International Journal of Information and Computation Technology.
ISSN 0974-2239 Volume 3, Number 11 (2013), pp. 1217-1224

http://www. irphouse.com /ijict.htm

Improving Software Requirements through Formal Methods: A Review

S.W.A. Rizvi[1], R.A. Khan[2] and R. Asthana[3]

[1]School of Engineering, Babu Banarasi Das University, Lucknow, U.P., INDIA.
[2]Department of Information Technology, Dr. Bhimrao Ambedkar University, Lucknow, U.P., INDIA.
[3]Department of Electrical Engineering, Babu Banarasi Das National Institute of Technology & Management,Lucknow, U.P., INDIA.

Abstract

Almost twenty five years later, eliciting, representing and organizing software requirements remains one of the most challenging problems in software development. Many requirements errors are passed undetected to the later phases of the life cycle and correcting these errors during or after implementation have been found to be extremely costly. Studied have suggested that formal methods have tremendous potential for improving the clarity and precision of requirements specifications and in finding important and subtle errors. The aim of this paper is to systematically review the available literature exploiting the benefits of formal methods in improving software requirements. The paper presents a systematic literature review comprising of four phases through which appropriate papers have been selected. The review highlights the role of formal methods in improving software requirements. At the end, paper presents findings and observations identified, during literature review, followed by motivation and future work.

Keywords: Software Requirements, Formal Methods, Requirements Elicitation, Systematic Literature Review.

1. Introduction

The complexity of modern software systems raise rapidly over recent years as a result of rising number of requirements for a new system. Complexity of requirement will rise up the difficulty of system development process. This is crucial for highly reliable systems whose development often requires the critical areas. (Roslina and Noraziah, 2010) In a landmark article published in 1987, (Fred Brooks, 1987) states that *"The hardest single part of building a software system is deciding what the requirements are …… No other part of the work so cripples the resulting system if done wrong …..[or] is as difficult to produce and hard to fix later on."* It has also stated that 'early defect fixes are typically two orders of magnitude cheaper than late defect fixes'. (Rzepka, 1989)

In a study Lutz found that safety-related software errors arose most often from inadequate or misunderstood requirements. (Lutz, 1993) It is also clear that conventional techniques fail to catch many requirements errors. (Kelly et al., 1992) However, the use of formal methods in real industrial projects is increasing. New software engineering has served formal methods in the development of critical-safety systems. Formal method forms the basis of developing reliable software for critical systems because these methods are based on mathematics and logic. Therefore they are provable. (Nami and Malekpour, 2008) As the awareness of Formal Methods is growing, so is the research literature on various mechanisms, challenges and strategies of handling them. However, there has not been any significant effort to systematically identify, synthesize, and report the literature on the use of Formal Methods to improve requirements. To address this research gap, this systematic literature review seeks to collect and compare existing evidence on formal methods, with the aim to provide researchers with a direction of future research and practitioners with advice in formal method technology adoption. In this review, authors only investigate papers which solely focus on FM applications to refine quality requirements.

The paper is structured as follows: Section 2 describes the research method used in this systematic review. In Section 3, individual papers are reviewed, while Section 4 discusses findings and observations from the review regarding improvements in software requirements through formal methods and the paper concluded with future work in section 5.

2. Research Method

This study has been carried out as a Systematic Literature Review (SLR). A SLR is a methodical way of identifying, assessing, and analyzing available literature that is relevant to a particular research problem. (Kitchenham, 2007) A SLR includes activities such as planning the review, search strategy and search, selection of studies and quality assessment. All these steps are followed in the following sub-sections.

2.1 Data Sources and Search Strategies

The search strategy included electronic databases and manual searches of conference proceedings. The following electronic databases were searched:

- ACM Digital Library (portal.acm.org)
- ScienceDirect – Elsevier (www.sciencedirect.com)
- IEEE Xplore (ieeexplore.ieee.org)
- SpringerLink (www.springerlink.com)

In addition, the proceedings of the International Symposiums on Formal Methods were manually searched. These symposiums are focused on formal methods and organized by Formal Methods Europe (FME) roughly every eighteen months. Besides it, volumes of International Requirements Engineering Conference which is the major event in requirements engineering were also searched. The process for selection of studies comprised four phases. In phase 1, databases are searched using the relevant search terms. The search was performed on April 15, 2013 and resulted in a total of 2245 unduplicated papers.

2.2 Publication Selection

After getting 2245 relevant studies from phase 1, the first author went through the titles of all the 2245 studies in the second phase, in order to know their relevance to the Systematic Literature Review. In this phase, articles with titles that indicated clearly that the articles were out of scope of the SLR were excluded. To minimize the threat of excluding relevant papers, the first author randomly selected two sample sets (with different papers) of 10% of the excluded papers. The second and third authors were provided with one sample set each to include or exclude papers. Any disagreement between the authors was resolved by discussion that included all three researchers. At the end of second phase, 1072 relevant titles were identified. During the third phase, the first author reviewed all 1072 abstracts, and the second and third author reviewed 25% of the excluded papers. Six papers were up for discussion in phase 3 and one paper was added to the included papers. At the end of phase III, 205 papers were left for the last phase of the selection process.

2.3 Publication Screening and Quality Assessment

The following screening criteria, inspired by (Tore and Torgeir, 2008), were used to ensure the quality of the papers and to exclude unrelated research papers.

- SC1: Is the study focusing on improving requirements?
- SC2: Are the research questions, objectives of the study and aims well defined?
- SC3: Is the studied context well defined?

On the basis of the above screening criteria 18 papers out of the 205 papers are finally selected. Because of the restriction of maximum four pages, author has presented the individual review of only five studies out of 18 in the following section, while the findings and observations are being compiled from all the eighteen.

3. Related Works

Research in a particular field requires an elaborate review and study of literature related to that subject. A systematic review of the literature provides information regarding, what has been done in the area, leading to a significant investigation. Detailed and careful reviews of the experts and researchers also promote greater understanding of the field, procedures, methods and algorithms and enable to frame useful hypothesis.

- In a technical report (Easterbrook et al., 1997) of the research (carried out in part by the Jet Propulsion Laboratory, California under a contract with the NASA) Steve Easterbrook, described three case studies in the lightweight application of formal methods to requirements modeling for spacecraft fault protection system. The formal methods were applied very early in the requirement engineering process, to validate the evolving requirements. The results were fed back into the projects, to improve the informal specifications. In all three cases, formal methods enhanced the existing verification and validation processes, by testing key properties of the evolving requirements, and help to identify weaknesses.

From the paper it is noticed that each case of formal modeling was carried out by a small team of experts who were not part of the development team. Results from formal modeling were fed into the requirements analysis phase, but formal specification languages were not adopted for baseline specifications.

- In (Fatwanto, 2012), author proposes a new method for translating software requirements specified using natural language to formal specification. Requirements specification written in a scenario-like format will be transformed into class diagram's components**.**

From the paper it is noticed that the proposed method has at least two limitations. First, it can only translate software requirements specified in the Concern-Aware Requirements Engineering (CARE) format only. Second, the translation cannot translate requirements specification into a complete set of class diagram. It can only transform requirements specification into classes along with their respective attributes. It still lacks the capability to obtain relationships among classes, class associations, and class generalization or specialization.

- In (Dubravk, 2007) author proposed an approach to derive formal specifications of reactive systems from their informal requirements. The paper also proposed a new requirement language, and showed how to transform the informal requirements of a reactive system into requirements written in this new intermediate language. The derived requirements allow to better structure the informal requirements. Subsequently the author showed how these requirements are then systematically translated into a formal specification in the B- Method. The author also validated the proposed approach through a case study.

From the paper it is noticed that the paper has used a different concept of intermediate requirement language that works as bridge between informal and formal requirements. Also the approach used distinguishes requirements from specifications. The initial informal requirements are transformed through an intermediate requirement language into a formal specification.

- In (Jingang and Shenghui, 2010), authors specified and verified the design of library management system of Beijing University of Technology using Prototype Verification System (PVS). In the paper author described the requirements of the system and gave its Entity Relationship (E-R) model, then designed the formal specification of the E-R model and database operations based on the requirement analysis. Finally the author verified the design by proving some critical properties according to the specifications.

From the paper it is noticed that PVS has provided a well-integrated environment for development and analysis of formal specifications. It is beneficial to formal specification and verification of system requirement and system design.

- In (Mat et al., 2012) authors describe an application of the SOFL (Structured Object-Oriented Formal Language) approach to the construction of a specification during requirements analysis. In order to describe on how this approach can be applied to capture requirements using SOFL easily, author used a case study to develop an examination monitoring system for construct abstract requirements.

From the paper it is noticed that the study demonstrates the suitability of SOFL to capture detail requirements and provides with an insight into the knowledge of how SOFL approach can be effectively used. Also SOFL is straightforward, easy to follow, and provides simple formal notation for developing specifications. But the study only limited to informal and semi-formal specifications and not formal. Therefore using other approaches, such as UML and B might be more fruitful.

After going through above paragraphs it can be easily inferred that the research is going on continuously in the direction of improving software requirement development process through the applications of formal methods and still there is a lot of scope to further contribute in the same area.

4. Findings and Observations

From the reviews presented in previous section it can be inferred that there is growing interest in formal methods because they offer rigorous support of computer system development. Formal methods are particularly desirable in safety-critical applications such as process control, aviation, medical systems, railway signaling and many others. It is difficult to find an application that would not benefit from the rigor brought by formal methods. Adopting formal methods in a software company is more a strategical and methodological issue than a technical one. The author does not believe or advocate

the widespread use of these methods in the software industry in general; however their application should instead be considered when reliability, safety or security is a concern. Conscientious industrial applications of formal methods have already been conducted successfully in key areas that have become flagship application areas.

Formal Specifications and verification are not easy or cheap, but the real cost has to be considered in the long term. On the other hand, their conclusions have to be taken with care, formal methods can only be used to specify or prove what was carefully stated beforehand, and cannot be used to reason about what was not. Formally specifying and verifying a whole system is unlikely to be feasible or even reasonable. The advisable practice is to determine the critical parts of the system to be designed and validated, and to apply formal methods on those parts only.

5. Conclusion and Future Work

Every discipline must learn as much, if not more, from its failures as its successes. In this spirit the literature has been reviewed systematically, to better understand past work and outlines possible avenues for future success. The review of the role of formal methods to improve requirements presented in the paper highlights the strengths of formal methods in a way that makes the requirements unambiguous, consistent, complete and precise.

Specifically, it could be concluded that the use of formal methods in software development is going to be a continuing challenge for many years to come. One important observation that is being noticed in this review is implementing formal methods early in software development life cycle specially in requirements elicitation. Therefore, there appeared to be a need for establishing an alignment between requirement elicitation and formal methods, that enables organizations to work on requirements that are more unambiguous, complete, verifiable, consistent, modifiable and traceable. Improving the ability to perform elicitation will also improve the likelihood that developed system will meet their intended customers' needs.

References

[1] A. Fatwanto (July, 2012), "Translating Software Requirements from Natural Language to Formal Specification", *IEEE Conference on Computational Intelligence & Cybernetics*, Bali, pp. 148-152.

[2] A. R. Mat, M. A. khairuddin, A. B. Masli and M. N. Jambli (June, 2012), "Applying SOFL to Construct Requirements Specification for Examination Monitoring System", *3rd International Conference on Software Engineering and Service Science (ICSESS 2012)*, Beijing, pp. 71-74.

[3] B. A. Kitchenham (July, 2007), "Guidelines for performing Systematic Literature Reviews in Software Engineering, *Tech. Rep., EBSE-2007-001. UK URL<http://www.dur.ac.uk/ebse/>*.

[4] D. Tore and D. Torgeir (Aug., 2008), "Empirical Studies of Agile Software Development: A Systematic Review", *Information and Software Technology,* **50**, *9-10,* pp. 833-859.

[5] F. P. Brooks (April, 1987), "No Silver Bullet: Essence and Accidents of Software Engineering", *IEEE Computer,* **20,** *4*, pp. 10-19.

[6] I. Dubravk (July, 2007), "Deriving Formal Specifications from Informal Requirements", *Proceedings of the 31st IEEE Annual International Computer Software and Applications Conference,* Beijing, pp. 145-152.

[7] J. C. Kelly, J. S. Sherif and J. Hops (Feb.,1992), "An Analysis of Defect Densities Found during Software Inspections", *Journal of Systems and Software,* **17**, *2,* pp. 111-117.

[8] M. R. Nami and A. Malekpour (2008, July), "Formal Specification of a Particular Banking Domain with RAISE Specification Language", *IEEE Symp. on Computers and Communications,* Marrakech, pp. 695-699.

[9] M. S. Roslina and A. Noraziah (2010, May), "A Venn Requirement Language for User Requirement", *IEEE International Conference on Electronic Computer Technology (ICECT-2010),* Kaula Lumpur, pp. 223-227.

[10] N. Jingang and S. Shenghui (Dec., 2010), "Design Verification of BJUT Library Management System with PVS", *IEEE Int. Conference on Computational Intelligence and Security,* Nanning China, pp. 624-628.

[11] R. R. Lutz (Jan., 1993), "Analyzing Software Requirements Errors in Safety-Critical Embedded Systems", *Proc. of the IEEE International Symposium on Requirements Engineering,* San Diego, pp. 126-133.

[12] S. Easterbrook, R. Lutz, R. Covington, J. Kelly, Y. Ampo, and D. Hamilton (1997), "Experiencing Using Light-Weight Formal Methods for Requirements Modeling", *Technical Report of the research (carried out in part by the Jet Propulsion Laboratory, California under a contract with the NASA).*

[13] W. E. Rzepka (Jan., 1989), "A Requirements Engineering Testbed: Concept, Status, and First Results", *Proc. of the 22nd Annual Hawaii International Conference on System Sciences,* Kailua-Kona, pp. 339-347.