A Facet-based Model Mapping Method for EA Alignment and Evolution

Jonathan Pepin[1,2], Pascal André[1], Christian Attiogbé[1], and Erwan Breton[2]

[1] AeLoS Team LINA CNRS UMR 6241 - University of Nantes, France
{firstname.lastname}@univ-nantes.fr
[2] Mia-Software - Nantes
ebreton@sodifrance.fr

Abstract Information System evolution requires a well-structured Enterprise Architecture and a rigorous management. The alignment of the architecture elements according to various abstraction layers contributes to the management but appropriate tools are needed. We propose improvements of the Facet approach and the accompanying tools to master the problem of Enterprise Architecture alignment and evolution. The tools have been experimented on real life cases and are integrated in the Eclipse EMF Facet project.

Keywords: Enterprise Architecture, Alignment, Evolution, Legacy Information Systems, Model Mapping

1 Introduction

Enterprise Architecture (EA) engages much interest by academia and practitioners during the last decade and numerous methods such as TOGAF or Zachman, have been proposed [10]. Modeling languages such as Archimate have been proposed [16] to enable the specification of architectures from business goal to technology infrastructure. However technologies and tool support are still missing [15,4,11], especially when we consider the maintenance of legacy systems [8].

The information technology (IT) evolution is driven by technological upgrade steps and by continuous business change imposed by market competition or law obligations. These are expensive maintenance activities. Mastering the evolution of information systems (IS) according to an EA approach requires: (i) to reduce the distance between the business and its IT (the Business/IT Alignment (BITA) problem), (ii) to master the transition between various IS versions in order to compare different evolution scenarii before deciding the one to develop. While numerous contributions emerged to answer the BITA problem [4,6] by providing means to link different abstraction layers, assisting the maintenance of legacy systems is still challenging [8]. Our motivation is to help decision makers to model the alignment of legacy systems with the related business models in order to put in evidence the cross effects of IT and business evolutions. We focus on operational concerns rather than on a theoretical vision. From the model-driven

In: S. España, M. Ivanović, M. Savić (eds.): Proceedings of the CAiSE'16 Forum at the 28th International Conference on Advanced Information Systems Engineering, Ljubljana, Slovenia, 13-17.6.2016, published at http://ceur-ws.org

engineering (MDE) point of view, alignment and evolution is viewed as *mapping models*, provided that business and IT models are available at different EA layers.

The contribution of this paper is a method to represent EA alignment and evolution using MDE and its de facto standards such as OML, OCL, MOF, EMF. We propose a method for mapping models using an improved *Facet approach*. Our proposal is accompanied with tools and we lead experimentations to show the method effectiveness. This work is part of a general EA maintenance proposal [14].

The article is structured as follows. In Section 2, we compare different mapping techniques according to criteria linked to EA alignment and evolution concerns. The Facet approach is the most promising but must be improved. Section 3 presents our enhanced Facet method that covers all the criteria. Section 4 illustrates our method on real-life case studies. In Section 5 we discuss related works. Finally, Section 6 concludes the article and draws some perspectives.

2 Model Mapping Techniques for Enterprise Architecture

Model mapping helps in automating the process of combining models one with another to get dedicated and consistent views of a system. Considering alignment and evolution, we retain five techniques: extension, merging, weaving, annotation and facets. We compare them according to four criteria:

- Non-intrusiveness: the mapping must not modify the individual models,
- Ontology: the mapping supports a semantic relation,
- Link Resolution: the mapping includes navigation between models,
- Serialization: the links mapping must be persistent.

Extending meta-models Each meta-model is extended in order to include the concepts from the other related meta-model. Accordingly the common concepts of the two meta-models are mixed. However, this approach violates the construction of the Domain Specific Language (DSL) supported by a meta-model; another drawback is to anchor the concepts to be aligned while EA needs flexibility. Consequently, any evolution of the EA meta-models will involve the creation of an extended meta-model for each new version.

Merging meta-models Merging creates a global meta-model including all the concepts from the individual meta-models. The main drawback is that a unique meta-model is less flexible to evolution and may lead to confusions. Merging only consists in modifying the initial meta-model by adding explicitly new attributes.

Weaving models Model weaving is the most flexible and non-intrusive mapping technique. Model weaving involves the creation of an independent model which refers to the models to weave and the links between the considered model concepts *e.g.* Atlas Model Weaver [9] and Virtual EMF [3]. However this solution is still limited: the created links have yet no conceptual meaning, they are generic and can store any kind of concept; the navigation by resolving link is expensive because it requires a complete traversal of the models mapping graph.

Annotating models Annotation facilities enable one to add independent cross-cutting concerns between models. `MAnnotation`[3] is a small meta-model inspired by the Ecore `Eannotation`. Creating annotations is very simple and link resolution (model navigation) is supported natively by the EMF standard implementation. But, there is the same issue as in model weaving: the links have no conformance rules, each concept can be linked with any other one without avoiding meaningless annotations.

EMF Facet Facet is a technique to extend a meta-model without intrusion; it is implemented in the EMF Facet open-source project. A Facet natively provides a mechanism to virtually add attributes or operations as new features without modifying the initial meta-model. However, a new feature systematically calls a query that returns a value. Queries must be executed during the model loading, with a considerable impact on the response time. Another shortcoming is that new features cannot be valued manually, as classical attributes or references.

Each facet has at least a name and a meta-class type. A Facet optionally extends an existing Facet and refers to the base meta-class by its absolute Universal Resource Identifier (URI). A Facet has three kinds of features: *FacetAttribute*, *FacetReference* and *FacetOperation*. A *FacetReference* has a name, a multiplicity, a type (a reference of the current Facet set or any Ecore reachable meta-model) and an opposite reference if the relationship is bidirectional. A *FacetAttribute* has a name, a multiplicity and the meta-class type as in *FacetReference*.

For example, suppose an *Application* meta-model describing the IT component architecture and a *Business Process* meta-model describing the business side of the information system. A new string FacetAttribute *artifactUri* is added to the *Activity* meta-class of *Business Process*, it stores the artefact URI. If we want to align models, we add a new FacetReference to *Activity* that establishes a bidirectional link between *Activity* and the *ApplicationComponent* meta-class from the *Application* meta-model.

Comparison We experimented all these techniques in the context of EA alignment and evolution, we summarise the comparison results in Table 1.

Table 1. Comparison between mapping techniques

Technique	Nonintrusive	Ontology	Link Resolver	Serialization
Merge	✗	✓	✓	✓
Extension	✗	✓	✓	✓
Weaving	✓	✗	✗	✓
Annotations	✗	✗	✓	✓
Facets	✓	✓	✓	✗

The Facet approach is the most promising one despite its main drawback. It requires to perform manual weaving and to serialize mappings in an intermediate models. Section 3 addresses this shortcoming.

[3] Mia-Software Company`http://www.mia-software.com/`

3 An Enhanced Facet Method and Support

To fix the EMF Facet limitations, we modify the meta-model, the manager and the serialization mechanisms. Then we extend the tooling facilities.

EMF Facet Extensions We add *FacetAttribute* and *FacetReference* accessors to avoid the systematic query computation. Enabling value access turns into weakening the multiplicity of *FacetAttribute* and *FacetReference* that extend *DerivedTypeElement*. Thus the multiplicity 1..1 of the eReference *query* on *DerivedTypedElement* is replaced by 0..1. A new eReference named *fOpposite* creates a reflexive reference mechanism like *eOpposite* on *EReference* in the Ecore meta-model. We implemented a new behaviour of the Facet engine called FacetManager. To store the new virtual attributes and references without queries, we extended the existing mechanism of Facet persistency, the SerializationManager engine. The feature values are handled with the *Property View*, a tree viewer built on the Eclipse Modisco TreeEditor compatible with EMF Facet. However, TreeEditor is a read-only viewer. We developed a new *Property View* to bypass the read-only restriction; enabling *FacetSet* virtual features in addition to the EMF native ones.

EA Model Weaver The ergonomics of the *Property View* is too limited to provide users assistance to map model elements to each other. We therefore developed a specific weaving editor with multiple views as illustrated by Figure 1.

On the right part of the figure, an outline displays the different models to weave. On its left part, a specific view organizes the weaving result by facets. This design allows to drag and drop elements from right to left to link elements by references corresponding to the *FacetSet* definition.

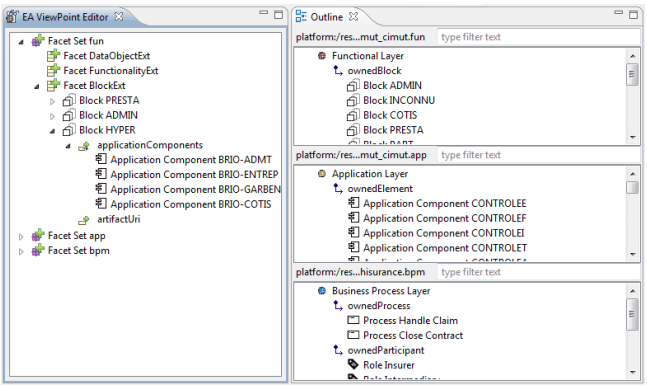

Figure 1. EA model weaving editor

Navigation and Query Using the links stored in *FacetReferences*, one can navigate through the various meta-model *e.g.* browsing bottom-up or top-down is an EA alignment or browsing the various version of an EA evolution. The *TreeEditor* encompasses navigation. Architects also require facilities related to quality *e.g.* define various kind of alignment or traceability metrics to drive a dashboard. OCL is the standard language to express statements including model navigation. We extended the query engine in order to make Facet virtual feature compatible and used as property in OCL statements.

All these extensions are included to the Eclipse *EMF Facet Project*.

4 Assessment of the Method with EA Case Studies

A previous paper [14] introduced our alignment method (models, process and tools). We recall only the elements mentioned in the experimentations.

The EA Framework The core of alignment is a triple of generic meta-models:

1. The application meta-model App which depicts an abstract view of the software architecture in terms of components, services and data. App is compatible with models like UML, WSDL, SCA, Archimate... *This model is built from legacy code by reverse-engineering steps (bottom-up).*
2. The business meta-model BPM depicts the IS business processes in terms of actors, activities, tasks and data. It is compatible with standard notations such as BPMN, UML activity. *The business model is an embodiment of the business strategy (top-down) provided by business architects.*
3. The functional meta-model Fun depicts the urban view of the information system in terms of functional blocks and sub-blocks (zone/district/plot) and data stores. *Usually this model is provided (or not) by enterprise architects.*

The interrelationships between the three meta-models support our Business-IT alignment method and the forthcoming evolution, according to two kinds of links: processing and data. A *Link* maps a concept of one layer to another similar concept of a different meta-models, as explained in [14]. The three meta-models App, BPM and Fun are implemented with Eclipse EMF by their corresponding Ecore models. The meta-models interrelationships are implemented by a specific *FacetSet* with *FacetReference* for each Processing and Data link.

Case Studies The experimentations aim to test the applicability of our method (framework, Facet mappings, tools) to real-life case studies with a special focus on alignment. We have big size legacy code, partial documentation, loose traceability between models and code, etc. The experimentations are led with three case studies (SAMM, SAMI and SAMUT) provided by french insurance companies. Each case includes specific models and concepts that cover all or a part of our three meta-models. Table 2 summarises the number of concepts obtained after model extraction. An empty cell value means no input data in the case study. The last column describes the mapping step: manually with our weaving editor or automatically with a model transformation.

Table 2. Case studies Comparison

Case Study	Fun		BPM		App					Weaving
	Block	Fun	Process	Activity	Comp.	Func.	Service	Interf.	DataObj.	
SAMM			360		1 002	4 808	2 334	502	4 203	Manual
SAMI	18	131	167	268	625		11 894			Auto
SAMUT	12				1 045				669	Manual

SAMM is composed of a Java source code with 33 4000 classes and a web portal exported from MEGA Enterprise Architecture[4] repository. The App

[4] http://www.mega.com/en/solution/business-architecture

model has been automatically reverse engineered, according to the bottom-up transformation described in [14]. This model is huge and loading it was a scalability challenge. The repository contains only a documentation of the business process diagrams (HTML pages and images) because the MEGA source files were not available. Consequently we specified manually only twelve representative diagrams in the BPM model; we mapped it to the extracted App instance. The original Mega had no connection to applications concepts, our Facet mapping enables to navigate from processes toward services.

SAMI is composed of a MEGA repository with available source files, containing both application, functional and process informations. We exported all the MEGA concepts to feed our models (App, BPM, Fun) and make an automatic alignment by transformation between the different layers. More precisely the transformation creates the *FacetReferences* from links that exist in the repository. This case reveals that our method can re-interpret an existing alignment by transformation. Also, the new mapping with Facet enhanced the possibility of information system evolution.

SAMUT is a legacy code only without functional or business information. The maintenance scenario is written from scratch. To fill the application model, we apply reverse-engineering to get the data objects from an old hierarchical database and the application components deduced from *SQL Stored procedures*. Next, an architect audited the employees and the business responsible of SAMUT to identify and model the functional blocks; then we mapped manually the Fun blocks and the App components with *FacetReferences*. Our tools really helped the architect to create the mapping, because he usually uses a spreadsheet program to store the namespaces. The OCL queries enable to identify orphan components (not classified in blocks) to refine the mapping.

Lessons Learned The experimentations validated by the architects of our company (Mia-Software), showed that our Facet method is adequate to handle real-life legacy applications. It provides to architects a generic, open, automated, scalable and efficient solution that replaces the current spreadsheet based practice.

Genericity: It was convenient for the different cases since we did not need to rewrite *FacetSet* rules. The Facet tool support is applicable in wider set of cases: up to a programming effort, an architect can specialize the *FacetSet* to its own EA meta-models or to existing standards *e.g.* BPMN2, UML or Archimate.... Facet adaptation and specialization are less time consuming than the other mapping approaches of Table 1.

Variability: The method accepts heterogeneous inputs. Each case study is specific because the quality of the legacy information is different from one organisation to another: partial/complete, informal/formal, EA maturity, process management,... Customized analysers are needed, depending on the input format; we wrote model transformations to translate XMI files into readable EMF inputs.

Automation: As soon as models are available one can apply model transformations. However the automation rate is never 100 % and manual processing is necessary to complete missing information, to capture missing abstraction *e.g.* mining missing components, to solve ambiguous mappings.

Efficiency and Scalability: The experiments showed that big mappings are hardly manageable by humans. Our weaving editor provides a friendly user interface to define mappings, including a search engine, model navigation and syntax highlighting. Additional tool is needed to visualize big mappings, to evaluate the mapping properties (consistency, completeness) and quality (misalignment, evolution traceability...). Our Facet query engine is a first step to reach this goal.

5 Related works

Model composition covers many categories of works; our technique belongs to the *model-based correspondence* category in the classification of Clavreul [7]. Chen et al. proposed a mapping between EA data sources in a repository [5]. They work on a semantic level and take into account conflicts but it is rather a merging approach and tools are missing. Limyr et al. propose a visual tool to map non-intrusively models (concepts and attributes) [12]. Useful to write small model transformations, it cannot handle big EA models.

Using MDE for enterprise system has been explored by other authors. Atkinson et al. preserve the principle of separation of concerns with an annotation mechanism [2]. The Open Source Melanee Tool is a workbench for creating domain-specific languages. But we did not find how to apply extension at runtime to an existing EMF model like our method with Facet. By the way it confirms that UML is not the appropriate notation because UML Profiles bring too much complexity to the extension intention.

Our EA framework does not fully include the strategic level of BITA, like *e3value*, *i** or *SEAM*. Meertens *et al.* describe a mapping between Business Model Canvas (including requirements) and Archimate [13]. A cross cutting contribution is to accept BMC as an input of our method and suggest the authors to implement their mapping with our technique. Note that our generic meta-models are kinds of pivots on which one can define bridges from and to standard notations (we did it for MEGA) in order to accept wider EA frameworks and models.

Beside MDE, mapping issues also exist when integrating the EA from different companies *e.g.* in case of fusion of companies. In [1], Anaya et al. represent impact and causality relationships but they do not propose solutions for practitioners.

6 Conclusion

Models are essential to represent the numerous concepts of Enterprise Architecture. These models need to be mapped to elicit alignment and evolution issues. We have proposed a non-intrusive technique, built on top of EMF Facet, to instrument model mapping that covers all the criteria we specified for EA (Table 1). We successfully submitted it as a contribution to the open-source Eclipse EMF Facet[5]. The experimentations on real-life case studies reveal that our method and tools are helpful to EA architects during the maintenance of legacy systems.

[5] https://bugs.eclipse.org/bugs/show_bug.cgi?id=463898

The ongoing works cover two main perspectives. A first one is to improve the automation ratio by detecting candidate model alignments. A second perspective is to exploit the evolution and alignment mappings in order to provide a dashboard including ratios, misalignment areas. We already defined fitness measures for alignment in OCL rules.

References

1. Anaya, V., Ortiz, A.: How enterprise architectures can support integration. In: Proceedings of the First International Workshop on Interoperability of Heterogeneous Information Systems. pp. 25–30. IHIS '05, ACM, New York, NY, USA (2005)
2. Atkinson, C., Gerbig, R., Fritzsche, M.: A multi-level approach to modeling language extension in the enterprise systems domain. Information Systems 54, 289 – 307 (2015)
3. Brunelière, H., Dupé, G.: Virtual EMF - transparent composition, weaving and linking of models. In: EclipseCon Europe 2011 (Nov 2011)
4. Chan, Y.E., Reich, B.H.: IT alignment: what have we learned? JIT 22(4), 297–315 (2007)
5. Chen, W., Hess, C., Langermeier, M., Stuelpnagel, J., Diefenthaler, P.: Semantic Enterprise Architecture Management. In: ICEIS Proc. pp. 318–325. SciTePress (2013)
6. Clark, T., Barn, B.S., Oussena, S.: A method for enterprise architecture alignment. In: Proceedings of PRET, vol. 120, pp. 48–76. Springer (Jan 2012)
7. Clavreul, M.: Model and Metamodel Composition: Separation of Mapping and Interpretation for Unifying Existing Model Composition Techniques. Ph.D. thesis, Université Rennes 1 (Dec 2011), http://tel.archives-ouvertes.fr/tel-00646893
8. Fischer, R., Aier, S., Winter, R.: A federated approach to enterprise architecture model maintenance. Enterprise Modelling and Information Systems Architectures 2(2), 14–22 (2007)
9. Jouault, F., Vanhooff, B., Bruneliere, H., Doux, G., Berbers, Y., Bezivin, J.: Inter-DSL Coordination Support by Combining Megamodeling and Model Weaving. In: Proceedings of the SAC 2010. pp. 2011–2018. ACM, New York, NY, USA (2010)
10. Lankhorst, M., al.: Enterprise Architecture at Work - Modelling, Communication and Analysis (3. ed.). The Enterprise Engineering Series, Springer (2013)
11. Lapalme, J., Gerber, A., der Merwe, A.V., Zachman, J., Vries, M.D., Hinkelmann, K.: Exploring the future of enterprise architecture: A zachman perspective. Computers in Industry pp. – (2015)
12. Limyr, A., Neple, T., Berre, A.J., Elvesæter, B.: Semaphore – a model-based semantic mapping framework. In: Proceedings of BPM'06. pp. 275–284. BPM'06, Springer-Verlag, Berlin, Heidelberg (2006)
13. Meertens, L.O., Iacob, M.E., Nieuwenhuis, L.J.M., van Sinderen, M.J., Jonkers, H., Quartel, D.: Mapping the business model canvas to archimate. In: Proceedings of SAC '12. pp. 1694–1701. SAC '12, ACM, New York, NY, USA (2012)
14. Pepin, J., André, P., Attiogbé, C., Breton, E.: A method for business-it alignment of legacy systems. In: Proceedings of ICEIS 2015, Volume 3, Barcelona, Spain, 27-30 April, 2015. pp. 229–237 (2015)
15. Rouhani, B.D., Mahrin, M.N., Nikpay, F., Ahmad, R.B., Nikfard, P.: A systematic literature review on enterprise architecture implementation methodologies. Information and Software Technology 62, 1 – 20 (2015)
16. The Open Group: Archimate 2.1 Specification. Van Haren Pub (2013)