Projet de coopération CNRS-Royal Society 2011

Test de logiciel basé sur des modèles prenant en compte raffinements, états et interactions

Model-based software testing for refinements, states and interactions

MoBaST

Ana Cavalcanti
Department of Computer Science, University of York, York YO10 5GH, England, UK
e-mail: Ana.Cavalcanti@cs.york.ac.uk
M.-C. Gaudel
Laboratoire LRI, UMR8623, 91405 Orsay, France
CNRS, 91405 Orsay, France
e-mail : mcg@lri.fr

Introduction :

L'intérêt de la génération de tests à partir de spécifications formelles de logiciel est maintenant bien établi, et de nombreux travaux ont été développé, concernant différentes sortes de spécifications ou de modèles, différentes relations de conformité, et les outils de génération et de conduite de test correspondants (voir par exemple [HBH 08]).
Ce projet vise à poursuivre une collaboration sur ce sujet qui a démarré en 2006 entre l'Université de York et le LRI d'Orsay.
Les spécifications considérées sont en CIRCUS, un langage développé à York qui permet de combiner états et types de données complexes (inspirés de Z) et processus parallèles communiquant inspirés de CSP [OCW 09]. De plus ce langage définit une notion de raffinement [CSW 03] qui donne tout naturellement une notion de conformité pour le test.
En se basant sur les résultats développés au LRI sur le test basé sur les spécifications formelles [Gau 95, GJ 99, LG 02, GLG 07], le projet se propose d'aborder les questions suivantes :
Quelles sont les stratégies de sélection de tests intégrées applicables à un tel langage qui combine des aspects jusqu'ici étudiés séparément du point de vue du test ? Comment justifier ces stratégies et les hypothèses qui les sous-tendent ?
Comment implémenter de manière bien fondée ces stratégies sur la base des outils de preuve et de génération de tests existants ?
De tels outils sont disponible à York [ZC 08] et au LRI : HOL-TestGen [BW 07], basé sur HOL/Isabelle, est développé au LRI depuis l'arrivée de Burkhart Wolff.

Résultats déjà obtenus :

Nous collaborons depuis 2006 sur les méthodes de test basées sur des spécifications de systèmes intégrant parallélisme, interactions par messages et mémoires partagées, structures de données complexes, en utilisant le langage Circus qui s'inspire de Z, CSP, et qui fournit une notion de raffinements.
Nous avons revisité dans [CG 07] les méthodes de test basées sur CSP, ce qui nous a donné une base pour étudier Circus. La sémantique de Circus étant donnée en UTP [OCW 09], nous avons publié sur le lien entre différentes notions de raffinement dans ce cadre [CG 08] et établi pour Circus les notions de jeu de test exhaustif et les hypothèses de test associées, ce qui a été publié dans Acta Informatica [CG 11]. Ces jeux de tests sont symboliques, et sont composés de traces symboliques contraintes, d'ensemble d'initiales symboliques associées à de telles traces, et d'ensembles d'ensembles d'acceptances symboliques. Leur consistance avec la sémantique opérationnelle de Circus a été démontrée.
Nous co-encadrons la thèse de Abderrahmane Feliachi avec Burkhart Wolff. Il s'agit de développer en Isabelle/HOL des théories pour UTP puis CIRCUS, afin de pouvoir utiliser ensuite l'outil HOL/TestGen pour la génération de tests. La partie sur UTP a été publiée [FGW 10].
Le fait d'utiliser un outil de génération de test basé sur un prouveur de théorèmes puissant comme HOL/Isabelle va permettre de vérifier formellement le bien-fondé des tests par rapport à la sémantique de Circus (validité et non-biais), ainsi que la complémentarité entre les hypothèses de test et les jeux de tests produits.
Par ailleurs, nous pensons que les théories développées seront réutilisables pour d'autres langages basés sur UTP, ou similaires à Z ou CSP.

Description Scientifique et Planning

Cette partie est en Anglais. Elle est commune avec la soumission à la Royal Society.

This project aims at strengthening and consolidating an existing and fruitful collaboration in the area of software testing based on formal specifications.
The considered specifications are in CIRCUS, a language developed in York, which integrates the notions of states and complex data types (in a Z-like style) and communicating parallel processes inspired from CSP.
Morover, the language comes with a formal notion of refinement and allows to take into account abstract specifications and their transitions to models of programs.

On the bases of the theory of formal software testing and the proof and test generation tools developed in LRI the project will address the following questions:
What are the integrated testing strategies applicable to such languages, which combine aspects that have been studied separately, so far, with respect to testing?
How to justify these strategies and their coherence with the underlying test hypothesis?
How to implement them in a well-founded way, starting from the existing proof and generation tools that exist in York and Orsay?

Circus is a process algebra that combines two well-established formalisms: Z for data modelling, and CSP for behavioural specification. Circus is distinctive as a refinement language, since its flexible approach to integrating Z and CSP allow us to write both high-level abstract specifications and program models.

Previously, as foundational work, we have defined (infinite) exhaustive test sets with respect to the Circus notion of refinement and the corresponding hypotheses [CG 11].
To cater for the rich data models of Z, we have symbolic tests and test sets (see the section on previous results), in a novel treatment of concurrency.

We want now to address the problem of the selection of finite test sets. Since, in the exhaustive test sets, we have a symbolic version of the tests, with labels constraining communicated values, it is natural to consider strategies based on constraints decomposition and solving. We propose to go further and address the challenging problem of providing coverage of complex internal data operations, and justify the soundness of the techniques.

We propose the following work packages. The plan is for a two-year collaboration.

WP1. Selection criteria based on exhaustive test sets

This workpackage will involve the implementation of a symbolic animator and prover for Circus probably above HOL/Isabelle on the basis of the denotational and operational semantics of CIRCUS.
It will be used to calculate constrained symbolic traces, initials and acceptances, which are the basic components of the symbolic tests cases in the exhaustive test sets mentioned above.
This implementation will be used to propose and implement several selection criteria of finite subsets of the exhaustive test sets, to develop the corresponding generation algorithms, and finally to experiment them on some case studies.

This work package will be performed within the first year, but the experiments are likely to be performed at the beginning of the second year.

The first year trip schedule is:
Visit of MC Gaudel to York : 25 March-4 April
Visit of A Cavalcanti to Paris: 8-21 July
Visit of A Feliachi to York: one week, end of September
Visit of MC Gaudel at the same time.
Visit of A Cavalcanti to Paris : 2-15 December

WP2. Selection criteria based on specifications.

For every selection strategy, we need to prove that the test sets generated are sound with respect to the formal semantics of Circus, i.e. valid and unbiased.
It is where the use of a generation tool such as HOL/testGen, which is built above the powerful HOL/Isabelle theorem prover is likely to be very productive. In the case of WP1, proof of validity and unbias for the proposed test sets is trivial, because the selection criteria will be expressed as restrictions on the exhaustive test sets.

In this work package, we consider a number of more elaborate criteria, and corresponding selection hypothesis, based on the Circus specification.
Examples of such criteria are: coverage of every Z operation or Z operations sub-cases, various sorts of coverage of inter-process communications, transposition of data-flow criteria in languages such as Circus, definition of transition coverage in presence of state, guard and/or Z operations.

We will study how to express such criteria. The formulations of the criteria have an impact on both the proofs of soundness (validity and unbias) and the generation algorithms.
We will study how to perform such proofs. They will build on results on the correspondence between the operational and denotational models of Circus, and on properties of refinement.
The use of test generation tools based on powerful theorem provers, such as HOLTestGen above HOL/Isabelle, should allow some first attempts of formal verification of the soundness of the tests with respect of the semantics of Circus, as well as of the complementarity of the test hypotheses and the generated test sets.

This work package will be performed during the two years.

The second year trip schedule is similar to the first year but A. Feliachi visit is replaced by a visit to York of B. Wolff, possibly in March-April.

Résumés des Résultats Attendus

Dans ce projet nous voulons étudier des critères de couverture de test spécifiques à Circus, dans le prolongement de la publication [CG 10]. De tels critères prendraient en compte de manière intégrée la couverture des communications inter-processus et des propriétés des données communiquées.
De plus nous étudions avec Abderrahmane Feliachi et Burkhart Wolff comment prendre en compte de tels critères de couverture dans le système de génération de cas de test HOL-TestGen, dans le prolongement de la publication [FGW 10].
Il s'agit de formaliser en Isabelle/HOL la sémantique de Circus qui est donnée en UTP (Unifying Theory of Programming), puis d'utiliser HOL/TestGen pour engendrer des tests sur la base de la théorie du test pour Circus (jeux de test exhaustifs, hypothèses de test) établie en [CG 11]. La sémantique de Circus a été déjà été formalisée en ProofPower par F. Zeyda and A. Cavalcanti (article à paraître dans SCP). L'intérêt de reprendre cette sémantique dans Isabelle/HOL est le lien avec la génération de test et HOL/TestGen.

Références

[BW 07] Achim D. Brucker and Burkhart Wolff. Test-sequence generation with HOL-TestGen - with an application to firewall testing. In Bertrand Meyer and Yuri Gurevich, editors, *TAP 2007: Tests And Proofs*, number 4454 in Lecture Notes in Computer Science, pages 149-168. Springer-Verlag, Zurich, 2007.

[CG 07] Ana Cavalcanti and M.-C. Gaudel. *Testing for refinement in CSP*. In Formal Methods and Software Engineering, ICFEM 2007, volume 4789 of Lecture Notes in Computer Science, pages 151-170. Springer Verlag, 2007.

[CG 08] Ana Cavalcanti and Marie-Claude Gaudel. *A note on traces refinement and the « conf » relation in the Unifying Theories of Programming*. In Andrew Butterfield, editor, Unifying Theories of Programming, Second International Symposium, UTP 2008, Trinity College, Dublin, Ireland, September 8-10, 2008, Revised Selected Papers, volume 5713 of Lecture Notes in Computer Science, pages 42-61. Springer, 2008.

[CG 10] Ana Cavalcanti and Marie-Claude Gaudel. *Specification coverage for testing in Circus.* In Unifying Theories of Programming 2010, volume 6445 of Lecture Notes in

Computer Science, pages 1-45, Shanghai, China, November 2010. Springer Verlag. invited lecture.

[CG 11] Ana Cavalcanti and Marie-Claude Gaudel. *Testing for refinement in Circus*. Acta Informatica, 48(2):97{147, 2011.

[CSW 03] Cavalcanti, A.L.C., Sampaio, A.C.A., Woodcock, J.C.P.: *A Refinement Strategy for Circus*. Formal Aspects Comput. 15(2–3), 146–181 (2003).

[FGW 10] Abderrahmane Feliachi, Marie-Claude Gaudel, and Burkhart Wolff. *Unifying theories in Isabelle/HOL*. In Unifying Theories of Programming 2010, volume 6445 of Lecture Notes in Computer Science, pages 188{206, Shanghai, China, November 2010. Springer Verlag.

[Gau 95] M.-C. Gaudel. *Testing can be formal, too*. In TAPSOFT'95, International Joint Conference, Theory And Practice of Software Development, volume 915 of *Lecture Notes in Computer Science*, pages 82-96, Aarhus, Denmark, 1995. Springer Verlag.

[GJ 99] M.-C. Gaudel, P. R. James, *Testing Algebraic Data Types and Processes: a unifying theory*, Formal Aspects of Computing, 10(5-6), 436-451, 1999.

[GLG 07] M.-C. Gaudel and P. Le Gall. *Testing data types implementations from algebraic specifications.* In Formal Methods and Testing, R. Hierons, J. Bowen, and M. Harman,eds, volume 4949 of *Lecture Notes in Computer Science*. Springer-Verlag, 2007. 209-239

[HBH 08] Hierons, R. M., Bowen, J. P., Harman, M. (Eds.) (2008). *Formal Methods and Testing*. (LNCS Vol. 4949). Springer-Verlag.

[LG 02] Lestiennes, G., Gaudel, M.C.: *Testing processes from formal specifications with inputs, outputs, and datatypes*. In: IEEE International Symposium on Software Reliability Engineering, pp. 3–14 (2002)

[OCW 09] Oliveira, M.V.M., Cavalcanti, A.L.C., Woodcock, J.C.P.: *A UTP semantics for Circus*. Formal Aspects of Comput. 21(1–2), 3–32 (2009).

[ZC 08] F. Zeyda and A. Cavalcanti. *Encoding Circus Programs in ProofPower-Z*. In Unifying Theories of Programming, 2nd International Symposium, volume 5713 of Lecture Notes in Computer Science, pages 217-236. Springer, September 2008