Journal of Engineering and Applied Sciences 13 (2): 493-500, 2018
ISSN: 1816-949X

Generative Automatic Matching Between Heterogeneous Meta-Model' Systems

[1]Zouhair Ibn Batouta, [1, 2]Rachid Dehbi, [1]Mohammed Talea and [1]Omar Hajoui
[1]Faculty of Science Ben M'Sik, LTI Laboratory, Hassan II University, Casablanca, Morocco
[2]Faculty of Science Ain Chock, LIAD Laboratory, Hassan II University, Casablanca, Morocco

Abstract: Building computer systems has become increasingly difficult, this is essentially due to the great number of existing solutions. The aim of this study is to propose a new approach allowing the matching between meta-models of different systems, this will allow the generation between models conforming to these connected meta-models. First, we will elaborate a taxonomy study on existing approaches, then we present the architecture of our generative matching approach named GAM (Generative Automatic Matching), after that, we will introduce a case study explaining our approach. Finally, we will conclude by a SWOT analysis between the different matching approaches.

Key words: Matching, automation, different matching approaches, generative automatic matching, meta-model, SWOT analysis

INTRODUCTION

Today, computer systems have become more important in everyday life, these systems have become more and more complex as well as their realization. From these facts, a new has emerged, namely the Model Driven Engineering (MDE) (Batouta *et al*., 2015, 2016), the goal being to adopt model-driven engineering instead of code-driven engineering. Model-driven engineering focuses on some of the more important aspects of models. Indeed in this approach, the model no longer relies on a simple contemplative vision whose objective is to improve the documentation and the specifications, the model passes to a productive vision aiming the generation of code for a given platform.

Although, MDE has made a significant contribution to today's global software engineering, there are still many challenges that need to be addressed, indeed, the application of the MDE leads to the creation of a large number of DSLs (Batouta *et al*., 2015) in the absence of a universal consensus that governs their creation, several meta-models are created having similar or complementary uses and objectives, hence, the need to achieve an approach that allows to automatically match all these heterogeneous meta-models but not only, at the same time it must allow the automatic generation between these meta-models, it is the proposed approach that we have called Generative Automatic Matching (GAM). The goal of this approach is therefore to allow the automatic generation of a global system consisting of different DSLs based on the matching (i.e., the correspondence through relationships and links) between the different meta-models constituting the system. In our previous study, we presented an approach called Tertiary and Systematic Mapping Review (TSMR) (Batouta *et al*., 2016), one of the results found is that between 49 and 85% of the studies offer their own code generation platforms by creating new DSLs (Fig. 1 and 2), this alarming finding shows even more the urgency and the need to propose an approach

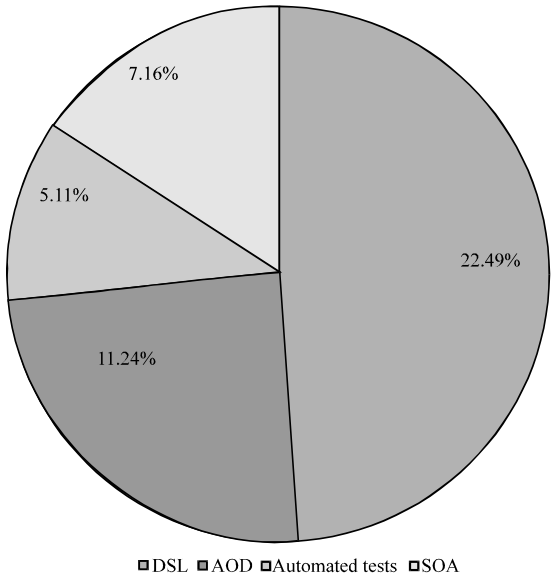

Fig. 1: SMS percentage of primary studies per technique

Corresponding Author: Zouhair Ibn Batouta, Faculty of Science Ben M'Sik, LTI Laboratory, Hassan II University, Casablanca, Morocco

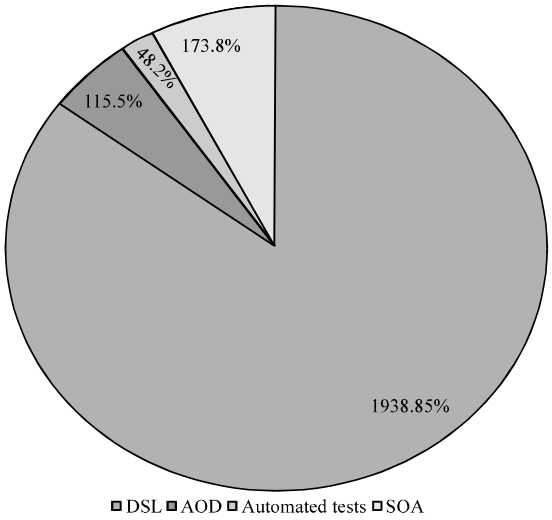

Fig. 2: TR percentage of secondary studies per technique

that permit both to match the meta-elements of distinct and heterogeneous meta-models and allow the automatic generation of models conforming to these linked meta-models.

MATERIALS AND METHODS

Taxonomy: Matching is the process of finding correspondences and relations between elements of two distinct schemas or models which is a fundamental problem in software engineering, be it in the areas of matching schemas, models or database applications, such as integration, data warehousing and semantic query processing.

In current implementations, correspondence is usually done manually which has significant limitations. On the other hand, previous research papers have proposed several techniques for performing partial automation of the matching operation for specific application domains. We present a taxonomy that covers many of these existing approaches, we will start by defining the existing matching techniques.

Matching techniques: In the literature, we have found several techniques for calculating the correspondence between the models, even if these techniques don't necessary deal with generating artifacts between the linked elements. We can classify them into four essential techniques.

Static Identifier Based technique (SIB): In this technique, it is assumed that each element of the model has a unique persistent and nonvolatile identifier that is assigned at the creation, this identifier is called UUID (Universally Unique Identifier) or GUID (Globally Unique Identifier). Therefore, the basic approach for mapping between models is to identify the corresponding model elements based on their corresponding identities (example, IBM Rational Software Architect RSA).

Signature-based techniques SIG: In this technique, the identity of each model element is not static but instead it is a dynamically calculated signature Kompose (Morin *et al*., 2008).

Similarity-based technique SIM: While the two previous approaches deal with the problem of model matching as equality or true/false identity (whether static or dynamic), the category of similarity-based approaches treats artifacts as a graph of typed attributes and try to identify related items based on the aggregate similarity of their features. Typical examples of this category of approaches are SiDiff (Schmidt and Gloetzner, 2008) and the integrated approach of EMF compare (Lehoux and Vallee, 2004), the similarity flood algorithm presented by Melnik *et al.* (2002) and DSMDiff (Lin *et al*., 2007) (which also incorporates the pairing by signature).

Custom Specific Language technique (CSL): This category use matching algorithms adapted to a specific modeling language; typical examples of this category of approaches are UMLDiff (Xing and Stroulia, 2005) and the research by Nejati *et al.* (2007) that specifically target UML Models. Approaches such as EMF compare (Toulme and the Epsilon Comparison Language (ECL) (Kolovos, 2009).

Generative automatic matching approach: This notion of generative matching or generative automatic matching has never been used before in the literature; indeed, most of the approaches that have dealt with the problem of heterogeneities of the meta-models dealt only with the matching part between the models or even the meta-models without treating the generation component between the models conforming to these meta-models (Morin *et al*., 2008; Schmidt and Gloetzner, 2008; Lehoux and Vallee, 2004; Melnik *et al*., 2002; Lin *et al*., 2007; Xing and Stroulia, 2005; Nejati *et al*., 2007; Kolovos, 2009) which presents the strong point of our approach. In this study, we will present our new approach GAM that addresses the problem of increasing heterogeneity of solutions, The aim of this approach is to allow the automatic generation of a system based on the matching between the different meta-models constituting the system, this approach is compose of two phases:

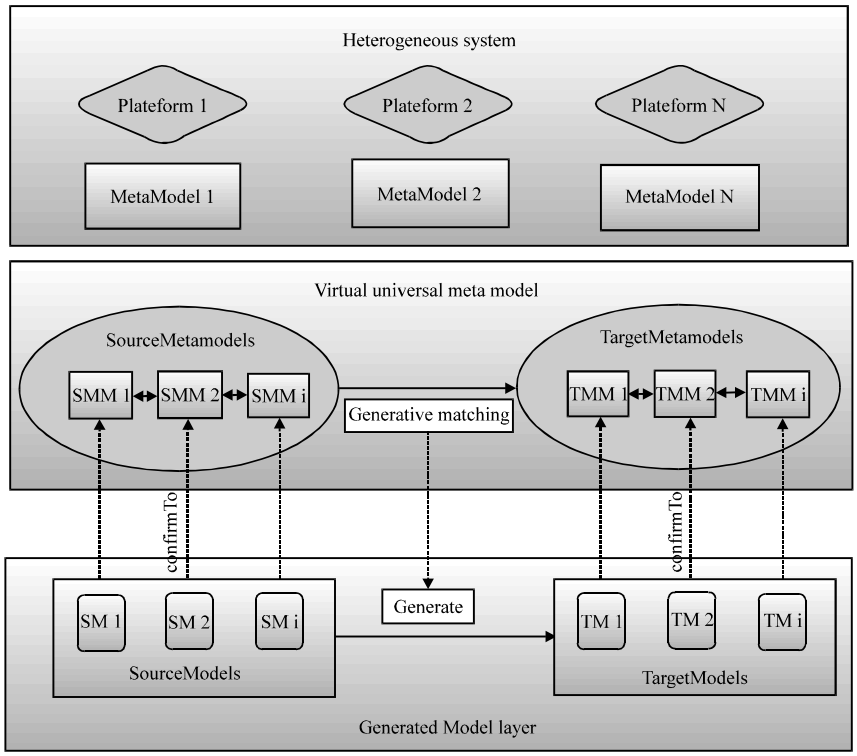

Fig. 3: Global architecture of the generative matching approach

The first step of our approach allows the linkage between completely heterogeneous meta-models, not just the linkage between level 1 schemes or 3 instances and models or even meta-models dealing with the same domains or representing different meta-models of the same version as is the case for traditional matching approaches for example this approach encompasses the alignment between database meta-models (relational or big data) and meta-models representing schemas as XML meta-models and meta-models of different object-oriented languages like c# or java.

The second step of our approach is the treatment of a second essential component, namely automatic generation between models, indeed, the result of the matching is exploited to generate models called target of a final system from source models and that based on the automatic matching found in the first step.

Figure 3 shows the overall architecture of our approach. As shown in Fig. 3, the set of platforms are seen as a heterogeneous global system, consisting of exogenous meta-models representing various domains. The first step of the approach is to constitute a universal virtual meta-model consisting of source meta-models SMM1, ..., SMMi which will be matched with TMM1, ..., TMMj target meta-models. The matching process is explained in detail in this study. The resulting matching will allow descending to layer 1; indeed, the approach will allow, from source input models SM1, ..., SMi compliant with the source meta-models, to generate target models TM1, ..., TMj conforming to the target meta-models. The following section explains the process of our approach.

Gam meta-model: To realize our approach, it is essential to design a meta-model that will make it possible to identify the basic concepts that will be treated by our approach. For example, the concept of elements constituting source and target meta-models, the relations between them, the types of links and correspondences possible between two or more elements of the different meta-models, the management of the versions of the correspondences and also of the history of pairing (matching between two meta-models). We call this meta-model MMG (Generative Matching Meta-Model). The core of the MMG is presented in Fig. 4, it introduces the following notions.

Element: Which is the generalization of all the other elements, this element contains the attributes:

- Name which characterizes the name of the element
- Id which represents the identifier of the element
- Description which contains description information

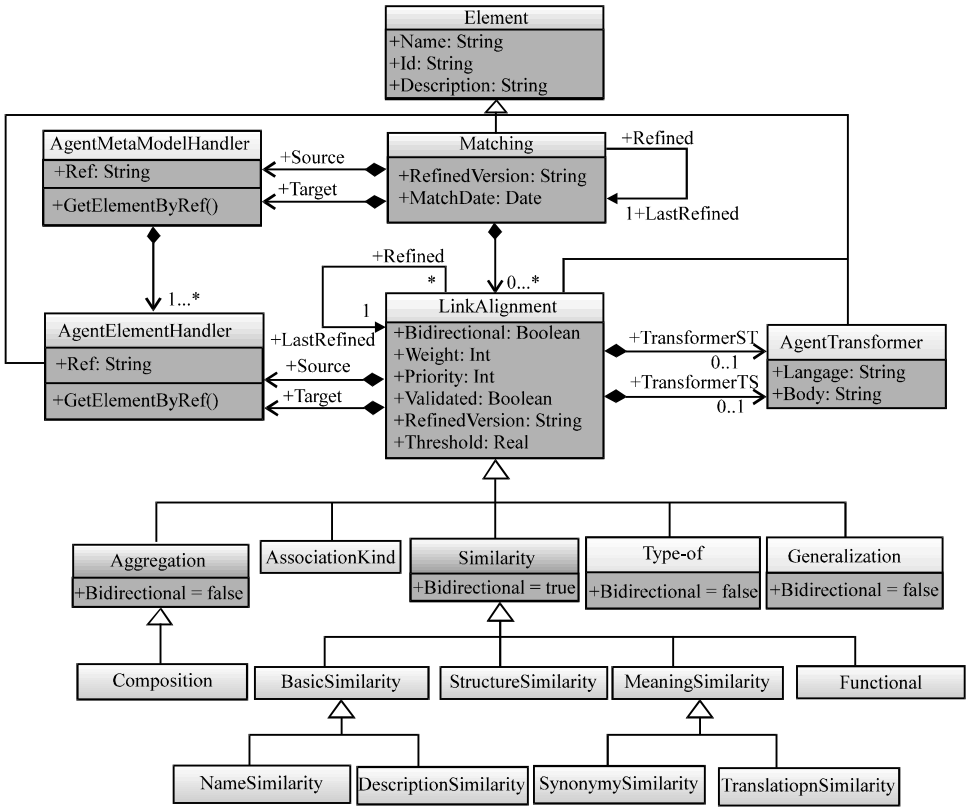

Fig. 4: Core of the generative matching meta-model

Matching: The main element that contains all relationships between source and target meta-models. It contains:

- refinedVersion: allows managing the different versions and refinements for the same matching between the same source and target meta-models
- matchDate: makes it possible to store the date of the match
- The different versions are linked together through a reflexive relationship identified by the two roles refined and lastRefined
- Source: identifies the source (meta) model
- Target: identifies the target (meta) model
- AgentMetaModelHandler: allows manipulation of source and target meta-models using their references
- AgentElementHandler: allows handling meta elements of the meta-models
- AgentTransformer: this element permit the transformation between the source and target elements linked by a relation of similarity, it contains:
  - Language: determines the language used or the transformation (c #, java, ATL, ...,)
  - Body: determines the script of the transformation written in the specified language
  - TransformST: indicates that the transformation is directed from the source element to the target element
  - TransformTS is used in the case of bidirectional relationships to also indicate the inverse transformation from the target element to the source element

LinkAlignment: It is used to determine the type of correspondence between two or more meta-elements, it is extended into several types of 5 relation, namely aggregation, AssociationKind, Type-of, generalization and similarity, it contains the following fields:

Bidirectional: It indicates whether the relationship is bidirectional or not, it has two possible values: true (bidirectional) or false (one direction from the source element to the target element).

Weight: It represents the weight allocated to this type of correspondence, this notion of weight is used to manage

the conflicts between the different correspondences found as well as for the calculation of a global similarity.

Priority: It allows to assign a priority to each correspondence, it makes it possible to highlight a type of correspondences compared to the others especially in the cases of conflicting correspondences.

Validated: This field allows the validation of a match found.

refinedVersion: It allows storing the version of the correspondence in case of refinement.

Threshold: This is the threshold from which the matching of the relationship can be accepted.

Aggregation: A non-symmetric relationship, it expresses a strong coupling and a relationship of subordination; an element is part of another element.

Composition: The composition is a strong aggregation, the container and the contents are linked structurally and physically, the life cycles of the elements are linked: if the container is destroyed (or copied) its components are also. By transitivity, the composition is a non-bidirectional relationship.

Generalization: It allows the classification of elements, it is used when an element is a special case of another element; it is a non-bidirectional relationship.

Type-of: This relation determines that an element A is of type B.

AssociationKind: It is the generic relationship between two elements, the role of this element is very important: indeed, we cannot represent in a single Kernel meta-model all the existing types of relationships, hence, the importance of this element which must be extended to add other types of possible relationships used in the input meta-models, this will allow to have a new refined meta-models MMG.

Similarity: Is the key element representing the correspondences between the meta-elements source and target in order to present an hybrid similarity computing, we distinguish in our meta-model four types of relations of essential similarities, namely BasicSimilarity, MeaningSimilarity, StructuredSimilarity and Functional, this element can be expanded to add other kinds of similarities in the refined matching meta-model MMG which makes our approach adaptive. BasicSimilarity, MeaningSimilarity and Structured Similarity.

BasicSimilarity: It represents the lexical correspondence between two elements, this element is extended into two types of similarities, NameSimilarity and Description Similarity.

NameSimilarity: Similarity between elements based on lexical comparison between names.

DescriptionSimilarity: Similarity between elements based on the lexical comparison between descriptions.

StructuredSimilarity: Similarity between elements based on structural comparison between elements.

MeaningSimilarity: Similarity dealing with the semantic aspect of elements, it is extended into two types, Synonymy Similarity and TranslationSimilarity.

SynonymySimilarity: Links the source and target elements having the same meaning, taking into account their belonging to the same language.

TranslationSimilarity: Matches elements belonging to distinct languages and having the same meaning.

Functional: Links source and target elements with the same functional meaning, this type of similarity are generally added by the expert. After detailing the kernel of the meta-model of generative automatic matching MMG used by our approach, we will in the next section detail the GAM process.

GAM process: The GAM's process is composed as shown in the two figures of the following steps (Fig. 5 and 6):

Step 1: Selection of source and target meta-models.

Step 2: Refinement the kernel meta-model MMG presented in Fig. 4, we note the use of the Cloud Store for the storage of the different refined versions of the models used, this allows access at anytime and anywhere to different data.

Step 3: Generation of the MG Model (Generative Matching Model), MG is conform to the meta-model MMG, this model contains the different correspondences

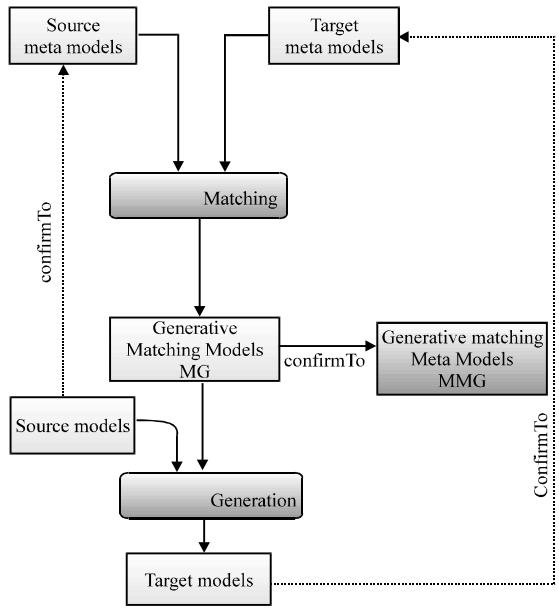

Fig. 5: GAM 's generic process

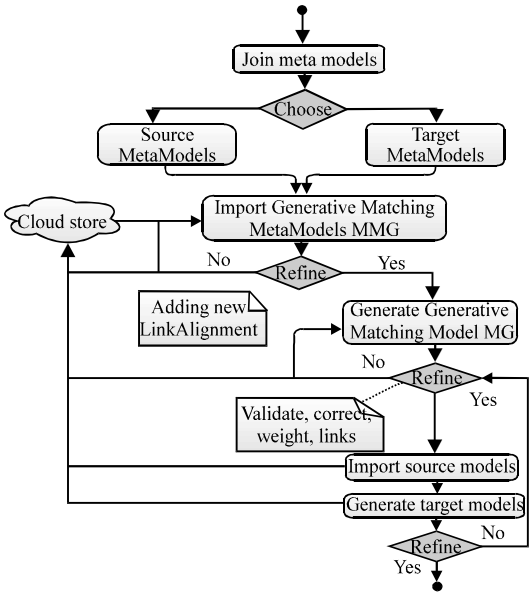

Fig. 6: GAM's detailed process

between the elements of the two input meta-models, it must be refined to correct the correspondences detect in the first iteration, this refinement can be done by executing other matching iterations or by the direct intervention of the expert to validate or cancel correspondences.

Step 4: In this phase, one (or more) source model conforming to the source meta-model is used to generate the target model (or models), the target model can be refined automatically modifying the MG or manually by the expert intervention.

RESULTS AND DISCUSSION

Case study: The following case study explains the application of our approach on two distinct source and target meta-models, it include the generation from a source model (conforming to the source meta-model) to a target model (conforming to the target meta-model).

Figure 7 shows the source meta-model and Fig. 8 shows the target meta-model: Fig. 9 shows the bidirectional similarities found (in dashed arrows): Figure 10 presrent the source model conforming to the source meta-model and its generated target model.

V SWOT analysis of existing matching

Approaches: In this study, we will present a SWOT analysis between the different matching approaches. Since, we are interested in the heterogeneity of the DSLs and meta-models, we focus this analysis on two categories of approaches:

- Model based approaches: these approaches allow the link between models at level 1 and instances
- Meta-model based approaches: these approaches allow the link between meta-models (level 2)

Graph-based and schema-based approaches like SIDIFF or agreement maker are untreated in this SWOT analysis.

Table 1 summarizes the existing approaches regarding their technique. We will now present in Table 2 the summary of the SWOT analysis containing the advantages and disadvantages of each of the existing approaches based on their used techniques.

As shown in Table 2, the existing matching approaches show multitudes of gaps in correspondence between (meta) models, namely the use of fixed heuristics and dealing with a single type of correspondence and relations in addition the correspondence is set only between similar and non-heterogeneous models, furthermore, the matching is often done in a manual and not automatic way, without forgetting that these approaches do not exploit the results of the links found to automate the generation from a model towards the others.

Table 1: SWOT analysis

	Approaches	
Technique	Model based	Meta-model based
Custom Specific Language technique (CSL)	ECL; UMLDiff	AMW
Static Identifier Based technique (SIB)	RSA	
Similary-based technique (SIM)	DSMDiff; Kompose	AML; EMFCompare; Falleri; Matchbox; SAMT4MDE
Signature-based technique (SIG)	DSMDiff; Kompose	

Table 2: SWOT analysis

Variables	Statics Identifier Based technique (SIB)
Positives	Fast implementation is based directly on UUIDs for correspondence
	No configuratonfrom the user's point of view
Negatives	Does not apply to heterogeneous models built independently
	Often applied to compare two models built from another ancestor model
	Inability of this technique to adapt to changing models and new versions, especially in the case of deletion and recreation of the same element which directly leads to the change of its UIID
	Low adapatability: Not applicable for model representation technologies that do not support the maintenance of unique UUID identities
	Does not handle the automatic generation between models
	Matching relationships are often created manually
Signature-based techniques (SIG)	
Positives	Compare models that have been built independently
Negatives	User-side effort: Specify a series of functions that calculate the identity of different types of model elements
	This kind of match is not appropriate for all types of model elements
	Does not handle the automatic generation between models
	Mapping relationships are often created manually
Similary-based techniques (SIM)	
Positives	Compare models that have been built independently
	More acceptable matches
Negatives	Use of fixed and non-hybird computational heuristics (dealing with a signal type of match)
	Does not handle the automatic generation between models
	Mapping relationships are often created manually
Custom Specific Language Technique (CSL)	
Positives	Ability to integrate semantics
Negatives	Specify the correspondance algorithm manually
	Fixed and non-hybird heuristics

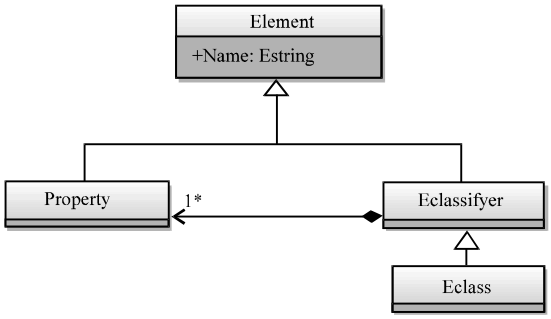

Fig. 7: Source meta-model example

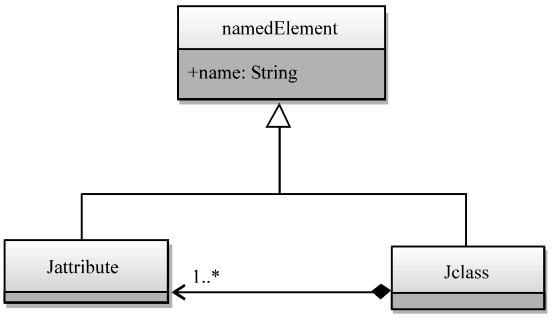

Fig. 8: Target meta-model example

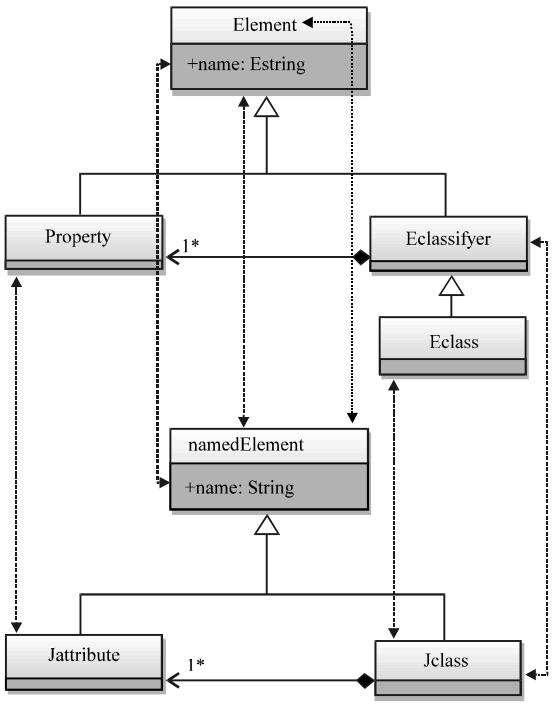

Fig. 9: Matched similarities for our case study

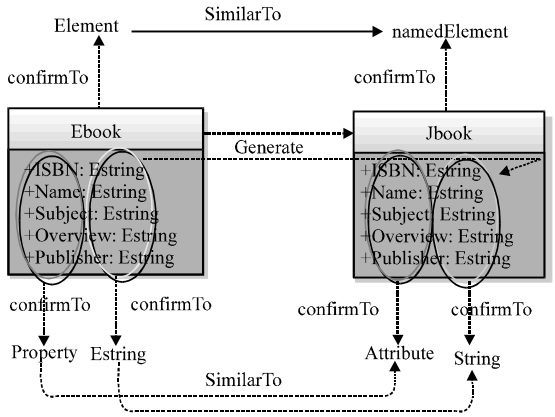

Fig. 10: Generation example between source and target models

CONCLUSION

We presented in this study the architecture and the essential components of our approach called GAM allowing the generative matching between distinct meta-models, we also presented a taxonomy of existing approaches and a case study explaining our approach and we finally completed a SWOT analysis between the different matching approaches. We are currently working on the implementation of the GAM approach using. Net platform in an environment consisting of heterogeneous big data databases; the solution architecture is based on MAS (Multi-Agents System).

REFERENCES

Batouta, Z.I., R. Dehbi, M. Talea and O. Hajoui, 2015. Multi-criteria analysis and advanced comparative study between automatic generation approaches in software engineering. J. Theor. Appl. Inf. Technol., 81: 609-620.

Batouta, Z.I., R. Dehbi, M. Talea and O. Hajoui, 2016. Automation in code generation: Tertiary and systematic mapping review. Proceedings of the 4th IEEE International Conference on Information Science and Technology (CIST'16), October 24-26, 2016, IEEE, Tangier, Morocco, ISBN:978-1-5090-0752-3, pp: 200-205.

Kolovos, D.S., 2009. Establishing correspondences between models with the epsilon comparison language. Proceedings of the 5th European Conference on Model Driven Architecture-Foundations and Applications, June 23-26, 2009, Netherlands, pp: 146-157.

Lehoux, N. and P. Vallee, 2004. [Multicriteria Analysis]. Ecole Polytechnique de Montreal, Montreal, Quebec, (In French).

Lin, Y., J. Gray and F. Jouault, 2007. DSMDiff: A differentiation tool for domain-specific models. Eur. J. Inf. Syst., 16: 349-361.

Melnik, S., H. Garcia-Molina and E. Rahm, 2002. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. Proceedings of the 18th International Conference on Data Engineering, February 26-March 1, 2002, IEEE, San Jose, California, USA., pp: 117-128.

Morin, B., J. Klein, O. Barais and J.M. Jezequel, 2008. A generic weaver for supporting product lines. Proceedings of the 13th International Workshop on Early Aspects, May 12, 2008, ACM, Leipzig, Germany, ISBN:978-1-60558-032-6, pp: -18.

Nejati, S., M. Sabetzadeh, M. Chechik, S. Easterbrook and P. Zave, 2007. Matching and merging of statecharts specifications. Proceedings of the 29th International Conference on Software Engineering (ICSE'07), May 20-26, 2007, IEEE, Minneapolis, Minnesota, USA., pp: 54-64.

Schmidt, M. and T. Gloetzner, 2008. Constructing difference tools for models using the SiDiff framework. Proceedings of the Companion of the 30th International Conference on Software Engineering, May 10-18, 2008, ACM, Leipzig, Germany, ISBN:978-1-60558-079-1, pp: 947-948.

Xing, Z. and E. Stroulia, 2005. UMLDiff: An algorithm for object-oriented design differencing. Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering, November 07-11, 2005, ACM, Long Beach, California, USA., ISBN: 1-58113-993-4, pp: 54-65.