Development of correct by construction variants: managing variants of specifications together with their proofs

Résumé du projet de recherche (Langue 1)

Context of the thesis Software Product Line Engineering (SPLE) is a recent software engineering paradigm that focuses on reuse and variability. Thus products of a same family – also called product line member or variant – are characterized by their common functionalities (features) and also by their differences. The building blocks (assets) associated to these features are assembled according to a given configuration in order to derive a specific product. Software product lines may be defined at different levels of abstraction. The thesis will consider product lines of specifications, more or less abstract, and proofs. The objective is to propose a framework allowing the formal specification of features and assets and the derivation of specific products correct with respect to the selected configuration. In this context assets are specifications and proofs. Developing a formal specification with its corresponding proofs is a difficult work. Formal development environments such as Focalize (http://focalize.inria.fr) aim at facilitating this work providing ways of managing specifications and proofs, in particular their dependencies. These dependency computation is crucial when inheritance and redefinition are allowed because in such cases proofs may have to be erased and done again. Objectives of the thesis In the SPLE context, variability may cover more complex operations than those involved in object oriented frameworks focusing mainly on specialization (underlying inheritance). Thus the main task will be to identify a set of elementary operations reflecting the modifications that can arise when defining a product line and to define the associated constraints and impacts on proofs. In many existing approaches, the product line is defined and analysed under the closed world assumption assuming that all the possible products are known beforehand. This assumption is not realistic since new artifacts may be proposed. Thus the expected framework will have to take the possibility of taking into account new variants (open world assumption) without having to re-organize or re-analyse the complete product line. Furthermore the framework will rely on Focalize. To sum up, the aim of this thesis is to define new ways to reuse both specifications and their proofs within the Focalize proof assistant when a specification varies or evolves. Roadmap The work will first focus on a deep understanding of Focalize and the way it manages the proofs and modification impacts. Then, some specific modifications and their impact on related proofs will be studied. The expected result is an algebra of operations allowing the definition of a new variant. Last a tool allowing these operations will be implemented. Finally, a feature tree will be defined in order to manage the different variants of a specification and proofs. References Klaus Pohl, Günter Böckle, Frank van der Linden. Software Product Line Engineering - Foundations, Principles, and Techniques. Springer 2005. Francisca Losavio, Oscar Ordaz, Nicole Levy, Anthony Balotto. Graph Modelling of a Refactoring Process for Product Line Architecture Design. In proc XXXIX Latin American Computing Conference (CLEI 2013), Venezuela. Virgile Prevosto, Damien Doligez. Algorithms and Proofs Inheritance in the FOC Language. J. Autom. Reasoning 29(3-4): 337-363 (2002). Virgile Prevosto and Mathieu Jaume. Making proofs in a hierarchy of mathematical structures. Proceedings of Calculemus 2003, Rome, Italie.