PROB : un outil de modélisation formelle

MICHAEL LEUSCHEL

Université de Southampton, Angleterre
email : mal@ecs.soton.ac.uk

RÉSUMÉ. *Le développement de modèles formels constitue souvent une étape décisive lors de la conception de logiciels critiques. Il est alors essentiel de vérifier et valider ces modèles formels. Nous présentons* PROB*, un outil développé en programmation logique avec contraintes permettant d'exprimer de tels modèles formels via la méthode B.* PROB *permet une animation complètement automatique des modèles B et peut être employé pour des tests systématiques.* PROB *autorise entre autres les opérations non-déterministes, les instructions ANY, les opérations avec arguments complexes, les ensembles, les séquences, les fonctions, les lambda expressions, les constantes et les propriétés. Les facilités concernant l'animation permettent à l'utilisateur d'accroître la confiance qu'il porte aux spécifications. Contrairement à d'autres outils similaires, l'utilisateur n'a pas à deviner les bonnes valeurs pour les arguments d'opérations ou les variables de choix. Ceci est géré par co-routinage et résolution de contraintes dans les domaines finis. Enfin,* PROB *incorpore un vérificateur temporel et un vérificateur à base de contraintes, pouvant tous deux détecter des erreurs dans les spécifications B.*

ABSTRACT. *The development of formal models is often a key step when developing safety or mission critical software. In this setting it is vital to formally check and validate these formal models. We present* PROB*, a toolset for formal models expressed in the B method, which was developed using constraint logic programming technology.* PROB *allows fully automatic animation of B models, and can be used to systematically check a B model for errors.* PROB *supports B features such as non-deterministic operations, ANY statements, operations with complex arguments, sets, sequences, functions, lambda abstractions, set comprehensions, constants and properties, and many more.* PROB*'s animation facilities allow users to gain confidence in their specifications, and unlike other animators, the user does not have to guess the right values for the operation arguments or choice variables. This is achieved by using co-routining and finite domain constraint solving. On top of the animation features,* PROB *contains a temporal model checker and a constraint-based model checker, both of which can be used to detect various errors in B specifications.*

MOTS-CLÉS : *programmation logique avec contraintes, vérification, méthode B*

KEYWORDS: *constraint logic programming, verification, model checking, B method*