Mise en correspondance de modèles hétérogènes par points de vue

Mahmoud El Hamlaoui*,**- Sophie Ebersold*- Bernard Coulette*
*Laboratoire IRIT, UTM
Toulouse, France
{mahmoud.el-hamloui,sophie.ebersold,bernard.coulette}@irit.fr
Mahmoud Nassar**
**Laboratoire SIME, ENSIAS
Rabat, Maroc
nassar@ensias.ma

Abstract

Pour permettre la compréhension et la manipulation d'un système complexe, il faut en général le raffiner en parties indépendantes. Ainsi, le modèle d'un tel système est le plus souvent obtenu à partir d'un ensemble de modèles intermédiaires qui ciblent des parties spécifiques du système. Dans la pratique, les relations entre ces modèles partiels sont soient mal identifiées (pas complètement), soient insuffisamment formalisées pour être maintenues lorsque les modèles évoluent. Ceci limite leur utilisation et ne permet pas de les exploiter pleinement dans un cadre de composition, d'interopérabilité ou encore de co-évolution de modèles. Cet article propose une approche permettant d'établir et de formaliser des liens entre des modèles hétérogènes, exprimés dans des DSLs différents. Ces liens seront exploités par la suite pour composer les modèles partiels, gérer le maintien de leur cohérence, etc. Cette proposition est illustrée par l'exemple d'un système de gestion d'anomalies (BTS: Bug Tracking System).
Mots-clés : DSL, composition, liens/correspondances, modèles/métamodèles, hétérogénéité, modèle intermédiaire, modèle global, cohérence, BTS.

1 Introduction

Pour développer des systèmes complexes, l'ingénierie du logiciel utilise depuis longtemps le principe de décomposition par points de vue [2, 7]. Dans le domaine de l'avionique par exemple, il est d'usage de développer divers modèles correspondant aux points de vue métier : électricien, thermicien, informaticien, etc. La difficulté majeure réside ensuite dans la synchronisation et le maintien de la cohérence entre ces modèles orientés point de vue et dans leur composition. En plaçant les modèles au coeur du développement des systèmes, l'IDM renouvelle cette problématique de décomposition et de composition par points de vue. Pour répondre à un besoin plus général et construire des systèmes complexes, il faut pouvoir combiner différents modèles par point de vue, réalisés par différents acteurs, et décrits dans des DSL différents, pour disposer d'une vue globale du système. Nous avons pour premier objectif de relier les modèles de points de vue du système par des relations qui seront instanciées à partir d'un méta-modèle de liens. Pour cela nous proposons un méta-modèle de liens extensible en fonction des spécificités des DSL impliqués dans la modélisation. Notre deuxième objectif concerne le maintien de la cohérence entre les points de vue. Les modèles évoluant dans le temps, la modification de l'un d'eux peut entrainer l'incohérence du modèle de correspondance d'où la nécessité de répercuter les modifications, ou tout au moins d'identifier les éléments de modèles qui seront impactés par les changements. Ces deux objectifs définissent deux champs de travail : (1) La construction des modèles de liens entre modèles hétérogènes, (2) L'exploitation des modèles de liens pour faire de la composition, de l'assistance à la gestion de la cohérence, etc. Cet article s'inscrit délibérément dans le premier champs.

Le cas d'étude que nous avons choisi pour illustrer notre approche est un système de gestion de bugs qui met en oeuvre trois points de vue et les modèles associés décrits dans les DSL suivants : SysML[13], Mantis [4] et BPMN [3]. La section 2 est dédiée à un bref panorama des travaux relatifs à cette proposition. Nous présentons ensuite l'approche adoptée (section 3) et nous montrons le principe de son application à un cas d'étude (section 4).

2 Travaux relatifs

Les approches de la littérature qui traitent de notre problématique visent essentiellement la composition de modèles. Pour cela, elles comportent généralement deux étapes : la mise en correspondance et la fusion. Nous avons étudié en premier lieu la mise en correspondance qui est faite dans ces approches. Le tableau 1 dresse le bilan de leur évaluation par rapport aux critères que nous jugeons pertinents : hétérogénéité, nombre de modèles en entrée, existence d'une synchronisation des modèles, type de représentation et mode de correspondance.

Approches	Hétérogénité	Nb de modèles en entrée	Synchronisation	Représentation à base de	Mode de correspondance
AMW	Oui	2	Non	modèles	Manuelle
ECL	Oui	2	Non	règles	Manuelle
MatchBox	Oui	2	Non	modèles	Automatique
Kompose	Non	2	Non	patterns	SA

Table 1: Comparatif d'approches établissant des correspondances (SA: Semi-Automatique)

AMW [11] permet d'utiliser des langages de transformation M2M dans un objectif de comparaison de modèles. Mais, selon ([9]) le méta-modèle générique d'AMW s'avère inexploitable pour identifier les liens de correspondance. Les développeurs se trouvent obligés d'ajouter des extensions pour définir les sémantiques. L'approche de composition EML [8] est dans la même veine qu'AMW dans le sens où il y a une pré-phase d'établissement de relations (règles de correspondance /modèle de tissage); la différence est qu'EML utilise un langage supplémentaire pour l'établissement de liens : ECL [9] (dont l'utilisation n'est pas évidente car elle requiert compétence et efforts considérables). Par ailleurs, le résultat de la phase de correspondance d'ECL est une trace de correspondance, « matching trace », qui contient les liens après exécution de l'ensemble des règles ; pour exploiter ce résultat le développeur devra ajouter des sérialisateurs afin de transformer les traces de correspondances en un modèle permettant d'exploiter le résultat à d'autres fins (ex: composition). En ce qui concerne Kompose [6], il faut noter que l'approche ne gère que les modèles homogènes. Par ailleurs, le processus de correspondance doit être paramétré, en définissant les signatures au niveau méta-modèle dans le but de définir des opérateurs de correspondance spécifiques. L'hétérogénéïté des modèles n'est pas encore prise en compte dans cette approche, et l'outillage est encore à l'état de prototype. Le système MatchBox transforme les méta-modèles d'entrée en un modèle arborescent : AMC (SAP Auto Mapping Core). Le processus se poursuit en appliquant un ensemble de stratégies de correspondance afin de produire le modèle de liens. L'inconvénient majeur du MatchBox est que c'est au développeur de définir les transformations vers le modèle AMC. Un autre inconvénient, soulevé par les auteurs, réside dans la perte d'informations lors des transformations.
En plus des inconvénients préalablement cités, les approches étudiées partagent toutes le même problème dû à l'absence de synchronisation. Pour les modèles à grande échelle, modifier des parties de (méta-)modèle, surtout si les modèles sont en phase d'exploitation, implique de re-

faire le modèle de correspondance « from scratch ». Cette tâche s'avère d'autant plus fastidieuse que l'approche de mise correspondance est manuelle.

3 Mise en correspondance de modèles hétérogènes

Dans cette section nous présentons une approche de mise en correspondance de modèles hétérogènes. Celle-ci consiste à analyser des modèles d'entrée afin d'identifier leurs différentes relations. Cette opération a été formalisée dans [5] par une fonction (Match) qui prend en paramètre un ensemble de modèles ($S = Mi;\ i = [1..n]$), recherche les liens entre leurs éléments et produit un modèle de correspondances global($C=(S)$). Les liens entre éléments de méta-modèles induisent par instanciation des liens entre éléments de modèles. Un modèle de correspondances est donc d'abord construit à partir des méta-modèles dont sont issus les modèles sources. Nous distinguons deux types de liens de correspondance : les liens entre éléments de méta-modèles, et les liens entre éléments de modèles.
Une correspondance peut être de type scalaire (i.e : $=$, $>$), fonction (conversion, concaténation,...) ou orientée-relation (agrégation, héritage, composition,...). Pottinger et al. [12] définissent deux catégories de métriques de correspondance entre modèles : égalité et similarité. Plusieurs types d'égalité sont référencés dans la littérature à savoir : égalité de noms, de synonymes et d'hyponymes. De même on trouve plusieurs types de similarité : Similarité de types, similarité de noms et similarité de relations. Pour représenter les différents types de liens

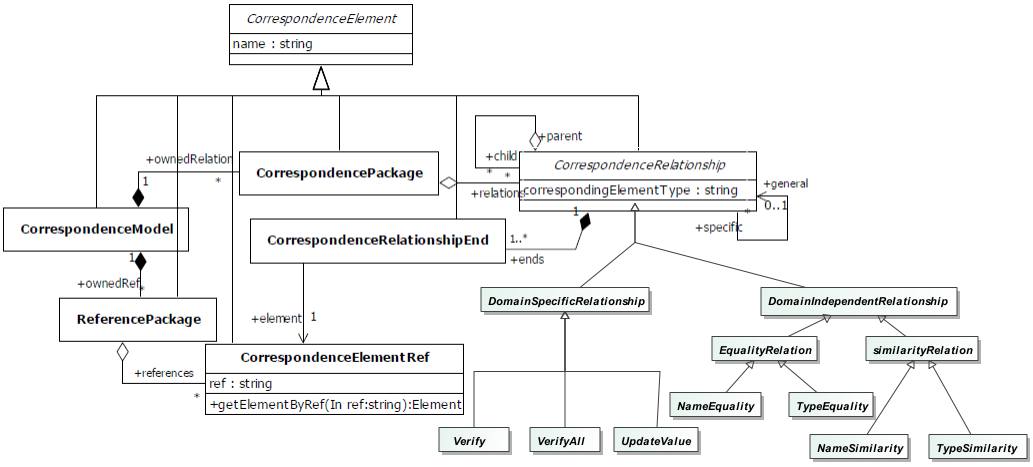

Figure 1: Méta-modèle de correspondance proposé

de correspondance, nous avons décidé d'étendre le méta-modèle proposé par Anwar et al. dans [1] afin de prendre en compte la diversité des DSL. Pour cela, nous avons notamment introduit deux concepts abstraits et des spécialisations de ceux-ci (figure 1) :

- « DomainIndependentRelationship » : cette méta-classe est la super-classe abstraite qui représente les relations génériques. L'égalité et la similarité (mentionnées ci-dessus) sont des spécialisations de ce concept ;
- « DomainSpecificRelationship » : cette méta-classe abstraite permet de représenter des liens entre modèles d'un même domaine. De nouveaux types de liens sont spécifiés par

spécialisation de ce concept comme illustré sur le méta-modèle à travers les relations Verify, VerifyAll et UpdateValue issues des DSL de notre cas d'étude.

4 Application à un cas d'étude

4.1 Problématique

Pour illustrer notre approche, nous avons choisi un cas d'étude inspiré d'un projet réel réalisant le suivi d'anomalies, appelé BTS (Bug System Tracking). Un tel système a pour objectif d'offrir aux acteurs, en fonction de leurs différents statuts (Chef d'équipe, développeurs, testeurs,...), la possibilité de signaler des disfonctionnements et de les commenter, de suivre l'état d'une anomalie, d'aviser les collaborateurs des problèmes rencontrés, de suggérer des solutions ou des possibilités de contournement,... Le choix de cet exemple nous parait judicieux parce qu'il fait intervenir différents acteurs d'un même système, travaillant avec différents points de vue et participant aux différentes étapes du cycle de vie d'un projet logiciel, depuis l'analyse des exigences utilisateurs jusqu'à l'implantation de la solution.

4.2 Modèles du BTS

Pour commencer, nous donnons un aperçu des différents modèles de points de vue de notre système. Pour avoir une approche qui soit la plus générique possible et parce que nous ne traitons pas la gestion des conflits (nommage,...), nous supposons que nos modèles utilisent le même dictionnaire de données partagés par les différents acteurs, et qu'il existe une phase préalable de résolution des conflits.
Supposons que lors de la modélisation du BTS, nous ayons trois types d'acteurs: l'analyste, l'architecte logiciel et l'ingénieur des procédés qui travaillent respectivement sur les modèles par point de vue suivants : point de vue gestion des exigences des utilisateurs, point de vue développement au niveau plate-forme et point de vue modélisation métier. Tout d'abord l'analyste est responsable de la modélisation des besoins clients sous forme de modèle d'exigences conforme au méta-modèle SysML (figure 2). Pour des raison de simplicité, nous nous sommes limités dans la description à quelques exigences du BTS. L'exigence « Déclaration d'une anomalie » par exemple, définit une sous-exigence « Résumé de l'anomalie », raffinée afin d'ajouter des contraintes supplémentaires à faire respecter par l'utilisateur, par exemple, lors de la déclaration d'une anomalie. Par la suite l'architecte logiciel est quant à lui responsable de la modélisation des anomalies avec le langage dédié Mantis décrit par [4]. Une anomalie est caractérisée par un identifiant unique, un ensemble d'informations sur la version du produit, les étapes à reproduire pour aboutir au problème soulevé (summary, description, stepsToReproduce,...) et des relations (severity, priority, assignedTo,...) permettant de mieux cerner l'anomalie. Enfin, nous supposons que l'ingénieur des procédés, responsable de la modélisation du processus métier, utilise la notation BPMN [3] pour modéliser le processus métier. Le traitement d'une anomalie peut être vu comme un processus métier, que les différents collaborateurs doivent mettre en oeuvre pour pouvoir la résoudre. Les extraits des modèles que nous proposons pour le BTS sont décrits dans la figure 2

4.3 Liens de correspondance du système BTS

Pour prendre en compte les spécificités du domaine du BTS, le méta-modèle de correspondances comporte des méta-classes qui sont des spécialisations de « DomainSpecificRelationship »:

- Verify : Ce concept modélise la relation entre une exigence et un élément de modèle. Il permet d'inspecter si l'élément de modèle vérifie l'exigence à laquelle il est lié,
- VerifyAll : Ce concept a pour objectif de s'assurer que l'ensemble des exigences dérivées et raffinées sont vérifiées, en reliant l'élément de modèle à l'exigence parente,
- UpdateValue : Ce concept permet d'examiner que la valeur de l'élément de modèle a été modifiée après l'exécution de la tâche associée.

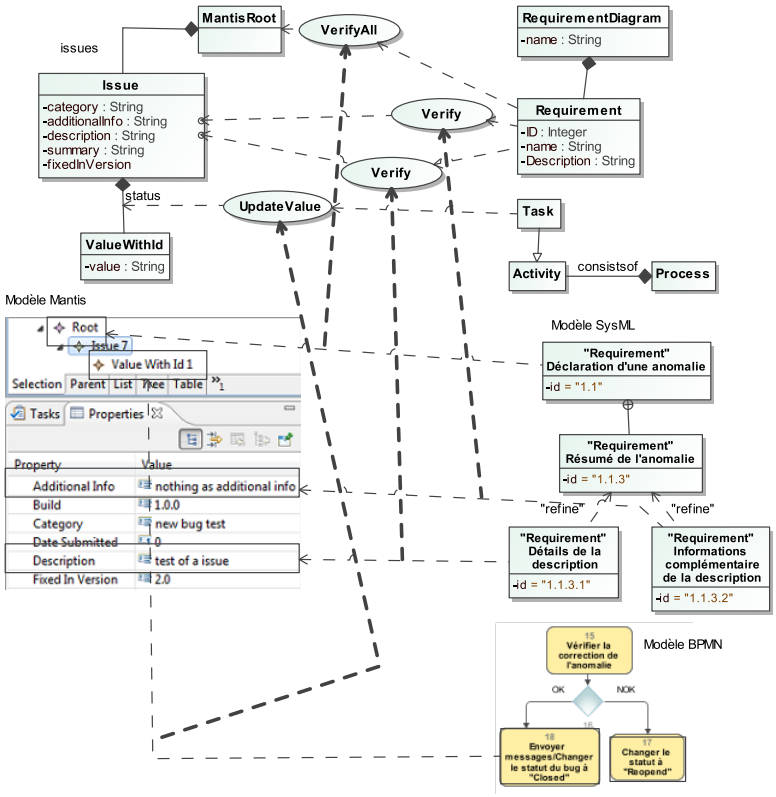

Figure 2: Un extrait des liens entre les éléments des modèles

La figure 2 illustre un ensemble de liens identifiés entre des exigences, des tâches métiers du processus BPMN et le modèle Mantis.

5 Conclusion et perspectives

Notre approche est similaire à AMW dans le sens où elle est fondée sur des modèles avec un méta-modèle générique, permettant d'identifier les correspondances, qui peut être étendu pour supporter des exigences spécifiques. Notre contribution se distingue de AMW car nous proposons un ensemble prédéfini de concepts génériques supportant les liens les plus courants entre (méta-)modèles. Nous offrons en plus la possibilité d'étendre le méta-modèle proposé afin de combler le manque de concepts et de supporter les exigences relatives à des domaines

spécifiques. Nous réalisons actuellement un mécanisme de gestion de la synchronisation entre les (méta)-modèles assurant le maintien de la cohérence entre modèles de correspondance, et modèles à grande échelle. Les suites à donner à ce travail préliminaire sont multiples. Dans un premier temps, nous allons enrichir notre méta-modèle de liens en intégrant d'autres concepts (agrégation et héritage par exemple, comme relevé dans [14, 10]). Dans un second temps, nous réaliserons un mécanisme de gestion de synchronisation, de façon semi-automatique ou assistée, entre les (méta-)modèles d'entrée et le modèle de correspondance afin de maintenir et gérer la cohérence du modèle de liens en particulier au cours de l'évolution des modèles par point de vue du système. Il est possible également d'utiliser le modèle de liens pour composer (par exemple par fusion ou virtualisation) les modèles partiels pour obtenir un modèle global unifié.

Remerciements : Cet article décrit les premiers résultats d'un travail de recherche effectué dans le cadre du projet PHC Volubilis MA/11/254.

References

[1] A. Anwar, S. Ebersold, B. Coulette, M. Nassar, and A. Kriouile. A rule-driven approach for composing viewpoint-oriented models. *Journal of Object Technology*, 9(2):89–114, 2010.

[2] E. Baniassad and S. Clarke. Theme: An approach for aspect-oriented analysis and design. In *Proceedings of the 26th International Conference on Software Engineering*, pages 158–167, 2004.

[3] OMG BPMN. Omg bpmn-v2.0. http://www.omg.org/spec/BPMN/2.0/PDF, January 2011.

[4] J. Bézivin, H. Bruneliere, F. Jouault, and I. Kurtev. Model engineering support for tool interoperability. In *Proceedings of the 4th Workshop in Software Model Engineering (WiSME 2005), Montego Bay, Jamaica*, volume 2, 2005.

[5] Jean Bézivin, Salim Bouzitouna, Marcos Didonet Del Fabro, Marie-Pierre Gervais, Fréderic Jouault, Dimitrios Kolovos, Ivan Kurtev, and Richard F. Paige. A canonical scheme for model composition. In *Model Driven Architecture–Foundations and Applications, Lecture Notes in Computer Science*, volume 4066/2006, pages 346–360, 2006.

[6] Z. Drey, C. Faucher, F. Fleurey, and V. Mahé. Kermeta language. *Reference Manual*, 2009.

[7] J. Klein, L. Hélouët, and J.M. Jézéquel. Semantic-based weaving of scenarios. In *Proceedings of the 5th international conference on Aspect-oriented software development*, pages 27–38, 2006.

[8] D. Kolovos, R. Paige, and F. Polack. Merging models with the epsilon merging language (eml). *Model Driven Engineering Languages and Systems*, pages 215–229, 2006.

[9] Dimitrios S. Kolovos. Establishing correspondences between models with the epsilon comparison language. In *Model Driven Architecture - Foundations and Applications, 5th European Conference, ECMDA-FA 2009, June 23-26, 2009. Proceedings*, volume 5562 of *Lecture Notes in Computer Science*, pages 146–157. Springer, 2009.

[10] I. Kurtev. Metamodels: definitions of structures or ontological commitments? In *Workshop on Towers of Models*, pages 53–63, York, June 2007. University of York.

[11] Del Fabro M.D., Bezivin J., Jouault F., Breton E., and Gueltas G. AMW: a generic model weaver. *Proceedings of the 1ere Journée sur l'Ingénierie Dirigée par les Modeles (IDM05)*, 3:7–11, 2005.

[12] R.A. Pottinger and P.A. Bernstein. Merging models based on given correspondences. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 862–873, 2003.

[13] OMG SysML. Omg sysml-v1.1. http://www.sysml.org/docs/specs/OMGSysML-v1.1-08-11-01.pdf, November 2008.

[14] Konrad Voigt, Petko Ivanov, and Andreas Rummler. Matchbox: combined meta-model matching for semi-automatic mapping generation. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 2281–2288. ACM, 2010.