A Review of Formal Methods

학번 : 200514170
이름 : 한규희 (T4)

Contents

- INTRODUCTION
- DEFINITION AND OVERVIEW OF FORMAL METHODS
- SPECIFICATION METHODS
- LIFE CYCLES AND TECHNOLOGIES WITH INTEGRATED FORMAL METHODS

Introduction

- Certain precepts leads to better Programs.
- Design methodologies are varied
- Underlying principles are the same
- Understand Core Ideas and the central Foundation
- Core Ideas are invariant and *Formal Methods* define these

Definition and Overview

- Support reasoning about formulae in some language
- Formal language – set of strings over some well defined alphabet
- Proofs – axioms → inference rules → Premises → consequents
- Properties can be proven.

Definition and Overview (Cont.)

- A formal method in software development is a method that provides a formal language for describing a software artifact (for instance, specifications, designs, or source code) such that formal proofs are possible, in principle, about properties of the artifact so expressed.
- Such methods are adaptations of the axiomatic method in mathematics

Definition and Overview (Cont.)

- Use of Formal Methods
  - Record a system's functionality (Z, Larch, Communicating Sequential Processes (CSP) etc..)
  - Specify aspects other than functionality (safety, security etc)
  - Fault tolerance, response time, efficiency, reliability etc can also be addressed.

Definition and Overview (Cont.)

- Tools and Methodology
  - Proofs and programs should be developed in parallel
  - Clearly understood constructions should be used
  - “Cleanroom approach” and heuristics may be used

Definition and Overview (Cont.)

- Limitation
  - Requirements problem
  - Physical Implementation problems
  - Implementation Issues

Definition and Overview (Cont.)

- Requirements problem
  - “You cannot go from the informal to the formal by formal means”
  - Verification possible, not Validation.
  - Formal methods cannot replace the requirements engineer with deep domain knowledge

Definition and Overview (Cont.)

- Physical Implementation problems
  - A physical machine is different from the abstract machine for which the program is made.
  - Proofs limited to a particular machine with limits and real characteristics
  - Compilers cause some problems
  - Bugs in memory, chips
  - Formal methods might never supplant testing

Definition and Overview (Cont.)

- Implementation Issues
  - Users' intentions ←→ Formal Specifications
  - Physical implementation ←→Abstract proofs
    - These gaps create inherent limitations
  - Scaling up to large scale projects is a problem

Specification Methods

- Specification method says what a specification must say
- Language on the other hand determines in detail how the concepts in a specification can be expressed
- Different Methods
  - Semantic Domains
  - Operational and Definitional Methods

Specification Methods (Cont.)

- Semantic Domains
  - Exact rules state what objects satisfy a specification
  - Specification → set of formulae in a formal language
  - Specification languages can be classified by their semantic domains
    - ADT specification languages
    - Process specification languages
    - Programming languages

Definition and Overview (Cont.)

- ADT specification languages
  - used to specify algebras
  - ‘defines the formal properties of a data type without defining implementation issues
- Process specification languages
  - Specify state sequences, streams, sequences, partial orders and state machines
- Programming languages

Definition and Overview (Cont.)

- Model-Oriented Methods
  - Operational Model – Describes a system by providing a model
    - Functions from space of inputs to space of outputs
- Property-Oriented Methods
  - Definitional Models
    - Minimum set of conditions to be satisfied is the specifications
    - Algebraic (ADT) and axiomatic (preconditions and post conditions) models are the two classes.

Definition and Overview (Cont.)

- Use of Specification Methods
  - Customers should be provided English version, not formal version.
  - Details of project and skills of engineers to be considered
  - Operational models closer to programming practice
  - Definitional model harder to construct and consistency and completeness are difficult to establish.

Life Cycles and Technologies

- To get full advantages, Formal Methods should be incorporated in standard
- Two methods of integrating
  - Heavy use of automated tools
  - Nonmechanical, nonautomated proofs
- Division of verification tools
  - Theorem proving tool
  - Model checking tool

Conclusions

- Formal Methods provide
  - More precise specifications
  - Better internal communication
  - Ability to verify designs before execution testing
  - Higher quality and productivity
- Should be incorporated as standard
- Customized solutions may be required

Thank You