Functional Testing Using OCL Predicates to Improve Software Quality

A. Jalila, Department of Computer Applications, Thiagarajar College of Engineering, Madurai, India
D. Jeya Mala, Department of Computer Applications, Thiagarajar College of Engineering, Madurai, India
M. Eswaran, Department of Computer Applications, Thiagarajar College of Engineering, Madurai, India

ABSTRACT

Formal specification-based testing is one of the major criterions to detect faults in the requirement specification or design of a system. However, due to the non-availability of implementation details, test case generation from formal specifications become a challenging task. This research work has proposed an automated fault-based testing framework to test the specification of a system written in OCL (Object Constraint Language). Accordingly, the possible faults in OCL specification of a system have been anticipated by mutating its method based on predicate-based fault classes. Then, test cases are generated by applying Chaotic Simulated Annealing (CSA) algorithm. In this paper, a novel OCL predicate based fitness function is defined to evaluate the generated test data. The experiment results indicate that the OCL specification based testing adopting CSA form an excellent combination for performing functional testing at the earliest to improve software quality.

Keywords: Chaotic Simulated Annealing (CSA), Fitness Function, Functional Testing, Object Constraint Language (OCL)

INTRODUCTION

The fulfillment of requirement of the user is the major aim of software development. Improper specification and design leads to high cost in the implementation phase (Kuhn, 1999). Therefore, software systems are formally specified using any one of the specification languages, namely OCL, Z, B, VBM, alloy etc. In general, testing activity will be more effective when it is applied from the initial stages of software development. Furthermore, formal specification of the system serves both as an oracle and medium to generate functional test cases. Thus, there is a great demand for specification based testing in the field of formal specification based software development.

OCL has found its major application in precise software modeling (Cabot & Gogolla, 2012). However, the proposed approach extends the application of OCL in specification-based testing. Hence, the delayed fault and failure
detection increases the cost of error correction and maintenance (Frank, Alla, Jürgen, & Robert, 2012). In this paper, SUT would mean the OCL specification of the system which is derived from its requirements specification. In the proposed approach, predicate-based mutation testing is performed. OCL-based testing is preferable due to the following reasons:

- OCL constraints can be easily derived from requirements specification of the SUT.
- OCL describes the model of a system precisely hence, OCL based functional test results will be accurate.
- Test scenarios can be designed effectively.
- Intermediate representation and concretization of OCL predicates require less effort.
- Metrics can be extracted easily from OCL expressions, so as to design the test paths.

In our approach, test scenarios are obtained by transforming OCL predicates into equivalent size abstracts for effective reasoning. Then, mutants are automatically generated for all methods of the SUT by applying fault classes (Tai, Vouk, Paradkar, & Lu, 1994), namely Boolean Operator Fault (BOOF), Incorrect Relational Operator Fault (IROF), Incorrect Parenthesis Fault (ICPF), Incorrect Boolean Variable Fault (IBVF), Incorrect Arithmetic Expression Fault (IAEF). In the proposed approach, test paths are generated based on Direct Class Coupling (DCC) metric (Bansiya & Davis, 1997) value which can be derived from OCL expressions of the SUT.

The proposed algorithm has used CSA algorithm to generate efficient test cases. In this paper, test adequacy criteria based on branch coverage and mutation score are used to evaluate the fitness functions. The following are the major merits of the proposed framework:

- Efficient test data generation using CSA.
- For the purpose of this work, the terms specifications, conditions, constraints and expressions are used interchangeably. On the same lines, specification-based testing would also mean mutation testing or fault-based testing. Similarly, component would mean class of the SUT.

The remainder of this paper starts with the related earlier works and elaborates the background of the proposed OCL specification-based testing. Then we discuss the proposed algorithm and present a proof of the proposed algorithm implementation. Prior to concluding, the proposed work is compared with the earlier work.

**RELATED WORK**

Prior studies show that there are various techniques available to generate test cases from OCL specification of the system.

A theorem-prover based test data generation from OCL specifications has been proposed (Brucker, Krieger, Longuet & Wolff, 2011) which included language features such as recursive query-operations, object graph and equivalent relation. They used the concept called alias closure to derive an object graph. In their approach, object graphs are implicitly represented based on the equivalent relation. They have translated OCL Invariants into Recursive HOL-Predicates. However, in the proposed approach, OCL constraints are translated into equivalent size abstracts for improved reasoning.

According to Benattou, Bruel & Hameur-lain (2002) the partition analysis technique is used to identify the domains of the operation of a specification. In their study, they used mathematics inherent which is derived from various constraints of the OCL specification to generate test data. They reduced the mathematical expression of the OCL specification by transforming it into Disjunctive Normal Form (DNF).
Moving to SaaS: Building a Migration Strategy from Concept to Deployment
[www.igi-global.com/chapter/moving-saas-building-migration-strategy/72217?camid=4v1a](www.igi-global.com/chapter/moving-saas-building-migration-strategy/72217?camid=4v1a)

Applicability of Lehman Laws on Open Source Evolution
[www.igi-global.com/chapter/applicability-of-lehman-laws-on-open-source-evolution/135229?camid=4v1a](www.igi-global.com/chapter/applicability-of-lehman-laws-on-open-source-evolution/135229?camid=4v1a)