Robert L. vienneau

A REVIEW OF FORMAL METHODS

INDEX

- 1. INTRODUCE
- 2. DEFINING AND OVERVIEW OF FORMAL METHODS
- 3.SPECIFICATION METHODS
- 4.LIFE CYCLES AND THECHNOLOGIES WITH INTEGRATED FORMAL METHODS
- CONCLUSION

Introduction

- The 1970s witnessed the structured programming revolution.
- After much debate, software engineers became convinced that better programmers result from following certain precept in program design.

- Formal Methods have the potential of leading to further revolutionary change in practice and hav provided the underlying basis for past change.

Definition and overview of formal methods

- 2.1.Use of Formal Methods
  - They are directly applicable during the require ments, design, and coding phases and have im portant consequences for testing and mainten ance.
  - They have influenced the development and sta ndardization of many programming languages , the programmer's most basic tool.

Definition and overview of formal methods

- 2.1.Use of Formal Methods
  - A broad view of formal methods includes all applications of (primarily) discrete mathematics to engineering problems.

Definition and overview of formal methods

- 2.1.Use of Formal Methods
  - A more narrow definition, better conveys the change in practice recommended by advocates of formal methods.
    - First, formal methods involve the essenti al use of a formal language.
    - Second, formal methods in software support formal reasoning about formulae in the language.

Definition and overview of formal methods

- 2.1.1.What Can Be Formally Specified
  - Formal methods support precise and rigorous specifications of those aspects of a computer s ystem capable of being expressed in the langu age.

Definition and overview of formal methods

- 2.1.1.What Can Be Formally Specified
  - Since defining what a system should do, and understanding the implications of these decis ions, are the most troublesome problems in s oftware engineering, this use of formal meth ods has major benefits.

Definition and overview of formal methods

- 2.1.1.What Can Be Formally Specified
  - Formal methods can be used to specify aspects o f a system other than functionality.
  - For example formal methods are sometimes appl ied in practice to ensure software safety and secu rity properties of computer programs.
  - The benefits of proving that unsafe states cannot arise, or that security is assured, can justify the c ost of complete formal verifications of the releva nt portions software system.

Definition and overview of formal methods

- 2.1.2.Reasoning about a Formal Description
  - Dose a description imply a system should be i n several states simultaneously?
  - Do all legal inputs that yield one and only one output?
  - What surprising results, perhaps unintended, can be produced by a system?

Definition and overview of formal methods

- 2.1.2.Reasoning about a Formal Description
  - Formal methods support formal verification, t he construction of formal proofs that an impl ementation satisfies a specification.
  - The possibility of constructing such formal pr oofs was historically the principal driver in the development of formal methods.

Definition and overview of formal methods

- 2.1.3.Tools and Methodology
  - For proponents of formal methods, the ultim ate end product of software development is n ot solely a working system
  - Specification and demonstrations that the pr ogram meets its specification are of equal im portance.

Definition and overview of formal methods

- 2.1.3.Tools and Methodology
  - A proof is very hard to develop after the fact.
  - Consequently, proofs and programs should b e developed in parallel, with close interconne ctions in their development history.
  - Since programs must be proven correct, only those constructions that can be clearly under stood should be used.

Definition and overview of formal methods

- 2.1.3.Tools and Methodology
  - Formal methods have also inspired the devel opment of many tools.
  - Programs to help maintain and automate pro ofs are an obvious example of such tools.

Definition and overview of formal methods

- 2.1.3.Tools and Methodology
  - in some sense, no programmer can avoid for mal methods, for every programming langua ge is by definition, a formal language.
  - Ever Since Algol 1960 was introduced, standa rds defining programming languages have us ed a formal notation for defining language sy ntax, namely Backus-Naur Form.

Definition and overview of formal methods

- 2.2. Limitations of Formal Methods
  - Given the applicability of formal methods thr oughout the life cycle, and their pervasive pos sibilities for almost all areas of software engin eering, why are they not more widely visible?

Definition and overview of formal methods

- 2.2. Limitations of Formal Methods
  - One issue is pedagogic.
    - Revolutions are no made by conversion, but by the old g uard passing away.
  - One the other hand, it is not the case that the onl y barrier to the widespread transition of this tech nology is lack of knowledge on the part of practit ioners
  - Formal methods suffer from certain limita tions.

Definition and overview of formal methods

- 2.2.1 Requirements Problem
  - In particular, a formal verification can prov e that an implementation satisfies a forma l specification, but it cannot prove that a f ormal specification captures a user's intuiti ve understanding of a system.
  - In other words, formal methods can be used to verify a system, but not to valid ate it.

Definition and overview of formal methods

- 2.2.1 Requirements Problem
  - One influential study found that the three most important problems in software dev elopment are:
    - 1. The thin spread of application domain kn owledge
    - 2. Change in and conflicts between require ments
    - 3. Communication and coordination proble m.

Definition and overview of formal methods

- 2.2.1 Requirements Problem
  - These findings suggest the reduction of infor mal application knowledge to a rigorous speci fication is a key problem area in the develop ment of large systems.
  - Empirical evidence suggests, however, that fo rmal methods can make a contribution to the problem of adequately capturing requiremen ts.

Definition and overview of formal methods

- 2.2.1 Requirements Problem
  - Empirical evidence suggests, however, that formal methods can make a contribution to the problem of adequately capturing requirements.
  - The discipline of producing a formal specification can result in fewer specification errors.
  - Furthermore, implementers without an exceptional designer's knowledge of the application area commit fewer errors when implementing a formal specification than when relying on hazy knowledge of the application.
  - The discipline of producing a formal specification can result in fewer specification errors.

Definition and overview of formal methods

- 2.2.1 Requirements Problem
  - A specification acts as a "contract" between a user and a developer.
  - Using specifications written in a for mal language to complement natur al language descriptions can make this contract more precise.

Definition and overview of formal methods

- 2.2.1 Requirements Problem
  - Finally, developers of automated programmi ng environments, which use formal methods, have developed tools to interactively capture a user's informal understanding and thereby develop a formal specification.
  - Still, formal methods can never replace deep application knowledge on the part of the requ irements engineer, whether at the system or t he software level.

Definition and overview of formal methods

- 2.2.2 .Physical Implementation
  - The second major gap between the abstracti ons of formal methods and concrete reality li es in the nature of any physically existing com puter.
  - Formal methods can verify that an implemen tation satisfies a specification when run on an idealized abstract machine, but not when run on any physical machine.
  - Memory chips and integrated circuits may co ntain bugs.

Definition and overview of formal methods

- 2.2.3 .Implementation Issues
  - The gaps between users` intentions and formal specifications, and between physical implementations and abstract proofs, create inherent limitations to formal methods, no matter how much they may be developed in the future.
  - The introduction of a new technology into a large-scale software organization is not a simple thing, particularly a technology as potentially revolutionary as formal methods.

Definition and overview of formal methods

- 2.2.3 .Implementation Issues
  - Decisions must be made about whether the t echnology should be completely or partially a dopted. Appropriate accompanying tools nee d to be acquired.
  - Current personnel need to be retrained, and n ew personnel may need to be hired.
  - Existing practices need to be modified, perha ps drastically.

Definition and overview of formal methods

- 2.2.3 .Implementation Issues
  - Optimal decisions depend on the organizatio n and the techniques for implementing forma l methods.
  - One scheme for using formal methods on real -world projects is to select a small subset of c omponents for formal treatment, thus finessi ng the scalability issue.

Definition and overview of formal methods

- 2.2.3 .Implementation Issues
  - No matter to what extent an organization de cides to adopt formal methods, if at all, traini ng and education issues arise.
  - Education in formal methods should not be c onfined to degreed university programs for u ndergraduates newly entering the field.
  - Means need to be found, such as seminars an d extension courses, for retraining an existing workforce.

SPECIFICATION METHODS

- Formal methods were originality developed t o support VERIFICATIONS, BUT MANY PROJ ECTS USING FORMAL METHODS HAVE used them only to establish properties of specificat ions.
- This section briefly describes some characteri stics of different methods now available.

SPECIFICATION METHODS

- 3.1 Semantic Domain
  - A formal specification language contains an a lphabet of symbols and grammatical rules tha t define well-formed formulae.
  - These rules characterize a language`s "syntac tic domain." The syntax of a language shows how the symbols in the language ships betwe en them are characterized by the syntax of a l anguage.

Definition and overview of formal methods

- 3.1 Semantic Domain
  - Three major classes of semantic domains exis t.
    - 1. Abstract data type specification languages
    - 2. Process specification languages
    - 3. Programming languages

Definition and overview of formal methods

- 3.2 Model-Oriented and Property-Oriented Methods
  - The distinction between model-oriented and property-oriented methods provides another dimension for classifying formal methods.

SPECIFICATION METHODS

- 3.2 Model-Oriented and Property-Oriented Methods
  - Model-oriented methods have also been desc ribed as constructive or operational.
  - Typically, a model will use abstract mathemat ical structures, such as relations, functions, se ts, and sequences.

SPECIFICATION METHODS

- 3.2 Model-Oriented and Property-Oriented Methods
  - Property-oriented methods are also described as definitional or declarative.
  - A specification describes a minimun set of conditi ons that a system must satisfy.
  - Any system that satisfies these conditions is f unctionally correct, but the specification dose not provide a mechanical model showing how to determine the output of the system from t he input.

Definition and overview of formal methods

- 3.3 Use of Specification Methods
  - In general, formal methods provide for more pre cise specifications.
  - Since the earlier a fault is detected, the cheaper i t can be removed, formal specification methods c an dramatically improve both productivity and q uality.
  - In particular, customers should be presented with the English version, not a formal specification.
  - Choosing between model-oriented and property-oriented methods also depends on project-specif ic details and experience.

4.0 LIFE CYCLES AND TECHNOLOGIES WITH INTEGRATED FORMAL METHODS

- Two methods of integrating formal methods in software processes can be distinguished:
  - One with heavy use of automated tools
  - and the other with non-mechanical, non-automated proofs.

4.0 LIFE CYCLES AND TECHNOLOGIES WITH INTEGRATED FORMAL METHODS

- ## 4.1 Verification Systems and Other Automated Tools
  - An automated verification system provides a means for the user to demonstrate the existence of a formal proof of a software system.
  - Another set of tools support model checking.
  - Model checking tools overcome state explosion problem in practice by the use of symbolic techniques.

4.0 LIFE CYCLES AND TECHNOLOGIES WITH INTEGRATED FORMAL METHODS

- ## 4.2 The Cleanroom as a Life Cycle with Integrated Use of Formal Methods
  - **The Cleanroom methodology intergrates non-mechanized formal methods into the life cycle.**
  - Specification developed by the Cleanroom process include:
    - Explicit identification of functionality to be included in successive releases
    - Failure definitions, including level of severity
    - The target reliability as a probability of failure-free operation for a specified time
    - The operational profile for each increment, that is, a model of user behavior of the system
    - The reliability model that is applied in system testing to demonstrate reliability.

Conclusions

- This report has briefly surveyed various formal methods and the con ceptual basis of these techniques.
- Formal methods can provide:
  - More precise specifications
  - Better internal communication
  - An ability to verify designs before executing them during test
  - Higher quality and productivity
- knowledge of formal methods is needed to completely understand t hese popular technologies and to use them most effectively. These t echnologies include:
  - Rapid prototyping
  - Object Oriented Design (OOD)
  - Structured programming
  - Formal inspections.

Conclusions

- The full-scale use, transition, and cost-effective use of formal methods is not fully understood. An organization whose leaders can figure out how to effectively integrate formal methods into their software process will be likely to produce higher quality software and thereby gain a competitive advantage