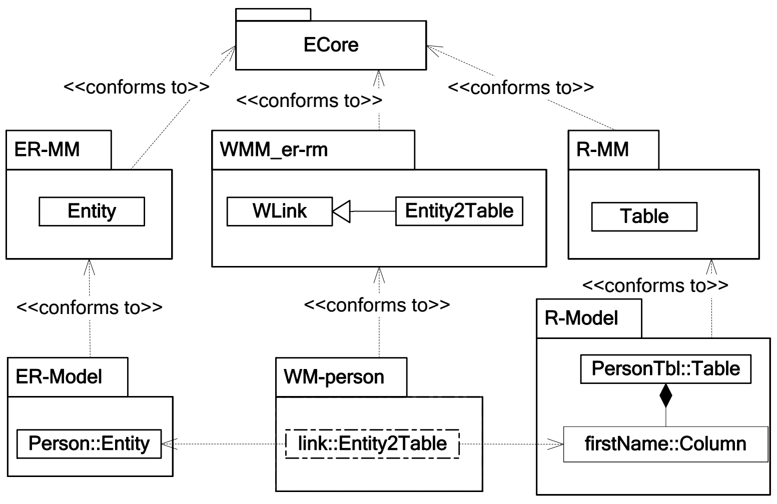


Fig. 4. A semantically invalid link in a weaving model

Figure 4 illustrates the situation that can happen when a modeler is careless or unaware of semantic mapping rules. The rule *Entity2Table* is specified in the weaving metamodel meaning that entities from ER models are translated to tables in relational schemas. However, as it is shown in the figure, it is possible to create a link which is an instance of the defined rules, but maps an entity to a column.

The Data level from the conceptual framework is not actually supported by AMW. However, application scenarios involving data instances still can be supported by lifting of models from the Data and Schema levels for one abstract level up, i.e. by expressing and treating data schemas as metamodels and data instances as M1 models. Such technique is employed in [5]. However, it leads to weaving models that must make up for a lack of models from the Language level, which is lost due to the level lifting. This technique introduces the accidental complexity to the definition of weaving models. Due to limited space, the further detailed discussion of this technique is beyond the scope of this paper.

To summarize the discussion, the following shortcomings and open issues exist:

- Weaving models may contain an invalid specification of schema mappings.
- Weaving models are not adequately constrained by their corresponding weaving metamodels.
- Link ends, which are part of link type definitions, cannot be typed, i.e. specified to which concept types they may relate.
- Data modelers are supposed to know the semantics of mapping rules and must take care about links they create.

5. Solution

The open issues discussed above can be resolved by a better alignment of the AMW approach with the conceptual data integration framework defined in Section 3. Better alignment in the context of schema mappings primarily means that link types defined in weaving metamodels have to constrain links in weaving models to relate those schema concepts that are meaningful. In other words, AMW approach needs to be extended with support for specifications of mapping rules in weaving metamodels which would enable typed end points of links in weaving models.

This paper proposes a solution to this problem by introducing several special weaving metamodels and models which have specific roles. The proposed solution is given in Figure 5.

One special weaving model having LM role is used to specify mapping rules by establishing semantically meaningful links between concepts from two schema languages. Due to constraints of EMF environment and AMW tool, this weaving model cannot be simultaneously used as a metamodel for weaving models from the Schema level. Hence, this weaving model is transformed by a model transformation into another special weaving metamodel augmented with OCL constraints. OCL constraints are integral parts of link type definitions and they specify types of data schema concepts that can be related by a particular link type. In this way, end points of links in weaving models are allowed to reference only instances of the specified types.

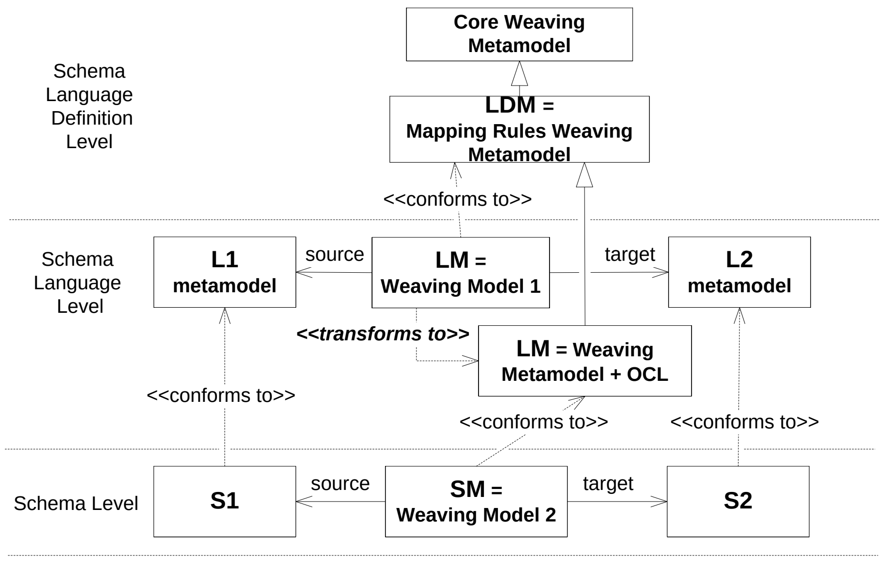


Fig. 5. Proposed solution

It is worth of observing that both the special weaving model and the generated weaving metamodel contain same information about mapping rules. However, the mapping rules are expressed in different ways in these models. In the weaving model they are expressed as structural constrains via (weaving) links, whilst in the weaving metamodel they are expressed via OCL as value based constraints. Therefore, both models have the same LM role defined in the conceptual framework in Section 3. In addition, both models are related to the same weaving metamodel. But unlike the AMW approach where they are related to the core weaving metamodel, here a new special weaving metamodel playing the LDM role is introduced. Also, they are related in a different way. The LM weaving model conforms to the LDM mapping rules weaving metamodel, whilst LM weaving metamodel is defined as its extension.

In order to describe and illustrate the proposed solution in more detail, in the rest of this section we first present LDM mapping weaving metamodel. Then, we illustrate the definition of mapping rules between concepts of ER and relational model using LM weaving model. Afterwards, we describe the model transformation from the LM weaving model into the LM metamodel augmented with OCL constraints. In the second section, we provide an example of specifying mappings between two concrete data schemas expressed in ER and relational model using the SM weaving model adhering to the mapping rules given in the generated LM metamodel. At the end of the section we summarize our apporach and describe its technical realization.

5.1. LDM Mapping Rules Waving Metamodel

LDM mapping rules weaving metamodel is defined as an extension of abstract core weaving metamodel (WMM) within AMW toolkit [8]. Both models are shown in figure 6 using UML class diagrams.

WMM model, which is shown in *mwcore* UML package in figure 6, defines abstract concepts used to specify mappings (class *Wlink*) between elements of some models (class *WlinkEnd*), as well as for referencing these models and their elements (classes *WModelRef* and *WElementRef*).

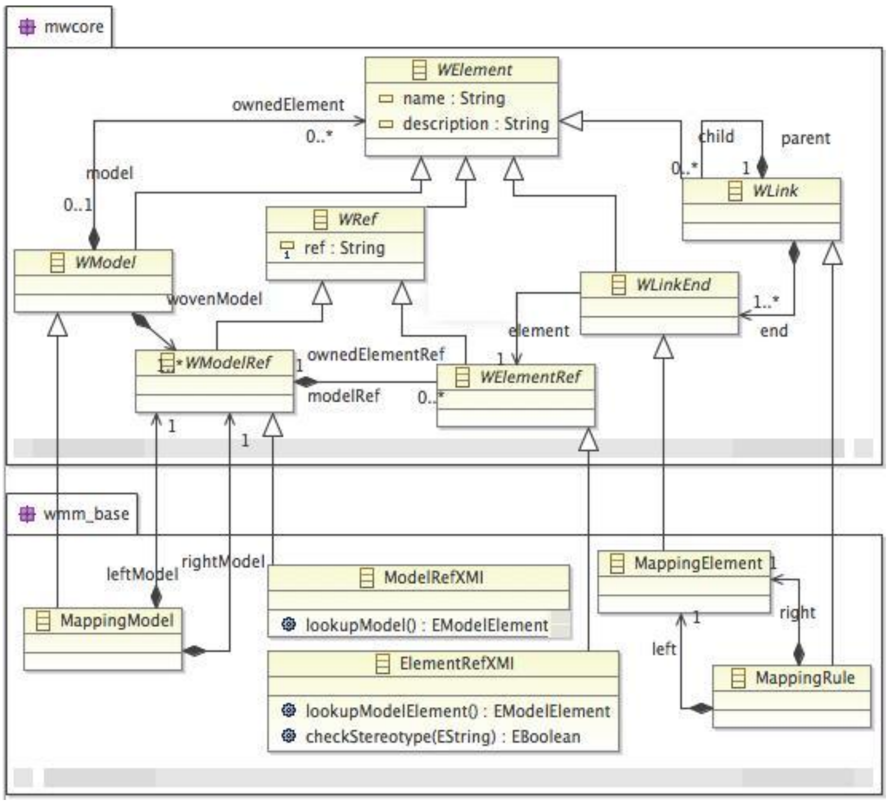


Fig. 6. LDM mapping metamodel

The definition of LDM mapping rules metamodel is given in the package *wmm_base*. It is assumed that mapping rules relates concepts of two different schema languages are represented as metamodels via standard UML class diagrams. *MappingRule* and *MappingElement,* modeled as subclasses of the concepts *Wlink* and *WlinkEnd* respectively*,* are used to define mapping rules between elements of these two different metamodels. References *left* and *right* enable to define a one-to-one mapping of elements, (reference left points to this) from one metamodel to an element (reference right points to this) of the other metamodel. *ModelRefXMI* and *ElementRefXMI,* modeled as subclasses of the concepts *WModelRef* and *WElementRef* respectively, represent references to UML models and elements serialized in XMI format. *ModelRefXMI has an* operation named *lookupModel*, which returns some UML2 model. (UML2 is an EMF-based implementation of the Unified Modeling Language 2.x OMG metamodel for the Eclipse platform). *ElementRefXMI* has two operations: operation *lookupModelElement* which returns an element of the UML2 model based on a reference, and operation *checkStereotype* which checks the applied stereotype of the UML2 model element to which the given reference points. These operations are introduced to simplify the definition of OCL constraints and queries in concrete weaving models.

5.2. An example of mapping rules

This subsection presents an example of mapping rules between concepts of ER and relational schema languages. Simplified metamodels for these two schema languages are shown in figures 7 and 8 respectively.

ER metamodel (fig. 7) includes the concepts of *Entity*, *Attribute*, *Relationship* and *Role*. *Entity* has an arbitrary number of *Attributes*. *Relationship* represents a binary association between two *Entities.* Each Entity participates in a *Relationship* with some *Role*.

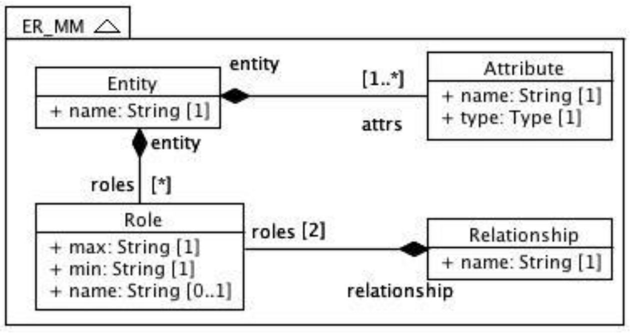


Fig. 7. Simplified ER metamodel

Relational metamodel (fig. 8) includes the concepts: *Table*, *Column* and *Key*. Each *Table* contains *Columns.* One or more columns of some Table represent its *Key*. *ForeignKey* and *PrimaryKey* are modeled as specializations of the concept *Key.* We use the reference refFK to connect a value of the foreign key in a table with a value of the primary key in some other table.

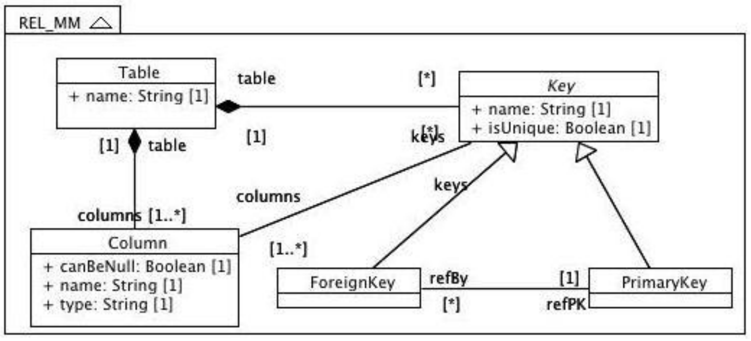


Fig. 8. Simplified relational metamodel

To define mapping rules between the ER and relational metamodel, we create a weaving model with LM role which is an instance of LDM mapping rules metamodel defined in the previous subsection. This weaving model is represented here as an UML object diagram shown in figure 9. It is an instance of the class diagram form figure 6. The LM weaving model in our example defines several mapping rules between concepts of data schemas, such as *Entity2Table*, *Attribute2Column* and *Relationship2FK.* These mapping rules constrain which concepts of ER data model can map to which concepts of relational data model. They are defined as instances of *MappingRule* class from LDM mapping rules metamodel.

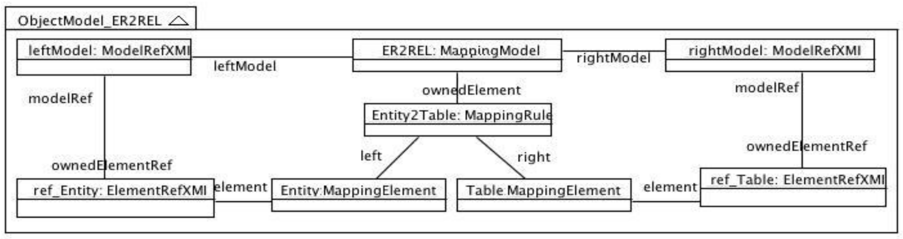
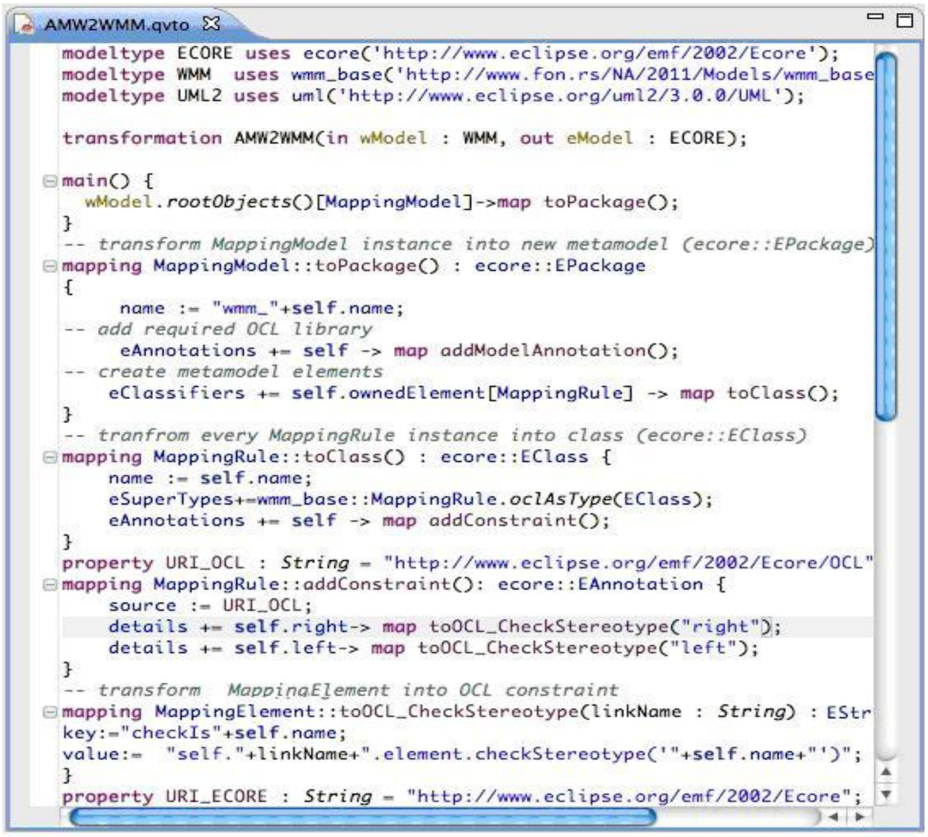


Fig. 9. An example of LM mapping model

5.3. Model transformation

The model transformation from LM weaving model to LM weaving metamodel is expressed using OMG's standard QVT Operational (QVTo) model transformation language [16]. Its implementation in EMF environment is depicted in figure 10.


AMW2WMM.qvto

```
modeltype ECORE uses ecore('http://www.eclipse.org/emf/2002/Ecore');
modeltype WMM  uses wmm_base('http://www.fon.rs/NA/2011/Models/wmm_base
modeltype UML2 uses uml('http://www.eclipse.org/uml2/3.0.0/UML');

transformation AMW2WMM(in wModel : WMM, out eModel : ECORE);

main() {
  wModel.rootObjects()[MappingModel]->map toPackage();
}
-- transform MappingModel instance into new metamodel (ecore::EPackage)
mapping MappingModel::toPackage() : ecore::EPackage
{
     name := "wmm_"+self.name;
-- add required OCL library
     eAnnotations += self -> map addModelAnnotation();
-- create metamodel elements
    eClassifiers += self.ownedElement[MappingRule] -> map toClass();
}
-- tranfrom every MappingRule instance into class (ecore::EClass)
mapping MappingRule::toClass() : ecore::EClass {
    name := self.name;
    eSuperTypes+=wmm_base::MappingRule.oclAsType(EClass);
    eAnnotations += self -> map addConstraint();
}
property URI_OCL : String = "http://www.eclipse.org/emf/2002/Ecore/OCL"
mapping MappingRule::addConstraint(): ecore::EAnnotation {
    source := URI_OCL;
    details += self.right-> map toOCL_CheckStereotype("right");
    details += self.left-> map toOCL_CheckStereotype("left");
}
-- transform  MappingElement into OCL constraint
mapping MappingElement::toOCL_CheckStereotype(linkName : String) : EStr
key:="checkIs"+self.name;
value:=  "self."+linkName+".element.checkStereotype('"+self.name+"')";
}
property URI_ECORE : String = "http://www.eclipse.org/emf/2002/Ecore";
```

Fig. 10. Model transformation in QVTo

This transformation consists of a series of QVTo transformation mappings, which are presented in table 1.

Table 1. QVT transformation rules

WMM model (source)		Ecore Model (target)
MappingModel	=>	EPackage which extends wmm_base
MappingRule	=>	EClass which extends wmm_base::MappingRule
MappingElement	=>	EAnnotation which respresnts OCL constraint

According to the transformation mappings, each source LM weaving model is transformed into a new target LM weaving metamodel (represented as an EPackage meta class in EMF). Each instance of *MappingRule* within the source LM weaving model becomes a new meta class (EClass in EMF) within the target LM weaving metamodel. This new meta class is generated as a subclass of *wmm_base::MappingRule,* indicating that it represents a specific mapping rule. Every *MappingElement* of some *MappingRule* is transformed into corresponding OCL constraints represented in the target LM weaving metamodel as EAnnotation meta classes. Each OCL constraint restricts types that some concrete mapping between two concrete data schemas is allowed to reference.

Figure 11 shows the result of the defined model transformation when it is applied to the LM weaving model from our example given in figure 9.

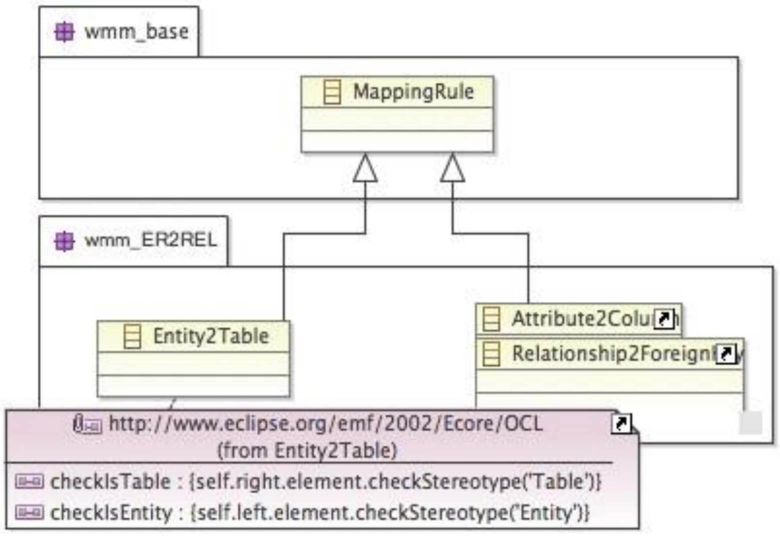
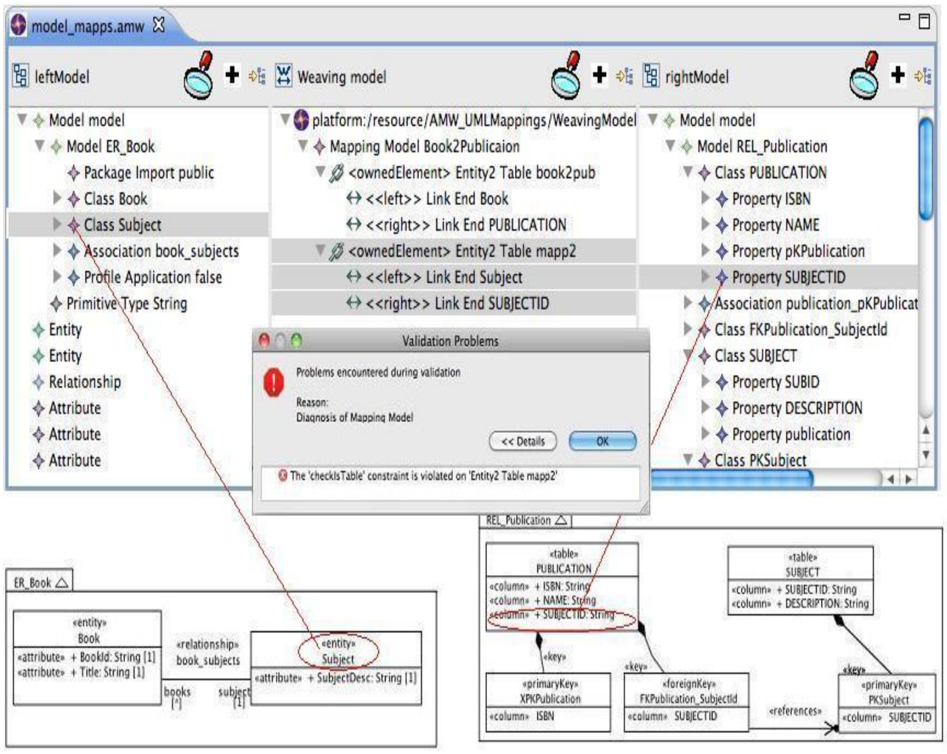

wmm_base
MappingRule
wmm_ER2REL
Entity2Table
Attribute2Colu
Relationship2Foreign
http://www.eclipse.org/emf/2002/Ecore/OCL
(from Entity2Table)
checkIsTable : {self.right.element.checkStereotype('Table')}
checkIsEntity : {self.left.element.checkStereotype('Entity')}

Fig. 11. Mappings rules between ER and relational model

The main effect of the transformation is in changing the way how mapping rules are represented. In the source model (LM model shown in fig. 9) they are represented through an UML object model, which is more suitable for their creation (i.e. specification of mapping rules). In the target LM weaving metamodel (shown in fig. 12), mapping rules are represented through an UML class diagram augmented with OCL constrains, which is more suitable for their instantiation (i.e. creation of concrete data schema mappings complying with the defined mapping rules).

5.4. An example of schema mappings

An example of concrete data schema mappings is given in figure 12. *ER_Book* and *REL_Publication* are two concrete data schemas expressed using ER and relational data model respectively.



Validation Problems

Problems encountered during validation

Reason:
Diagnosis of Mapping Model

The 'checkIsTable' constraint is violated on 'Entity2 Table mapp2'

Fig. 12. WM mapping model between two concrete models

The schemas are represented in the lower part of the figure as UML class diagrams using corresponding UML stereotypes defined for these two data models[3]. In the upper part of the figure the same schemas are depicted in the way how they are represented within AMW Weaving Editor, i.e. as trees of schema elements shown in the left and right tree-view subwindows within the main window of AMW Weaving Editor. Lines in the figure designate correspondences between the same elements in the UML class diagrams and tree nodes. The weaving model, specifying concrete mappings between

[3] Definition of these UML stereotypes is omitted in the paper for brevity.

elements of the two schemas, is also represented as a tree shown in the middle tree-view subwindow.

The shown example includes specification of several mappings between elements of the two concrete data schemas. One of the shown mappings is book2pub, which is an instance of Entity2Table mapping rule. This mapping maps entity Book from ER_Book schema into table PUBLICATION from the REL_Publication schema. Since both OCL constraints checkIsEntity and checkIsTable of Entity2Table mapping rule are satisfied, book2pub mapping is valid. Another shown mapping is *mapp2*, which maps entity *Class Subject* to column *SUBJECTID.* However, this mapping is invalid because it violates the OCL constraint *checkIsTable*, i.e. the target is not a table. Figure 12 shows the message displayed after checking the validity of the weaving model.

5.5. Discussion

In order to specify schema mappings, the fundamental requirement is to first identify correspondences between elements of schemas. Correspondences can be either generated through an automatic matching process, or can be provided manually by an expert. In our solution the process matching is not automated but supported through mapping rules which restrict semantically valid correspondences. However, our approach can be also used as a basis for the so called constraint-based automatic matching approach [12].

Our solution is primarily aimed to support manual specification of operational schema mappings. This manual specification of schema mappings is supported by the set of software tools consisting of open-source plug-ins for model weaving (AMW [8]), model transformation (M2M QVT Operational) and model validation (Eclipse OCL). All these tools are built on top of the Eclipse Modeling Framework (EMF) [19] and are available as open source from the GMT (Generative Modeling Technologies), M2M (Model-to-Model Transformation) and MDT (Model Development Tools) Eclipse modeling projects [20]. To provide support for our specific approach, existing software tool are extended by LM weaving metamodel (defined in section 5.3.), which restrict the number of match candidates for mappings between two concrete schemas.

To validate our approach, we conducted a set of experiments. Despite presented example being simple, it is easy to envisage the creation of mappings between very large source and target models. However, this solution is successfully evaluated in practice in the telecomunication domain. Our experiments tested mappings between internal and standard-based models (addressing both structural and behavioral aspects) of business processes, information entities and applications.

It is worth of noting that this approach to specify mappings could be applied on the data level as well. Models on the data level contain data which adhere to the corresponding schemas, so that we can also make correspondences between them using the weaving metamodel. We previously discussed how to use mapping rules which would restrict correspondences to meaningful ones. That discussion is also valid here.

Generally, the correspondences between data instances would not be defined explicitly, as is shown in figure 12. Instead, mapping rules would be used that are given in the schema mapping model (SM), and which define the transformation algorithms from source data to target data. SM model is used as a specification to produce transformation models in different target languages, such as ATL, XSLT, or SQL-like languages. The transformation models are extracted to the corresponding concrete syntax and additionally can generate a data weaving model (DM). Of course, in that case DM must be in the correspondence to the schema mapping SM. Also, the weaving metamodel that is generated, contains constrains which are used to prevent semantically forbidden links between the elements (data instances) of source and target data models. For example, it is not possible to link an instance *a1* of an entity *Author*, with an instance *p1* of the table *Publication* (Fig, 13). This is because there is no defined mapping between *Author* and *Publication* on the schema level. This weaving model, which is the result of a transformation, is usually called traceability model. Traceability model can be also used to check if transformation models are valid.

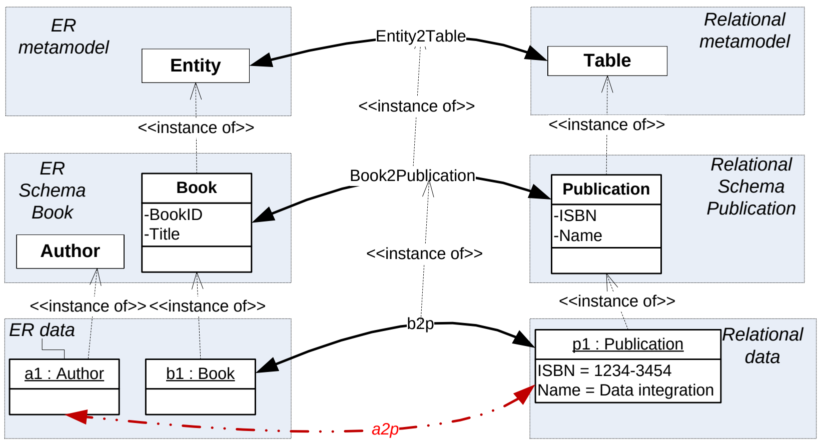


Fig. 13. Example of data mappings

6. Conclusion

This paper proposed an approach for data schema mappings which is based on weaving models. Main contributions of the paper are the following:

– A conceptual data integration framework introduced to identify kinds of models that occur in the context of data schema mappings and to precisely define their roles and mutual relationships;

– An analysis of suitability of weaving models for specification of schema mappings, which reveals that the existing approach supported by AMW toolkit does not properly support schema mappings. This is mainly due to inability of AMW approach to properly define mapping rules between schema languages;
– A proposed solution enabling the proper definition of mapping rules between schema languages, which is based on the introduction of special weaving models and metamodels with OCL constrains.

It can be concluded that the existing AMW approach based on weaving models has been carefully conceived from both theoretical and technological points of view to be general and flexible enough. This generality and flexibility enables weaving models to be applicable in a wide range of MDE related tasks. However, such generality and flexibility have shortcomings when applied in the specific domain of data schema mappings. Therefore, to overcome these shortcomings, a domain specific solution is proposed, which extends and improves the general one of AMW approach.

Acknowledgement. Research presented in this paper was supported by Ministry of Science and Technological Development of Republic of Serbia, Grant III-44010, Title: Intelligent Systems for Software Product Development and Business Support based on Models.

References

1. Lenzerini, M.: Data Integration: A Theoretical Perspective. PODS, 233-246. (2002)
2. Cui, Y., Widom, J.: Lineage Tracing for General Data Warehouse Transformations. VLDB J. 12, 41-58. (2003)
3. Ehrig, M., Sure, Y.: Ontology Mapping - An Integrated Approach. ESWS, 76-91. (2004)
4. Miller, J., Mukerji, J.: Model Driven Architecture (MDA). OMG Document. [Online]. Available: http://www.omg.org (current December 2011)
5. Del Fabro. M.D., Bézivin, J., Jouault, F., Valduriez, P.: Applying Generic Model Management to Data Mapping. In: Benzaken, V. (ed): BDA, Saint Malo, Actes. (2005)
6. Del Fabro. M.D., Valduriez, P.: Semi-automatic Model Integration using Matching Transformations and Weaving Models. SAC, 963-970. (2007)
7. Del Fabro, M.D., Valduriez, P.: Towards the Efficient Development of Model Transformations Using Model Weaving and Matching Transformations. Software and System Modeling 8(3), 305-324. (2009)
8. Del Fabro, M.D., Bézivin, J., Valduriez, P.: Weaving Models with the Eclipse AMW plugin. In: Eclipse Modeling Symposium, Eclipse Summit Europe. (2006)
9. Miller, R. J., Hernandez, M. A., Haas, L. M., Yan, L.-L., Ho, C. T. H., Fagin, R., and Popa, L.: The Clio Project: Managing Heterogeneity. SIGMOD Record 30(1), 78–83. (2001)
10. Melnik, S., Rahm, E., Bernstein P. A.: Rondo: A Programming Platform for Generic Model Management, Proc. SIGMOD 2003, 193-204. (2003)
11. Melnik, S.: Generic Model Management: Concepts and Algorithms Springer. Ph.D Dissertation, University of Leipzig. Springer LNCS 2967. (2004)

12. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB Journal 10(4), 334–350. (2001)
13. Atzeni, P., Cappellari, P., Torlone, R., Bernstein, P.A., Gianforme, G: Model-independent schema translation. The VLDB Journal 17, 1347–1370. (2008)
14. Atzeni, P., Gianforme, G., Cappellari, P.: A Universal Metamodel and Its Dictionary. A. Hameurlain et al. (Eds.): Trans. on Large-Scale Data- & Knowl.-Cent. Syst. I, LNCS 5740, 38-62. (2009)
15. Kensche, D., Quix, C., Li, X., Li, Y., Jarke, M.: Generic Schema Mappings for Composition and Query Answering. Data & Knowledge Engineering 68, 599-621. (2007)
16. Object Management Group: Meta Object Facility (MOF) 2.0 Query/View/ Transformation (QVT). [Online]. Available: http://www.omg.org/spec/QVT (current December 2011)
17. Jouault, F., Allilaire, F., Bézivin, J., Kurtev, I.: ATL: A model transformation tool. Sci. Comput. Program. (SCP) 72(1-2), 31-39. (2008)
18. Object Management Group (OMG): Object Constraint Language OMG Version 2.2. [Online]. Available: http://www.omg.org/spec/OCL/2.2/ (current December 2011)
19. Eclipse Modeling Framework Project (EMF). [Online]. Available: http://www.eclipse.org/modeling/emf/ (current December 2011)
20. Eclipse Modeling Projects. [Online]. Available: http://www.eclipse.org/modeling/ (current December 2011)

Nenad Aničić received the M.Sc. and Ph.D. degrees in Information Systems from Belgrade University, Serbia in 2001 and 2006, respectively. He is currently an associate professor in the Department of Information System and Technologies at the Faculty of Organizational Sciences, Belgrade University. He teaches courses in Business Process Modeling, Databases, XML Technologies and Applications, and Information Systems Design. His research interests include information systems development methodologies, model driven development, semantic technologies, and interoperable application systems.

Siniša Nešković received the Ph.D. and M.Sc. degrees in information systems from the University of Belgrade. He is a lecturer at the Faculty of Organizational Sciences (FOS), University of Belgrade, where he teaches courses in Data Structures and Algorithms, Business Process Modeling and Software Architectures. He currently heads the Information Systems Department and Laboratory for Information Systems of FOS. His research interests include information systems development, business process modeling and automation, model driven development, software product line engineering and advanced software architectures.

Milica Vučković works as an assistant professor in the Department of Information Systems, at the Faculty of Organizational Sciences, University of Belgrade. She received B.Sc., M.Sc. and Ph.D. degrees in information systems from the Faculty of Organizational Sciences. The areas of her research interest include: search of ontologically heterogeneous distributed resources on the Web, Model Driven Engineering, Model mappings and transformations. So far, she has authored/co-authored approximately 50 research papers. She participated in a number of research and industrial projects in the area of information systems development and software engineering.

Radovan Cvetković is an expert for Telco Information System development and implementation in company “Telekom Srbija”, Belgrade. In the same company he was the Director of IT development, and after that CIO. He received M.Sc. from Belgrade University in 1989 in domain of Information Systems and Technology. It is expected at the same university the defense of doctoral thesis "An approach to developing IS of telecommunication company which is based on the models", during 2012.

Received: August 23, 2011; Accepted: March 30, 2012.