Assistance pour l'intégration de méthodes fondée sur les graphes conceptuels

HAJ AYECH Fayçal - KHALFALLAH Adel - BEN AHMED Samir

Département d'informatique
Faculté des sciences de Tunis
Campus universitaire, Tunis-1060.
TUNISIE.
F_ayech@yaboo.fr

Institut Supérieur d'Informatique
2, Rue Abou Raihan El Bayrouni
Ariana-2080.
TUNISIE
Adel.Khalfallah@fst.rnu.tn - Samir.Benahmed@fst.rnu.tn

RÉSUMÉ. Les méthodes de développement actuelles ne permettent pas aux usagers d'être guidés efficacement dans leur travail, de partager et de réutiliser leurs expériences. Un environnement pour construire de nouvelles méthodes en réutilisant et en assemblant des composants de méthodes déjà existantes, sera très utile. Nous donnons le nom d'intégration à ce processus d'assemblage. Ce document propose une approche pour l'intégration de méthodes. Elle est basée sur le formalisme de graphes conceptuels de Sowa. A travers cette approche, nous définissons une démarche qui permet de systématiser le processus d'intégration de méthodes ainsi que leurs instances.

ABSTRACT. Actually, development methods do not allow users to be guided efficiency in their work, to share and reuse their experiences. An environment for building methods by reusing and assembling components of existing methods would be helpful. We give the name of integration for the assembling process. In this document, we propose an approach for integrating methods. It is based on Sowa's Conceptual Graph Theory. Through this approach, we define an approach for automating the integration of methods and their instances.

MOTS-CLÉS : méthode, intégration, graphe conceptuel, instance de méthode.

KEYWORDS: method, integration, conceptual graph, instance of method.

1. Problématique

Avec la croissance de la complexité des domaines d'application, la construction de nouvelles méthodes devient de plus en plus difficile. Une méthode qui a fait ses preuves dans un domaine d'application peut être inadéquate dans d'autres domaines. Il existe un nombre important de méthodes disponibles pour l'ingénierie des systèmes. Mais de nombreuses enquêtes [15], [16], [12] ont montré que ces méthodes ne sont pas bien adaptées aux besoins de leurs utilisateurs [6].

Plusieurs approches ont été suggérées pour faire face aux limitations de méthodes. Une de ces approches, qui est d'un intérêt récent : l'intégration de méthode. *L'intégration de méthode est le processus permettant de combiner deux méthodes ou plus, pour construire une nouvelle méthode qui est plus utile que les méthodes initiales.*

Trois problèmes de base devront être résolus : Fournir un moyen uniforme pour introduire et représenter une méthode, proposer un stockage de méthode dans une base de méthode et fournir un mécanisme d'intégration qui permet d'assembler deux méthodes pour établir une nouvelle méthode.

[3], propose une démarche pour la comparaison et l'intégration de graphes conceptuels dans le cadre de l'acquisition des connaissances à partir de multiples experts. Nous adapterons cette démarche pour développer un mécanisme d'intégration automatique permettant d'intégrer des graphes conceptuels correspondant à des méthodes.

Nous présenterons dans la première section un aperçu sur les graphes conceptuels ; nous précisons dans la deuxième section l'algorithme d'intégration de graphes conceptuels et qui est proposé par [3] ; dans la troisième section nous proposerons l'architecture de notre solution en terme de démarche d'intégration et stockage de méthode. La quatrième section sera consacrée à expliciter notre méthode d'intégration et qui se base sur deux stratégies : la spécialisation et la généralisation. Nous présenterons dans la cinquième section quelques exemples illustrant l'application de notre démarche d'intégration.

2. Graphes conceptuels.

Le formalisme des Graphes Conceptuels [13] est dérivé à la fois des modèles de représentation de connaissances généraux de type "réseau sémantique" [14] et des Graphes Existentiels de C.S. Peirce [10]. Les GCSs sont constitués de deux types de sommets: les sommets concepts (aussi appelés *concepts*) et les sommets relations (aussi appelés *relations conceptuelles*). Chaque sommet (ou nœud) est muni d'une étiquette. L'étiquette d'une relation conceptuelle est un type de relation, tandis que celle d'un

concept est constituée d'un type de concept et d'un référent. Les types et les référents utilisés dans les GCSs doivent avoir été préalablement déclarés ou définis dans un "*support*" afin d'ordonner et/ou de poser des contraintes sur ces types et ces référents. Nous assimilons un support à une ontologie. Les types des concepts sont ordonnés en une structure de treillis fini par une relation d'ordre partielle représentant la relation sémantique "sorte-de". Cette relation est une relation de *spécialisation/généralisation* entre types de concepts.

3. Démarche d'intégration

Soient CG1 et CG2 deux graphes conceptuels à intégrer et qui sont relatifs aux supports S1 et S2. Trois phases constituent la démarche d'intégration [3].

1ère phase : Construction d'un support commun.

Comparer le treillis de type de concept, ainsi que le treillis de type de relation utilisés par les deux experts. Résoudre les conflits de nom (reconnaître les synonymes et les homonymes à travers les noms de type).

2ème phase : Comparaison des graphes conceptuels CG1 et CG2 .

– Etablir les relations entre les liens élémentaires des deux graphes :

– Etablir les relations entre CG1 et CG2 .

3ème phase : Construire le graphe intégré en respectant la stratégie d'intégration choisie. Il y a six famille de stratégie d'intégration : généralisation, spécialisation, conceptualisation, instanciation, la compétence, le consensus entre experts.

4. Solution

Notre objectif est de fournir un système permettant aux utilisateurs d'introduire des représentations de méthode en terme de graphes conceptuels, d'instancier ces graphes, et de proposer des mécanismes d'intégration automatique aidant à définir de nouvelles méthodes à partir de celles existantes. Trois étapes régissent notre solution (Figure 1) :

– Modéliser les méthodes selon un modèle commun : les graphes conceptuels.

– Proposer une architecture de base de données, permettant de stocker l'ensemble des graphes conceptuels.

– Utiliser l'algorithme d'intégration défini ci-dessus, pour proposer de nouvelles méthodes à partir de méthodes existantes.

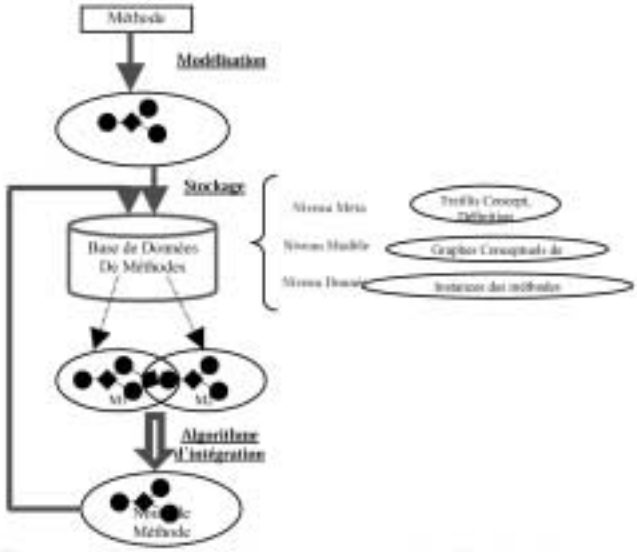

Figure 1. *Architecture de notre approche d'intégration*

L'étape de modélisation est un problème qui n'est pas encore résolu. Le processus d'intégration est appliqué au niveau des méthodes. L'utilisateur choisit la stratégie d'intégration qu'il veut adopter pour intégrer deux méthodes. En respectant cette stratégie, le processus génère une nouvelle méthode, mais il génère aussi les instances de cette nouvelle méthode à partir des instances des méthodes initiales.

5. Méthode d'intégration

La méthode d'intégration se base sur l'approche de DIENG [3]. Elle est guidée par une stratégie. Dieng propose un certain nombre de stratégies qui se basent sur la notion de lien élémentaire (la plus petite information significative dans un graphe conceptuel) ; il est constitué d'une relation conceptuelle et de l'ensemble des concepts qui lui sont connectés. Dieng définit une liste de relations possibles entre liens élémentaires. A partir de cette liste nous définissons une relation d'ordre partiel « < » entre l'ensemble des liens élémentaires des graphes conceptuels des deux méthodes à intégrer.

5.1. Opération sur les liens élémentaires

Nous définissons à partir de la liste de relations entre liens élémentaires quatre opérations : Plus-General(L1,L2) ; Ancetre-Commun(L1,L2) ; Plus-Specifique(L1,L2) ; Fils-Commun(L1,L2) .

5.2. Algorithme d'intégration

L'ingénieur de méthode se trouve devant deux méthodes modélisées sous forme de graphe conceptuel CG1 et CG2. Il décide de les intégrer selon une des stratégies : Généralisation ou spécialisation.

(CGi)i=1,2 : deux graphes conceptuels représentant deux méthodes à intégrer,

CGintegrer : Graphe résultat d'intégration.

– Choisir la stratégie : Généralisation (Resp. Specialisation)

– L1 ∈ CG1, L2 ∈ CG2 | Plus-General(L1,L2)≠ϕ (Resp. Plus-Specifique(L1,L2)≠ϕ)

– L = Plus-General(L1,L2) ; Inst(L) ← Inst(L1) ∪ Inst(L2)

– CG1 ← CG1 / {L1} , CG2 ← CG2 / {L2} .

– TantQue ∃ Link ∈ (CGi)i=1,2 | Plus-General(L,Link) ≠ ϕ , Faire

(Resp. TantQue ∃ Link ∈ (CGi)i=1,2 | Plus-Specific(L,Link) ≠ ϕ , Faire)

- L = Plus-General(L,Link) (Resp. L = Plus-Specific(L,Link))

- Inst(L) ← Inst(L) ∪ Inst(Link) (Resp. Inst(L) ← Inst(L) ∩ Inst(Link))

- Supprimer Link.

Fin TantQue

– CGintegrer ← CGintegrer ∪ {L}

– Répéter , jusqu'à : ∀ L1 ∈ CG1, L2 ∈ CG2 , Plus-General(L1,L2) = ϕ,

(Resp. , Plus-Specifique(L1,L2) = ϕ).

– CGintegrer ← CGintegrer ∪ CG1 ∪ CG2

– TantQue ∃ L=Ancetre-Commun(L1,L2) ≠ ϕ | L1 ∈ $CG_{integrer}$,L2 ∈ $CG_{integrer}$,Faire

(Resp. TantQue ∃ L=Fils-Commun(L_1,L_2) ≠ ϕ | L_1 ∈ $CG_{integrer}$,L_2 ∈ $CG_{integrer}$,Faire)

- CGintegrer ← CGintegrer / {L1,L2}

- Inst(L) ← Inst(L1) ∪ Inst(L2), (Resp. Inst(L) ← Inst(L1) ∩ Inst(L2))

- CGintegrer ← CGintegrer ∪ {L}.

Fin TantQue

CGintegrer est le graphe d'intégration de CG1 et CG2.

6. Exemple

Nous présenterons dans cette section des exemples d'intégration de la méthode AMDEC [8] (Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité) avec une méthode se basant sur les notations de UML. Pour illustrer notre algorithme

d'intégration, nous intégrons des parties spécifiques des deux méthodes. Nous proposons les représentations suivantes pour UML (Figure 2) et AMDEC (Figure 3) :

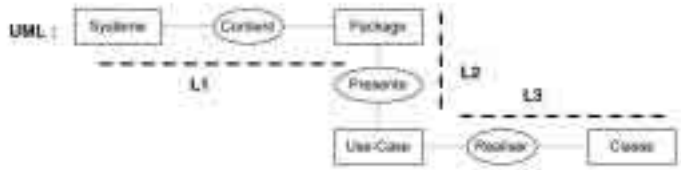

Figure 2. *Graphe conceptuel d'une méthode basée sur UML*

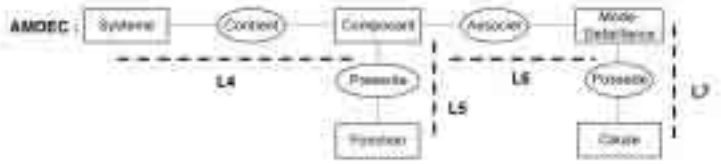

Figure 3. *Graphe conceptuel de la méthode AMDEC*

1ère Alternative : intégration des deux méthodes en les combinant.

Nous proposons le treillis des concepts unifiés suivant (Figure 4). L'ensemble de relation entre liens élémentaires est : { L4 < L1 ; L5 < L2 }.

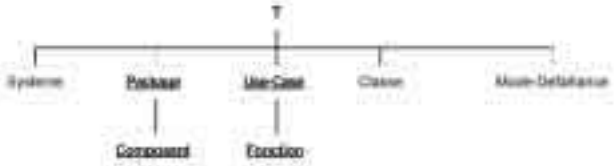

Figure 4. *Treillis des concepts unifié*

L'ingénieur d'application décide d'utiliser la stratégie de spécialisation pour créer une nouvelle méthode (Figure 5).

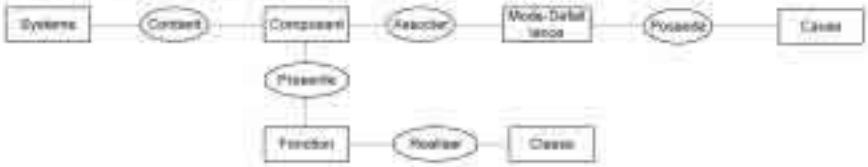

Figure 5. *Méthode intégrée selon la stratégie de spécialisation*

Toutes les instances du concept « composant » doivent figurer dans l'ensemble des instances du concept « package » ; il est de même pour les instances des concepts « fonction » et « use-case ». Dans le cas contraire, une erreur est signalée, qui peut être due à la non correspondance entre concepts des deux méthodes ou à une mauvaise conception du projet.

2ème Alternative : l'ingénieur utilise AMDEC ; à une étape donnée du processus de développement, il décide d'intégrer un composant UML : Uses cases et Diagramme de classes. Nous modifions légèrement le treillis des concepts (Figure 6). L'ensemble de relation entre lien élémentaire est : { L1 < L4 ; L2 < L5 }.

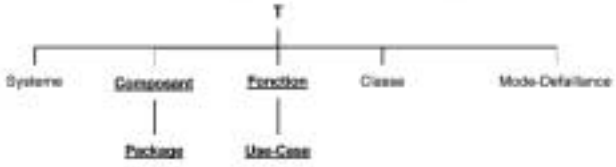

Figure 6. *Treillis des concepts unifié*

L'ingénieur d'application décide d'utiliser la stratégie de généralisation pour créer une nouvelle méthode (Figure 7).

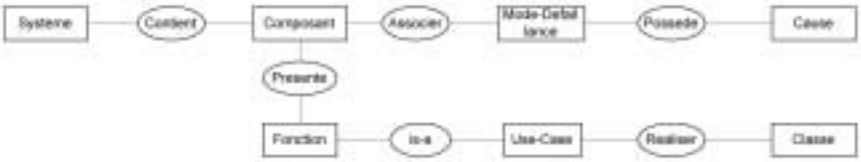

Figure 7. *Méthode intégrée selon la stratégie de généralisation*

Les instances du concept « use-case » sont établies automatiquement, elles correspondent à l'ensemble ou partie des instances du concept « fonction ».

7. Conclusion

Nous avons adapté l'algorithme de [3] pour intégrer des graphes conceptuels de méthodes ainsi que leurs instances. Nous espérons élaborer des exemples complets pour l'intégration de méthodes (graphes de méthodes et instances) afin de confirmer l'importance de notre solution. Une des orientations possibles pour notre solution étant la mémoire projet où elle peut être un support d'assistance pour le développement coopératif.

13. Bibliographie

[1] Booch G. *Object Oriented Analysis and Design with Applications*, Benjamin/Cummings, 1991.

[2] Chein M. & Mugnier M.L. *Conceptual graphs, fundamental notions*. In Revue d'intelligence Artificielle, 6(4):365-406, 1992.

[3] Dieng R. Comparison of conceptual Graphs for Modelling knowledge of Multiple Experts : Application to Traffic Accident Analysis. Rapport de Recherche, INRIA. Avril 1997.

[4] Ellis G. & Lehmann F. *Exploiting the Induced Order on Type-labeled Graphs for Fast Knowledge Retrieval*. In ICCS.1994.

[5] Hidding G.J. *Methodology information : who uses it and why not?*, Proc. WITS-94, Vancouver, Canada, 1994.

[6] Lyytinen K. *Different perspectives on information systems : problems and solutions*, ACM Computing Surveys, Vol 19, No1, 1987.

[7] Mugnier M.L. & Chein M. Représenter des connaissances et raisonner avec des graphes. In RIA, Revue d'intelligence Artificielle, Numéro spécial Graphes Conceptuels, Vol. 10, 1996.

[8] Orsoni B. Diagnostic prédictif sur robinet papillon. SAICA : Société Aquitaine d'Ingénierie et de Conseil en Automatique.

[9] Ralyté J. Ingénierie des méthodes à base de composants. Thèse de doctorat en informatique, Université Paris 1 – Sorbonne, Janvier 2001.

[10] Roberts D.D. *The existential Graphs of Charles S. Peirce*. Mouton, The Hague, 1973.

[11] Rolland C., Souveyet C., Moreno M. *An Approach for Defining Ways-Of-Working*, in the Information Systems Journal, 1995.

[12] Russo et al, *The use and adaptation of system development methodologies*, Proc. 1995 Intl. Resources Management. Association Conference, Atlanta, 1995.

[13] Sowa J.F. *Conceptual Structures - Information Processing in Mind and Machine*, Addison-Wesley, 1984.

[14] Sowa J.F - Editor (1991). *Principles of Semantic Networks: Explorations in the Representation of Knowledge*. (Ed.: Sowa J.F.), Morgan Kaufmann, 1991.

[15] Wijers, VanDort H.E. *Experiences with the use of CASE tools in the Netherlands*, Advanced Information Systems Engineering, pp 5-20, 1990.

[16] Yourdon E. *The decline and fall of the american programmer*, Prentice Hall, Englewood Cliffs, NJ, 1992.