International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 10, October 2015
Available at http://internationaljournalofresearch.org

# Formal Methods of Software Development in the 2000's: A case study of Nigeria

Zainab Magaji Musa
Masters student, Universiti Sultan Zainal Abidin, Terengganu, Malaysia
EMAIL: zeemusa5@gmail.com
Nordin M A Rahman
Associate prof., Universiti sultan Zainal Abidin, Terengganu, Malaysia
EMAIL: mohdnabd@unisza.edu.my

## Abstract

*Formal methods of software engineering involves the application of mathematical techniques in a software life cycle. It is not only restricted to softwares as it is applicable in hardware requirements and design specifications. Despite their proven importance, formal methods are not applied as expected in this era. Most software developers do not apply them in their work. This paper gives an overview of formal methods. It also gives a detailed explanation of the level at which information technology (IT) professionals in Nigeria apply formalmethods during their real life work of systems development. The statistical results given in this paper are obtained through a statistical analysis of the data obtained from the questionnaires distributed to 250 computing sciences and information technology professionals who are working in various IT organizations in Nigeria. Furthermore, the authors point out some suggestions which when followed will lead to the success of formal methods in present and future world of computing.*

**Key Words:**

Formal methods; software; hardware; millennium

## Introductions

Formal methods are applicable in a wide variety of context to both hardware and software. They are those techniques that results in systems of highest integrity when applied correctly [1]. This have been proved through the steadily growing success stories of using formal methods for real systems, ranging from reengineering of existing systems to the development of new systems [2]. They are useful at several levels of abstraction in software development ranging from requirements capture to specification design, coding, compilation and the underlying digital hardware itself [3].

Despite the successes obtained in applying formal methods, they are still not applied at the expected level. Taking a lookat the many different systems are which are developed annually in this era, one will find out that, only a very low percent of them were developed through formal methods application.

Such an important aspect of software engineering should not be neglected without giving to it much concern; updating and improving it to eradicate is limitations so as to fully benefit from its function of generating systems of high integrity.

This study involves a survey which is made among 250 IT professionals in Nigeria. The respondents are professionals of different aspects of computer science and information technology. The main aim of the study is to find out the level at which formal methods are applied in Nigeria; the degree of the skills of using formal methods among the IT professionals in Nigeria; the level at which formal methods are thought in A-level IT courses; the availability of formal methods research materials (text books, articles, reports etc) in university libraries.

## What are formal methods?

According to Lan Sommerville [4], formal methods refer to any activity that rely on mathematical representations of software including formal system specification, specification analysis and proof, transformational development and program verification.

Formal methods allow the logical properties of a system to be predicted from a mathematical model of the system by means of logical calculation, which is a progress analogous to numerical calculation [3]. This is to say using formal methods makes it possible to calculate whether a certain description of a system is internally consistent, whether certain properties are consequences of proposed requirement, or whether requirements have been interpreted correctly in the derivation of a design [3].

Formal methods are popularly used in the area of critical systems development where emergent system properties such as safety, reliability and security are very important [4].

From the definition given by Sommerville, it is clear that formal methods comprises of different techniques. In this paper, emphasis will be given on formal specification and formal validation.

## Formal Specification

Formal specification is the major component of formal methods because all other components depend on it [4].

Formal specification is the expression of a collection of properties that a system should satisfy in a formal language and at a level of abstraction [2]. It is a concise description of the behaviour and properties of a system written in a mathematically based language, specifying what a system is supposed to do as abstractly as possible, thereby eliminating distracting details and providing a general description resistant to future modification [3].

It is use to provide an unambiguous and precise supplement to natural language descriptions. It can be rigorously verified and validated leading to early detection of specification errors during a system development process [5].It reduces the overall development cost and time since errors that will occur after the system development have been corrected at an earlier phase. Thus, systems that are developed through formal specification have high integrity, efficiency and reliability [3], [4] and [5].

Formal specifications use mathematical notation to describe the system in a precise way. Formal specification languages consist of Vienna Development Method (VDM), OBJ, Larch, Communicating Sequential Processes (CSP), Lotos, Petri-nets, Z and B [4].

## Reasons for Using Formal Specifications

The question why formal methods should be used is often discussed. Already the fact that a specification must be an exact description of a problem might be reason enough but a better argumentation is given by [6]:

Software engineers strive to be true engineers; true engineers use appropriate mathematics; therefore, software engineers should use appropriate mathematics. Thus, given that formal methods is the mathematics of software, software engineers should use appropriate formal methods [6].

## Advantages of Using Formal Specifications

i. Formal specifications allow defects in requirements and designs to be detected earlier than they would be otherwise and greatly reduce the incidence of mistakes in interpreting and implementing correct requirements and designs [3]. Unlike informal specification which is often ambiguous with much difficulty in detecting and correcting errors, a formal specification is precise; meaning that even if such a specification is wrong (i.e., not what the user required), it is easier to tell where it is wrong and correct it [7].
ii. Using a formal notation increases the understanding of the operation of a system, especially early in a design. Formal specification helps to organize the thoughts of a designer, making clearer, simpler designs possible [7].
iii. The likelihood of errors in a design is reduced because errors may be pinpointed more easily. The number of times round the design–implementation–testing cycle would be reduced since more errors will be found and corrected at the design stage [7].
iv. The quality of documentation of the system can be improved. By using the formal design as a basis for the manual for a system, it is likely that less information will be left out. The final document should include a prose description relating the formal text to the real world [7].
v. The overall cost of development will be lowered. Errors corrected at the design stage are much cheaper to correct than if they are found later [7].

## Formal Validation

Formal validation is the act of proving or disproving the correctness of a system with respect to a certain formal specification using mathematical methods [8].

The formal specification of a system are verified to ensure that they are written correctly. They are validated to prove consistency and completeness using formal validation techniques before system design andimplementation.Formally validating the properties of a system specification involves showing that the properties can be derived from the specification using mathematical logic rules [9].

There are several approaches to formal verification but the two fundamental ones are model checking and deductive verification (theorem proving) [8]. Both methods can be done using different automated tools. A few examples of tools used in model checking are CADP (construction and analysis of distributed processes), CPAchecker, ECLAIR, FDR2, Java Pathfinder, LTSmin, Promela, mCRL2, UPPAAL language [10]. While those use in theorem proving are Z/EVES, E, Otter, SETHEO, Vampire, SPASS.

## Account of Formal Methods

Formal methods have been in existence since the good olden days of computing science. In the late 1940’s, Turing observed that reasoning about sequential programs was made simpler by annotating them with properties about program states at specific points [2]. In the 1960’s Floyd, Hoare and Naur recommended using axiomatic techniques to prove programs meets their specifications. In the 1970’s Dijkstra used formal calculus to aid in the development of non-deterministic programs. Since then, the interest in the use of formal methods in systems development continued to grow [2].

In the 1980’s, many software engineering researchers proposed that using formal development methods was the best way to improve software quality. They predicted that a large proportion of software would be developed using formal methods. Unfortunately for them, their predictions has not come true[4].

Sommerville [4] stated four main reasons for the failure of this prediction as:

1. Successful software engineering
2. Market changes
3. Limited scope of formal methods
4. Limited scalability of formal methods

According to Lamsweerde [2], the prediction failed to succeed because of the following reasons:

1. Limited scope of formal methods
2. Poor separation of concerns
3. Low level ontologies
4. Isolation
5. Poor guidance
6. Cost
7. Poor tool feed back

Furthermore, he also point out some requirements and challenges to meet so as to make formal methods to become essential for future engineering and reengineering of higher quality softwares. They are:

1. Constructiveness
2. Support for comparative analysis
3. Integration
4. Higher level of abstraction
5. Richer structuring mechanism
6. Extended scope
7. Separation of concerns
8. Lightweight techniques
9. Multi paradigm specification
10. Multi button analysis
11. Multi formal specification
12. Reasoning in spite of errors
13. Constructive feedback from tools
14. Support for evolution
15. Support for reuse
16. Measurability of progress

Bowen and Hinchey [1] also drawn up 10 commandments which they feel will greatly increase the likelihood of success of formal methods if adhered. These rules are as follows:

1. Thou shalt choose an appropriate notation.
2. Thou shalt formalize but not over formalize
3. Thou shalt estimate cost
4. Thou shalt have a formal method guru
5. Thou shalt not abandon thy traditional development methods
6. Thou shalt document sufficiently
7. Thou shalt not compromise thy quality standards
8. Thou shalt not be dogmatic
9. Thou shalt test, test and test again
10. Thou shalt reuse

But these measures cannot be influential in improving formal methods unless if formal methods are tightly gripped with much concern in the world of computing.

## Analysis of the rate of formal methods application today

In order to find out the present state of formal methods in Nigeria, we made a survey of IT organizations in the different part of country. A total of 250 questionnaires with a response rate of 100% (n=250) were distributed among different professionals of information technology and computing sciences in the country. The questionnaire is divided into two sections viz: socio demographic questions section and formal methods questions section. Thus, the analysis will also be done based on the two sections.

1. Analysis of Socio Demographic Information

| Item | Frequency | Percent (%) |
|---|---|---|
| **Qualification** | | |
| PhD | 25 | 9.7 |
| MSc | 70 | 27.1 |
| BSc | 116 | 45.0 |
| Diploma | 18 | 7.0 |
| Others | 21 | 8.1 |
| **TOTAL** | **250** | **100** |
| **gender** | | |
| Female | 84 | 32.6 |
| Male | 166 | 64.3 |
| **Total** | **250** | **100** |
| **Age** | | |
| 25-30 | 74 | 28.7 |
| 31-35 | 40 | 15.5 |
| 36-40 | 39 | 15.1 |
| 41-45 | 14 | 5.4 |
| 46-50 | 27 | 10.5 |
| 51-55 | 26 | 10.1 |
| 56-60 | 23 | 8.9 |
| Above 60 | 7 | 2.7 |
| **Total** | **250** | **100** |

Table 1: Socio Demographic Information of the Respondents

Out of the 250 respondents 45% of them have a first degree as their highest qualification. 27% of the respondents have a master's degree as their highest qualification. 9.7% of the respondents are PhD holders. Those with diploma as their highest level qualification constitutes 7% of the respondents. And the remaining 8.1% of the respondents claim to have other qualifications different from the ones stated on the questionnaire.

The respondents are from different genders with 166 males and 84 females. 28.7% of the respondents are within the age range of 25-30, 15.5% of them are within the age range 31-35, and those within the age range of 36-40 have 15.1% of the respondents. 5.4% of the respondents fall within the age range of 41-45 while 10.5% are in the range of 46-50 years. 10.1% of the respondents are between 51-55 years and 8.9% of them are 56-60 years. Those that are more than 60 years of age are only 2.7% of the respondents.

In summary, the respondents are male and femaleprofessionals with knowledge of computing where majority of them are youths.

2. Analysis of formal methods information

| Option | Frequency | Percent (%) |
|---|---|---|
| **How did you come to know about formal methods?** | | |
| from school | 70 | 27.1 |
| from colleagues | 53 | 20.5 |
| from research | 81 | 31.4 |
| others | 46 | 17.8 |
| Total | 250 | 100 |
| **How good are you in formal methods?** | | |
| poor | 136 | 52.7 |
| fair | 68 | 26.4 |
| good | 26 | 10.1 |
| excellent | 20 | 7.8 |
| Total | 250 | 100 |
| **Have you ever apply formal methods in real life work?** | | |
| once | 21 | 8.1 |
| twice | 10 | 3.9 |
| several times | 4 | 1.6 |
| never | 215 | 83.3 |
| Total | 250 | 100 |
| **Formal methods are very useful in software engineering** | | |
| strongly agree | 62 | 24.0 |
| agree | 140 | 54.3 |
| strongly disagree | 23 | 8.9 |
| disagree | 25 | 9.7 |
| Total | 250 | 100 |
| **How can you rate the availability of formal methods research materials in Nigeria** | | |
| Poor | 158 | 61.2 |

| | | |
|---|---|---|
| Fair | 92 | 35.7 |
| Good | 00 | 00 |
| Excellent | 00 | 00 |
| Total | 250 | 100 |

Table 2: formal methods information

From the information provided in table 2 it is clearly shown that majority of the respondents got to know about formal methods of software engineering through research. This is because 31.4% of the respondents choose the “from research” option of the questionnaire. Next are those that got to know about formal methods from the schools they attended. They constitute 27.1% of the respondents. 20.5% of the respondents know formal methods through interaction with their colleagues who are good in the subject matter. They fewest group are those that got to know about formal methods through other means which are not specified in the questionnaire.

Taking a look at the level of literacy of the respondents, and the percentage of them that knew formal methods from schools, one can say that, formal methods are not well treated in Nigerian institutions of learning.

52.7% of the respondents are poor in formal methods. 24% of them have a fair ability of working with formal methods. 10.1% of the respondents are good in formal methods while only 7.8% of them claim to have excellent skills of working formal methods of software engineering. This result proves that only 17.9% of the respondents will be able to apply formal techniques of systems development in real life.

Table 2 also shows that 83.3% of the respondents have never apply formal methods in their real life work. 8.1% and 3.9% of the respondents apply them once and twice respectively. Only 1.6% of the respondents applied the techniques several times. From this result, one can see that, not all those that are good in formal methods apply them in their real life work. Because, a total of 17.9% of the respondents will be able to apply formal techniques of software engineering but only 13.6% of them have apply them once or twice or several times.

Furthermore, the table also shows that majority of the respondents carrying 78.3% support the fact that formal methods are very useful. And the remaining 18.7% of them did not support the fact. This means that, even though majority of the respondents are not good formal methods and cannot apply them, they agree that applying them is advantageous.

Table 2 finally shows that the respondents claim that formal methods research materials are not sufficiently available in Nigeria.

## Results Obtained from the Analysis

The analysis of the survey clearly shows the following results:

1. Formal methods are not part of the curriculum of computing science courses in most institutions of learning in Nigeria.
2. Most of the professionals of computing science in the country have poor skills of formal methods. This is because best training is obtained from learning institutions and most of the ones that are available in the country have failed in this aspect.
3. Formal methods are not well applied in real works. The reason for this is not hidden as most of the professionals lack good skills of applying the methods. Although, even the ones that claim to be good in the aspect do not apply them.
4. Despite the fact that formal methods are neglected, they are still regarded as useful methods that can lead to high quality systems.
5. Thereis a little availability of research materials of formal methods in the country. This might be the reason why the professionals of computing sciences in the country have poor knowledge of the said methods.

Having realized the problems associated with the failure of formal methods application, the authors propose the following solutions:

1. Formal methods should be introduced in the curriculum of computing science courses in all institutions of learning.This will lead to the production of professionals that are good in formal methods and can thus apply them in their work.
2. The research materials and studying materials of formal methods should be well provided. This will help interested persons to learn the skills of applying them on their own as most of the respondents claim to know about formal methods through research.

Through the above suggestions, the knowledge and skills of formal methods will be increased among individuals. As such, the experts of formal methods will be able to upgrade and revised them so as to eradicate their present limitations.

## Conclusion

Formal methods failed to succeed as predicted by many scholars. Although they are very useful to both hardware and software development cycle, they are neglected in this era and if care is not taken, they will also be forgotten and hence, become unknown in the future.

This paper gives an over view of what formal methods are. It also gives the result of a survey which is made in Nigeria among IT and computing science professionals to get a statistical proof of the rate at which formal methods are known and applied in this era.

## References

[1] Bowen J.P and M.G. Hinchey, (1995). “Ten Commandments of Formal Methods”. *Computer*, 28(4): 56-63.

[2] Lamsweerde, A. V. (2000). Formal specification: a road map. ACM 2000 article. Pp 223-234. citeseerx.ist.PSV,ed/viewdoc/summary?doi=10.1.1.35.4492/ Accessed on 17/05/2014.

[3] NASA (1995). Formal methods specification and verification guide book for software and computer systems. Volume: planning and technology insertion

[4] Somerville, I. (2009). Software Engineering, 9th edition. Ch27.

[5] Jusoh J.A., Saman M.Y.M. and Man M. (2011). Formal validation of DNA database using theorem proving technique. International Journal of the computer, the internet and management. 19:74 – 78

[6] Bowen J. and Stavridou V. (1993). Safety-critical systems, formal methods. In proceedings of the 16th AIAA/IEEE digital avionics systems conference. 1:1-22

[7] Bowen, J. (2003). Formal Specification and Documentation using Z: a case study approach. Pp 1-10

[8] Li G. (2010). Formal verification of programs and their transformations. PhD Thesis, University of Utah.

[9] Smith, G. (1995). Formal verification of object-Z specifications. Technical report no 95-55. Software verification center, dept. of Computer Science, University of Queensland, Australia.

[10] Clarke E., Grumberg O. and Long D. (2014). Model checking. http://www.cs.cmv.edu/~modelcheck/ed-papers/mc.pdf. Accessed on 04/07/2014