Semantic Heterogeneity in the Formal Development of Complex Systems: An Introduction

J. Paul Gibson[1(✉)], Idir Aït-Sadoune[2], and Marc Pantel[3]

[1] SAMOVAR, Télécom Sud Paris, CNRS, Université Paris Saclay,
9 rue Charles Fourier, 91011 Evry Cedex, Paris, France
paul.gibson@telecom-sudparis.eu
[2] LRI - CentraleSupelec - Université Paris Saclay, Gif sur Yvette, France
idir.aitsadoune@centralesupelec.fr
[3] Institut de Recherche en Informatique de Toulouse, Toulouse, France
marc.pantel@enseeiht.fr

Abstract. Nowadays, the formal development of complex systems (including hardware and/or software) implies the writing, synthesis and analysis of many kind of models on which properties are expressed and then formally verified. These models first provide separation of concerns, but also the appropriate level of abstraction to ease the formal verification. However, the building of such heterogeneous models can introduce gaps and information loss between the various models as elements that are explicit in the whole integrated models are only explicit in some concerns and implicit in others. The whole correct development should thus only be conducted on the whole integrated model whereas separate development is mandatory for scalability of system development. More precisely, parts of these systems can be defined within contexts, imported and/or instantiated. Such contexts usually represent the implicit elements and associated semantics for these systems. Several relevant properties are defined on these implicit parts according to the formal technique being used. When considering these properties in their context with the associated explicit semantics, these properties may be not provable or even can be satisfiable in the limited explicit semantics whereas they would be unsatisfiable in the whole semantics including the implicit part. Therefore, the development activities need to be revisited in order to facilitate handling of both the explicit and implicit semantics.

Keywords: Verification · Contexts · Domains · Implicit · Explicit

The semantic heterogeneity in the formal development of complex systems has many different, yet related forms, e.g.:

– Abstraction — as our models move from the abstract to the concrete — from the *what* to the *how* — there is usually a need for a mix of non-operational and operational semantics [1].

This work was supported by grant ANR-13-INSE-0001 (The IMPEX Project http://impex.gforge.inria.fr) from the Agence Nationale de la Recherche (ANR).

T. Margaria and B. Steffen (Eds.): ISoLA 2016, Part I, LNCS 9952, pp. 321–324, 2016.
DOI: 10.1007/978-3-319-47166-2_22

- Composition — systems are composed with other systems, and such compositions cannot reasonably be expected to be done within a single homogeneous semantic framework. As the number of possible ways in which we may wish to compose different types of systems is increasing, so too increases the importance of being able to model and manage the heterogeneity [2].
- Separation of concerns — complex systems involve many different aspects (data, behavior, safety, performance, security, etc.) which are usually handled in a separated timely manner to provide scalability according to their size and complexity. This leads to heterogeneous models where some parts are explicit and other implicit according the related concern. This can be related to composition where each model does not correspond to a part of the system but to a concern in the system development.
- Reasoning — the language in which one models a system is not usually the same language in which one reasons about the relationship between models, and the correctness of one model with respect to another [3].
- Implicit versus explicit — in every model, the semantics of the language used to establish the meaning of the model are implicit. In order to understand the meaning of any model one must implicitly understand the semantics of the language in which it is expressed. Similarly, in order to validate the model one needs to establish a relationship between the model and the implicit semantics of the real-world domain in which the model has relevance. Each of these 2 implicit semantics must be made explicit and combined in order to achieve a coherent integration. However, each of these implicit semantics has a very different nature. This type of heterogeneity is a major challenge [4].

A previous thematic track — addressing the same issues — introduced some of the first research results concerned with techniques that can be used to manage the heterogeneous nature of formal modeling [5]. In *Modeling and Verifying an Evolving Distributed Control System Using an Event-based Approach* [6] the techniques were concerned with component-based system engineering where it is necessary to be able to compose components whose behavior was expressed using different modeling languages. In *Requirements driven Data Warehouse Design: We can go further* [7] the techniques were based on the use of ontological reasoning mechanisms can used to automatically construct a set of requirements that are coherent and non-conflictual, even when expressed in a variety of modeling languages. Finally, the paper *On Implicit and Explicit Semantics: Integration issues in proof-based development of systems* [8], the techniques were founded on the principle that re-usable domain knowledge should be modelled explicitly using formal ontologies.

In this year's thematic track, we emphasis the heterogeneous nature of complex systems engineering and the need for automated tool support for supporting the techniques that manage the heterogeneity in a formal way. We note that the accepted papers are concerned with theoretical advances, together with pragmatic application of these advances in real-world industrial case studies.

In *On the Use of Domain and System Knowledge Modeling in Goal-Based Event-B Specifications* [9], we see an example of semantic heterogeneity due

to the application of several different formalisms in a single system development. The paper combines Goal Oriented Requirement Engineering [10] and the refinement-based Event-B formal method [11] in order to improve the handling of requirements in a formal development. The authors rely on an ontology in order to model part of the requirements usually expressed in natural language [12]. Then the content of the ontology is translated to Event B contexts. The proposal is illustrated with the Landing Gear case study proposed in the ABZ 2014 conference [13].

In *Strengthening MDE and Formal Design Models by references to Domain Ontologies. A Model Annotation Based Approach* [14], we see how we can enrich design models involved in critical systems development in order to integrate heterogeneous domain constraints. The paper proposes to integrate these domain constraints by enhancing design models with references to domain knowledge. The domain knowledge is modelled by means of ontologies and references are built by annotation mechanism linking design models to domains constraints. The key to the technique is the methodological combination of an MDE approach [15] using Eclipse together with a refinement and proof formal process using Event-B [11].

In *Towards Functional Requirements Analytics* [16], the authors present the design of warehouses for Functional Requirements [17]. The authors advocate the use of a pivot model as there can be a huge heterogeneity between the Functional Requirements stakeholders. Then, they apply the usual warehouse methods based on ETL (Extract, Transform, Load) [18]. The proposal relies on the implementation of a proof of concept based on the Oracle RDBMS using SparQL and the QB4OLAP W3C proposal. This proof of concept is based upon the common Course Management System example provided by the Van Der Bilt university.

In *Heterogeneous Semantics and Unifying Theories* [19], the paper illustrates the use of the Unifying Theory of Programming [20] to combine heterogeneous semantics. The paper reports on two core use cases: the introduction of references in the action part of Hoare logic based on separation logic [21]; and the introduction of the theory of design in CSP [22].

References

1. Hazzan, O., Kramer, J.: The role of abstraction in software engineering. In: Companion of the 30th International Conference on Software Engineering, ICSE Companion 2008, pp. 1045–1046. ACM, New York (2008)
2. Baldwin, W.C., Sauser, B.: Modeling the characteristics of system of systems. In: IEEE International Conference on System of Systems Engineering, SoSE 2009, pp. 1–6. IEEE (2009)
3. Adrion, W.R., Branstad, M.A., Cherniavsky, J.C.: Validation, verification, and testing of computer software. ACM Comput. Surv. (CSUR) **14**(2), 159–192 (1982)
4. Ait-Ameur, Y., Méry, D.: Making explicit domain knowledge in formal system development. Sci. Comput. Program. **121**, 100–127 (2016)

5. Gibson, J.P., Ait-Sadoune, I.: Semantic heterogeneity in the formal development of complex systems: an introduction. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part II. LNCS, vol. 8803, pp. 570–572. Springer, Heidelberg (2014)
6. Attiogbé, C.: Modelling and verifying an evolving distributed control system using an event-based approach. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part II. LNCS, vol. 8803, pp. 573–587. Springer, Heidelberg (2014)
7. Khouri, S., Bellatreche, L., Jean, S., Ait-Ameur, Y.: Requirements driven data warehouse design: we can go further. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part II. LNCS, vol. 8803, pp. 588–603. Springer, Heidelberg (2014)
8. Ait-Ameur, Y., Gibson, J.P., Méry, D.: On implicit and explicit semantics: integration issues in proof-based development of systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part II. LNCS, vol. 8803, pp. 604–618. Springer, Heidelberg (2014)
9. Mammar, A., Laleau, R.: On the use of domain and system knowledge modeling in goal-based event-B specifications. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016, Part I, LNCS, vol. 9952, pp. 325–339. Springer, Heidelberg (2016)
10. Van Lamsweerde, A.: Goal-oriented requirements engineering: a guided tour. In: Fifth IEEE International Symposium on Requirements Engineering, Proceedings, pp. 249–262. IEEE (2001)
11. Abrial, J.R.: Modeling in Event-B: System and Software Engineering. Cambridge University Press, Cambridge (2010)
12. Jureta, I., Mylopoulos, J., Faulkner, S.: Revisiting the core ontology and problem in requirements engineering. In: 2008 16th IEEE International Requirements Engineering Conference, pp. 71–80. IEEE (2008)
13. Mammar, A., Laleau, R.: Modeling a landing gear system in event-B. In: Wiels, V., Ait Ameur, Y., Schewe, K.-D., Boniol, F. (eds.) ABZ 2014. CCIS, vol. 433, pp. 80–94. Springer, Heidelberg (2014)
14. Hacid, K., Ait-Ameur, Y.: Strengthening mde and formal design models by references to domain ontologies. A model annotation based approach. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016, Part I, LNCS, vol. 9952, pp. 340–357. Springer, Heidelberg (2016)
15. France, R., Rumpe, B.: Model-driven development of complex software: a research roadmap. In: 2007 Future of Software Engineering, pp. 37–54. IEEE Computer Society (2007)
16. Djilania, Z., Berkani, N., Bellatreche, L.: Towards functional requirements analytics. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016, Part I, LNCS, vol. 9952, pp. 358–373. Springer, Heidelberg (2016)
17. McGinnis, L.: An object oriented and axiomatic theory of warehouse design. In: 12th International Material Handling Research Colloquium, pp. 328–346 (2012)
18. Vassiliadis, P.: A survey of extract-transform-load technology. Int. J. Data Warehous. Min. (IJDWM) **5**(3), 1–27 (2009)
19. Woodcock, J., Foster, S.: Heterogeneous semantics and unifying theories. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016, Part I, LNCS, vol. 9952, pp. 374–394. Springer, Heidelberg (2016)
20. Hoare, C.A.R., Jifeng, H.: Unifying Theories of Programming, vol. 14. Prentice Hall, Englewood Cliffs (1998)
21. Reynolds, J.C.: Separation logic: a logic for shared mutable data structures. In: 17th Annual IEEE Symposium on Logic in Computer Science, Proceedings, pp. 55–74. IEEE (2002)
22. Hoare, C.A.R., et al.: Communicating Sequential Processes, vol. 178. Prentice-Hall, Englewood Cliffs (1985)