Intégration des (multi-)exigences tout au long du développement des systèmes complexes

Florian Galinier, Jean-Michel Bruel, Sophie Ebersold, Bertrand Meyer

Intégration des (multi-)exigences tout au long du développement des systèmes complexes

Florian Galinier, Jean-Michel Bruel, Sophie Ebersold, Bertrand Meyer*

IRIT, Université de Toulouse, Toulouse, France
**Également Software Engineering Lab, Innopolis University, Russie et Politecnico di Milano, Italie*

1 Introduction

La création de systèmes complexes implique de nombreux acteurs, provenant de domaines différents. En raison de cette hétérogénéité, la spécification de ces systèmes est réalisée à l'aide d'outils différents, comme des documents textuels, des bases de données d'exigences, des diagrammes SysML (*Systems Modeling Language*)... Un des défis de l'IS (Ingénierie Système) est d'établir la cohérence et les liens entre ces différents artefacts dans l'objectif d'assurer la qualité du produit final.

En effet, il existe à ce jour un manque de cohérence entre les différentes vues de ces systèmes qui rend plus difficile l'analyse des exigences et la détection des conflits. Une approche multi-vues, avec un unique langage ou avec une abstraction commune des artefacts de spécification devrait ainsi permettre de détecter de telles incohérences en amont et faciliter les communications des parties prenantes. C'est une dimension que nous appellerons *horizontale* des multi-exigences.

Dans [1], Bertrand Meyer propose d'entremêler dans un langage de programmation la spécification et l'implémentation, afin de réduire l'écart entre les exigences et la réalisation concrète du système, dans une optique sans rupture. L'application de son concept de *multirequirements* à l'IS devrait ainsi permettre de faire le lien entre les différents niveaux d'abstraction de représentation du système et d'en faciliter la mesure d'impact du changement. C'est une dimension que nous appellerons *verticale* des multi-exigences. De plus, l'utilisation d'un langage commun pour la spécification et la conception devrait permettre la réintroduction dans le système des exigences déduites de l'implémentation.

L'objectif du travail à réaliser durant cette thèse est de définir des méthodes et outils pour permettre cette intégration sans rupture des exigences dans les deux dimensions vues précédemment, et ainsi contribuer à un problème important en ingénierie des exigences (IE) : le problème de la traçabilité entre les exigences et les artefacts développés pour y répondre, qui rend si difficile la mesure de l'impact des changements.

2 Expression des exigences dans différentes vues

L'INCOSE (*International Council on Systems Engineering*) a mis en avant le besoin de concilier les points de vues des différentes parties prenantes [2]. Notre démarche s'inscrit dans cet objectif. En effet, plutôt que d'imposer l'utilisation d'un unique langage pour exprimer les exigences, l'utilisation d'une interface commune permettrait de lier différents formalismes tout en permettant aux ingénieurs de continuer à utiliser leurs outils.

Outils en langue naturelle Il existe aujourd'hui de nombreux outils de gestion des exigences [3]. Parmi les plus connus, *IBM Rational DOORS* ou encore *Reqtify* de Dassault Systems fournissent des outils pour gérer les exigences. Ces solutions laissent les utilisateurs représenter de diverses façons les exigences et permettent de faire le lien entre elles. Ces outils permettent d'établir une traçabilité aussi bien entre les exigences et leurs réalisations, qu'entre les exigences elles-mêmes mais ils ne proposent pas de réelle sémantique pour ces liens.

Les approches GORE (*Goal-Oriented Requirements Engineering*) telles que KAOS [4] permettent également d'exprimer les exigences en langue naturelle ainsi que les relations existantes entre ces exigences. La sémantique sur les liens existants (raffinement d'une exigence, exigence composée d'autres exigences, etc.) est ici définie. Par contre, les exigences sont exprimées en langage naturel, sans syntaxe imposée, ce qui rend la déduction de liens difficile, ces derniers devant être indiqués par l'utilisateur.

Approche dirigée par les modèles L'initiative GEMOC [5] a pour objectif de définir une interface commune pour différents DSML (*Domain Specific Modeling Language*) utilisés pour exprimer les besoins spécifiques des différents acteurs impliqués dans un projet. L'utilisation des modèles comme artefacts de base est ainsi proposée, afin de combler l'écart entre différents DSML, de façon similaire à l'utilisation des artefacts comme passerelle entre spécification et implémentation proposée en ingénierie des modèles. De plus, l'acceptation de cette approche pourrait être facilitée par la croissance de l'intérêt pour l'approche MBSE (*Model-Based System Engineering*) par les industriels en IS, qui peut être utilisée afin d'exprimer les exigences comme des éléments de modèles, de la même manière que les autres artefacts de modélisation. Cette approche devrait ainsi permettre de créer des liens entre les exigences et les autres artefacts, que ce soit d'autres exigences ou des éléments de spécification fournis par les différentes parties prenantes. Ainsi, il serait possible de combiner des éléments de différents domaines dans une vue holistique, prenant en compte les liens entre les artefacts spécifiques à un domaine mais également entre ces artefacts et les exigences. Dans [6] par exemple, les auteurs proposent d'appliquer cette fédération de modèles aux exigences, considérant que chaque espace technologique est une technique d'expression des exigences. Cela permet par exemple de lier des exigences exprimées en langue

naturelle dans un document type Word avec des exigences et leurs liens exprimés grâce à l'approche KAOS.

3 Formalisation des exigences

Le standard ISO/IEC/IEEE 29148:2011 [7] définit un certain nombre de qualités nécessaires pour l'expression des exigences (la traçabilité, la vérifiabilité, la consistance et l'absence d'ambiguïtés, ...). L'utilisation d'un langage dédié aux exigences, plus formel que la langue naturelle habituellement utilisée, est un moyen possible d'assurer ces différentes qualités.

Expression des exigences L'utilisation d'un langage dédié a été étudiée à différentes occasions. Le profil SysML [8] propose ainsi un diagramme spécifique à l'expression des exigences (à la fois fonctionnelles et non-fonctionnelles), ainsi que les liens existant entre exigences ou entre les exigences et les autres éléments de modèles (blocs, cas d'utilisation). Les exigences en elles-mêmes y sont cependant représentées sous une forme textuelle (voir Fig. 1).

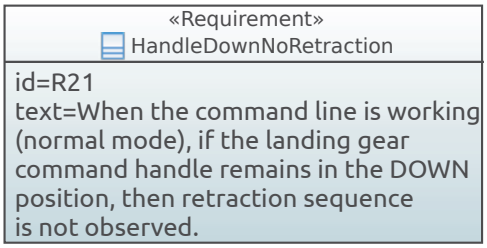

Figure 1: Représentation de l'exigence R21 de [9] en SysML

Cependant dans les approches proposées jusqu'à présent, les exigences sont toujours exprimées en langage naturel et sont, par conséquent ambiguës. Des travaux ont été proposés pour surmonter ce problème (e.g., [10], [11]). Si ces approches permettent de faire le pont entre une formalisation des exigences et une expression des exigences compréhensible par des non-informaticiens, l'utilisation de deux langages rend nécessaire de maintenir la cohérence entre les exigences ainsi exprimées, et la répercussion des changements n'est pas aussi immédiate que dans le cas de l'utilisation d'un formalisme unique.

Langages spécifiques d'expression des exigences Afin d'éviter cet écart entre langue naturelle et approches plus formelles, certains travaux mettent en avant des langages dédiés à l'expression des exigences (e.g., [12], [13]).

Cependant, à chaque fois, la syntaxe proposée n'est pas suffisamment simple ou proche des utilisateurs pour être adoptée facilement. De plus, ils s'adressent à des domaines particuliers et sont par conséquents très spécifiques.

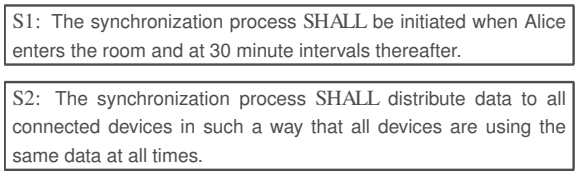

Figure 2: Exemple d'exigences en RELAX extrait de [12].

Expression formelle des exigences De nombreux travaux proposent d'exprimer les exigences avec des méthodes formelles (e.g., [14], [15], [16]). Il est à noter que ces approches ne sont pas du tout destinées à des non-spécialistes et ne peuvent ainsi pas être utilisées comme interface de discussion entre acteurs de différents domaines.

Dans [1], Bertrand Meyer propose une approche pour exprimer les exigences directement dans le code source Eiffel. Il applique ainsi le *Single Model Principle* proposé dans [17]. L'exigence proposée en Fig. 1 peut ainsi être exprimée sous forme de préconditions et postconditions d'une opération exécutant la boucle d'exécution du système (donnée List. 1).

```
r21
    -- (R21) When the command line is working (normal
    -- mode), if the landing gear command handle
    -- remains in the DOWN position, then retraction
    -- sequence is not observed.
    require
      handle_status = is_handle_down
    do
      main
    ensure
      gear_status /= is_gear_retracting
end
```

Listing 1: Exemple de représentation en Eiffel de l'exigence R21 présentée Fig. 1.

L'expression des exigences à l'aide des contrats permet ainsi de fournir une interface plus accessible que les méthodes beaucoup plus formelles, tout en bénéficiant des outils formels d'Eiffel – comme AutoProof [18] un vérificateur pour Eiffel – afin de détecter les éventuelles incohérences du système. Il propose également de lier au sein d'un même formalisme les exigences, exprimées en langue naturelle, avec leur spécification (au travers de diagramme) et leur implémentation. Cette approche permet ainsi de faciliter la traçabilité.

4 Travaux futurs

L'intégration des exigences tout au long du développement d'un système complexe est un moyen de réduire les coûts engendrés par les erreurs dues à une mauvaise analyse des exigences. Les méthodes présentées ont toutes pour objectif l'ajout d'un certain formalisme afin d'améliorer le traitement des exigences. L'utilisation de la langue naturelle comme principal outil d'expression des exigences nous amène cependant à nous poser un certain nombre de questions :

- Comment réussir à exprimer les exigences de façon à ce qu'elles restent compréhensibles par un non-spécialiste tout en étant analysables de façon automatique ?
- Comment faire le lien entre les exigences exprimées par différentes parties prenantes ?
- Quelle sémantique donner aux liens existant entre les exigences ? Entre les exigences et le système ?
- Comment utiliser la formalisation des exigences pour prouver des propriétés des exigences ?

Si les travaux cités jusqu'à présent permettent de donner des pistes de réponses, il n'en existe, à notre connaissance, aucun qui permette de répondre à toutes ces questions à la fois. Un des objectifs principaux de notre travail est ainsi de définir une formalisation des exigences, via un langage qui permettra de faire le lien entre les exigences de différents domaines. Ce formalisme devra également être utilisé afin de faire le lien entre les exigences et les autres artefacts de spécification. Nous souhaitons également fournir des outils afin d'assister les ingénieurs des exigences dans le contrôle de la qualité et de la validité du système (à l'aide de traces, de couverture, ...).

Un autre objectif important de notre projet est la simplicité d'utilisation et d'appropriation de cette interface. En effet, afin de permettre à des ingénieurs non-informaticiens d'utiliser ces outils, nous souhaitons qu'ils soient aussi proches que possible de leurs outils habituels.

Afin d'expérimenter notre approche, nous utilisons l'étude de cas du système de train d'atterrissage proposée dans [9], qui définit un système et ses exigences de façon détaillée. Cet exemple, proposé durant la conférence ABZ2014 [19], est accompagné d'un ensemble d'articles proposant des formalisations des exigences identifiées, avec lesquelles nous pourrons comparer notre approche. Par la suite, nous prévoyons de valider notre approche sur un exemple industriel réel.

References

[1] B. Meyer. Multirequirements. *Modelling and Quality in Requirements Engineering (Martin Glinz Festscrhift)*, 2013.

[2] INCOSE. *SE Vision 2025*. 2014. http://www.incose.org/docs/default-source/aboutse/se-vision-2025.pdf.

[3] J. M. Carrillo de Gea, J. Nicolás, J. L. F. Alemán, A. Toval, C. Ebert, and A. Vizcaíno. Requirements Engineering Tools. *IEEE Software*, 28(4):86–91, July 2011.

[4] A. van Lamsweerde. Goal-oriented requirements engineering: a guided tour. In *Proceedings Fifth IEEE International Symposium on Requirements Engineering*, pages 249–262, 2001.

[5] B. Combemale, J. Deantoni, B. Baudry, R. B. France, J.-M. Jézéquel, and J. Gray. Globalizing Modeling Languages. *Computer*, pages 10–13, June 2014.

[6] Fahad R. Golra, Antoine Beugnard, Fabien Dagnat, Sylvain Guerin, and Christophe Guychard. Continuous Requirements Engineering using Model Federation. *RE:Next! Track at 24th IEEE International Requirements Engineering Conference 2016*, 2016.

[7] ISO/IEC/IEEE International Standard 29148:2011. *ISO/IEC/IEEE 29148:2011(E)*, pages 1–94, December 2011.

[8] Object Management Group (OMG). OMG Systems Modeling Language (OMG SysML^{TM}), V1.0, 2007. OMG Document Number: formal/2007-09-01 Standard document URL: http://www.omg.org/spec/SysML/1.0/PDF.

[9] F. Boniol and V. Wiels. The Landing Gear System Case Study. In Frédéric Boniol, Virginie Wiels, Yamine Ait Ameur, and Klaus-Dieter Schewe, editors, *ABZ 2014: The Landing Gear Case Study*, number 433 in Communications in Computer and Information Science, pages 1–18. Springer International Publishing, June 2014.

[10] W. Scott and S. C. Cook. *A Context-free Requirements Grammar to Facilitate Automatic Assessment*. PhD thesis, UniSA, 2004.

[11] R. Hähnle, K. Johannisson, and A. Ranta. An Authoring Tool for Informal and Formal Requirements Specifications. In Ralf-Detlef Kutsche and Herbert Weber, editors, *Fundamental Approaches to Software Engineering*, number 2306 in Lecture Notes in Computer Science, pages 233–248. Springer Berlin Heidelberg, April 2002.

[12] J. Whittle, P. Sawyer, N. Bencomo, B. H. C. Cheng, and J. M. Bruel. RELAX: Incorporating Uncertainty into the Specification of Self-Adaptive Systems. In *2009 17th IEEE International Requirements Engineering Conference*, pages 79–88, August 2009.

[13] T. Nguyen. Verification of Behavioural Requirements for Complex Systems with FORM-L, a MODELICA Extension. In *26th International Conference on Software & Systems Engineering and their Applications*, EDF R&D, 6 quai Watier, 78110 Chatou, FRANCE, 2015.

[14] F.-L. Li, J. Horkoff, A. Borgida, G. Guizzardi, L. Liu, and J. Mylopoulos. From Stakeholder Requirements to Formal Specifications Through Refinement. In Samuel A. Fricker and Kurt Schneider, editors, *Requirements Engineering: Foundation for Software Quality*, Lecture Notes in Computer Science, pages 164–180. Springer International Publishing, March 2015.

[15] A. Mammar and R. Laleau. On the Use of Domain and System Knowledge Modeling in Goal-Based Event-B Specifications. In Tiziana Margaria and Bernhard Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques*, number 9952 in Lecture Notes in Computer Science, pages 325–339. Springer International Publishing, October 2016.

[16] A. Matoussi, F. Gervais, and R. Laleau. An Event-B formalization of KAOS goal refinement patterns. Technical Report Tech. Rep. TRLACL-2010-1, LACL, University of Paris-Est, 2010.

[17] R. Paige and J. Ostroff. The Single Model Principle. In *Proceedings of the Fifth IEEE International Symposium on Requirements Engineering*, RE '01, pages 292–, Washington, DC, USA, 2001. IEEE Computer Society.

[18] J. Tschannen, C. A. Furia, M. Nordio, and N. Polikarpova. AutoProof: Auto-Active Functional Verification of Object-Oriented Programs. In Christel Baier and Cesare Tinelli, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, number 9035 in Lecture Notes in Computer Science, pages 566–580. Springer Berlin Heidelberg, April 2015.

[19] F. Boniol, V. Wiels, Y. Ait Ameur, K.-D. Schewe, S. D. Junqueira Barbosa, P. Chen, A. Cuzzocrea, X. Du, J. Filipe, O. Kara, I. Kotenko, K. M. Sivalingam, D. Slezak, T. Washio, and X. Yang, editors. *ABZ 2014: The Landing Gear Case Study*, volume 433 of *Communications in Computer and Information Science*. Springer International Publishing, Cham, 2014.