Journal of Software Engineering and Applications, 2015, 8, 531-538
Published Online October 2015 in SciRes. http://www.scirp.org/journal/jsea
http://dx.doi.org/10.4236/jsea.2015.810050

Why Formal Methods Are Considered for Safety Critical Systems?

Monika Singh[1], Ashok Kumar Sharma[1], Ruhi Saxena[2]

[1]Faculty of Engineering & Technology (FET), Mody University of Science & Technology, Sikar, India
[2]Computer Science & Engineering, Thapar University, Patiala, India
Email: Dhariwal.monika@gmail.com

Received 20 March 2015; accepted 13 October 2015; published 16 October 2015

Abstract

Formal methods are the mathematically techniques and tools which are used at early stages of software development lifecycle processes. The utter need of using formal methods in safety critical system leads to accuracy, consistency and correctness in proposed system. In safety critical real time application, requirements should be unambiguous and very accurate which can be achieved by using mathematical theorems. There is utter need to focus on the requirement phase which is the most critical phase of SDLC. This paper focuses on the use of Z notation for incorporating the accuracy, consistency, and eliminates ambiguity in safety critical system: Road Traffic Management System as a case study. The syntax, semantics, type checking and domain checking are further verified by using Z/EVES: a Z notation type checker tool.

Keywords

Formal Methods, Safety Critical System, Z Notation, Z/EVES, Syntax & Type Checking, Domain Checking

1. Introduction

Formal specification languages are mathematically based on languages which are adequately used for construction of accurate, consistent and unambiguous systems and software. As formal methods are equipped with tool, which can be used for both the prospective *i.e.* describing a system and later on for analyzing their functionalities. The major obstacles behind formal methods to be used in practices frequently are the time spent on specification [1] [2]. Nevertheless, formal methods do not guarantee correctness, but their use emphasize to increase the understanding of a system by divulging errors or facets of incompleteness that may be expensive to correct them at any later point of time. However, formal methods play a critical role in safety critical system as they fo-

How to cite this paper: Singh, M., Sharma, A.K. and Saxena, R. (2015) Why Formal Methods Are Considered for Safety Critical Systems? *Journal of Software Engineering and Applications*, **8**, 531-538. http://dx.doi.org/10.4236/jsea.2015.810050

cus on refinement of requirements in the early stage of development which consequently increase the system's accuracy and consistency. Various formal languages are used for this purpose like VDM, B-Methods, Petri Net, and Z notation etc. Z notation is a model based on formal specification language which uses the set theory and first order predicates [3].

A lot of work has been done in this area of formal analysis of UML diagrams with formal approaches [4]-[8]. In article 8, UML based framework is presented to develop web applications. [5] represents the verification properties by HOL theorem prover. A formalization approach is developed for UML class diagrams in [6]. The paper [7] advocates how the formal methods can be used for safety properties of real time critical application such as railways. [8] explains an integrated approach of Z notation and Pertinet for analysis of safety critical properties.

In this article, Z notation is used for formal analysis of safety critical system *i.e.* Road Traffic Management System which is further verified by using the Z/EVES tool.

2. Proposed Approach & Methodology

In the first part of this section, the proposed approach is discussed. Then the tool and methodology used are discussed in section.

2.1. Proposed Approach

Figure 1 defines the proposed approach for designing the safety critical system using the formal methods.

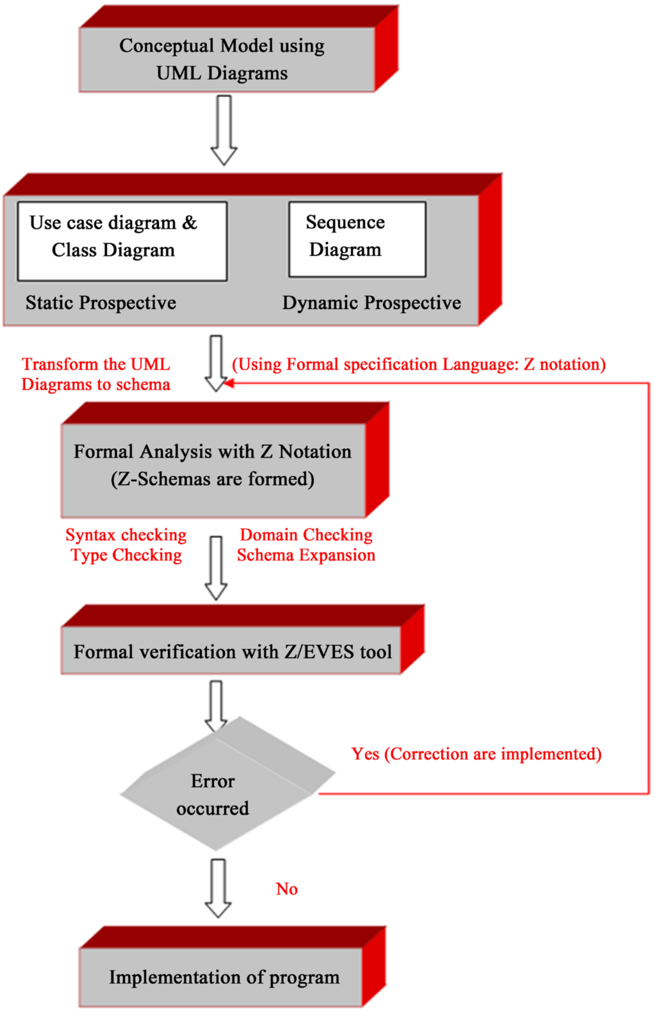

Conceptual Model using UML Diagrams
Use case diagram & Class Diagram
Sequence Diagram
Static Prospective
Dynamic Prospective
Transform the UML Diagrams to schema
(Using Formal specification Language: Z notation)
Formal Analysis with Z Notation (Z-Schemas are formed)
Syntax checking Type Checking
Domain Checking Schema Expansion
Formal verification with Z/EVES tool
Error occurred
Yes (Correction are implemented)
No
Implementation of program

Figure 1. The proposed approach for formal analysis of safety critical application.

2.2. Z/EVES

This tool is used for verifying the specification written in Z notation language. This verification includes syntax, semantics, type checking, and domain checking of the given system's specification. Z/EVES present two type of interface: graphical user interface and the command line interface [3] [9]. In this paper, we used the graphical user interface for verifying and composing the specification which were written in Z notation language. Moreover, Z/EVES propose two mode of operations *i.e.* "Eager" and "Lazy". In our article we use the "Eager" mode since in this mode a paragraph is checked if and only if all the previous ones are checked which is highly recommended for safety critical real time application. By using Z/EVES, following can be done:

- syntax and type checking;
- schema expansion;
- precondition calculation;
- domain checking;
- general theorem proving.

2.3. UML

Unified Modeling language is in fact the blue prints for the system to be developed. It provides a better way to understands the requirements of the propose system. UML consists of nine diagrams which are used for capturing the both aspects of the system *i.e.* static and dynamic [10]-[12]. This paper aims at the static behaviour by composing the use case diagram of RTMS system which is further verified by using Z/EVES type checker tool. The conceptual model of Road Traffic Management System (RTMS) is given in **Figure 2**.

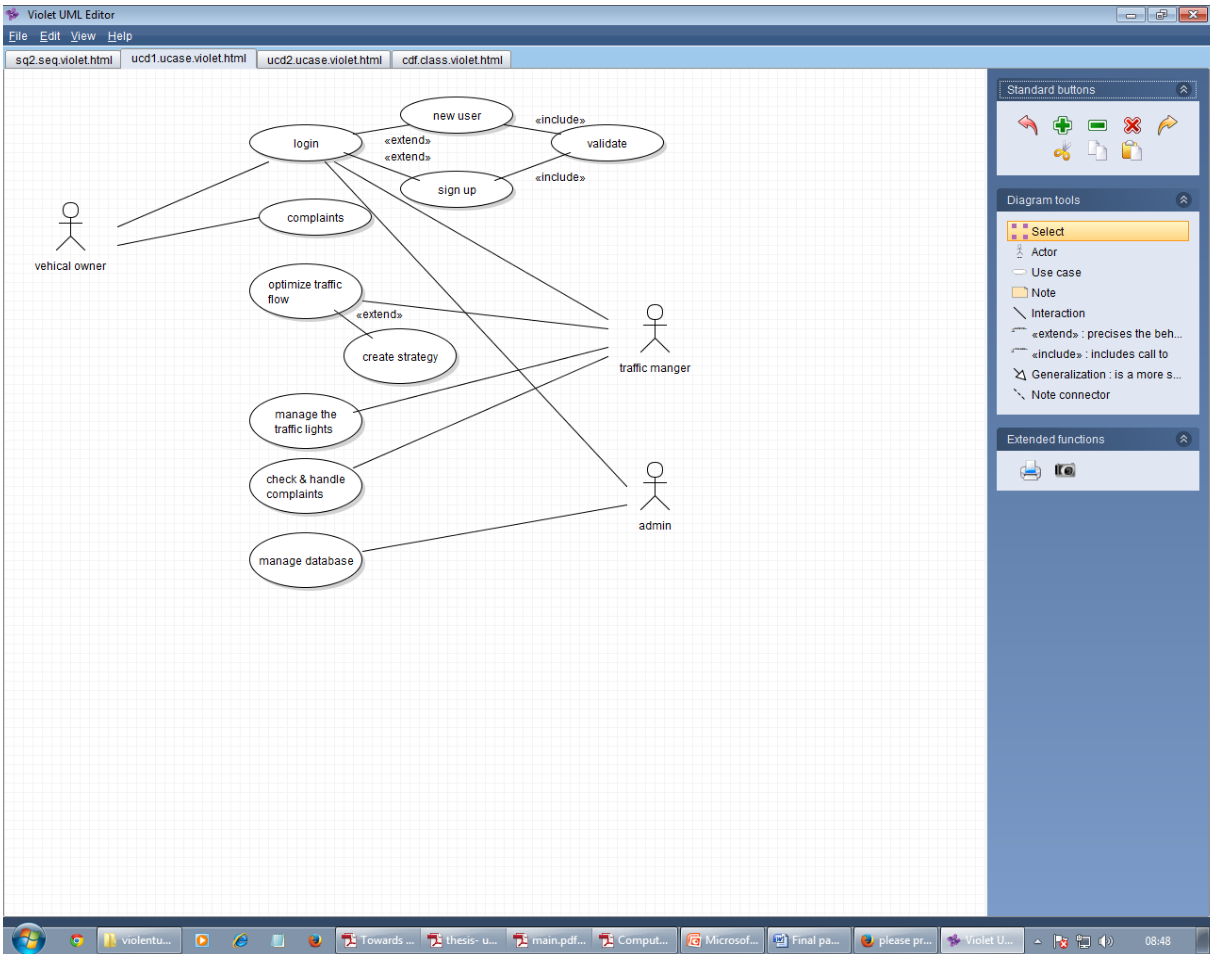


Figure 2. Use case diagrams of vehicle owner.

3. Formalization of Use Case Diagram Using Z/EVES

Z schema is the notion for structuring the specification including the pre, post condition and the list of invariant & variables. Z schema has two parts *i.e.* declaration part and predicate part. The Z schema has both declaration as well as predicate part that is shown in **Figure 3**.

The above part of central line consists of variables declaration and the below part of line describes the relationship the variable's various values. This paper emphasis on three main characteristics of formal analysis of safety critical system which are:

1) Syntax & Type checking; 2) Schema Expansion; and 3) Domain checking.

1) Syntax & Type Checking

The syntax and type checking facility is provided by the Z/EVES tool. The syntax & type checking facility enables that the syntax used in Z specification is correct which is automatically done by Z/EVES tool. In case of road traffic management system, the schema of Vehicle Owner is considered for syntax & type checking which is consists of two variables:

- *Vowner* is the set of names with RTMS registered.
- *Regist Vowner* is the function which when implemented on a particular Vehicle Owner name, provides the unique registration number associated with the person.

In **Figure 4**, the schema for Vehicle Owner with basic data type is given: [Name, Seqchar].

In Vehicle Owner schema, a partial function named "*Regist Vowner*" is defined which maps the corresponding vehicle owner with a registration number *i.e.*

Regist Vowner*: *Name*⇸*Seqchar

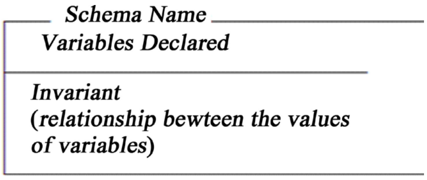
Moreover, "*Regist Vowner*" is a one-to-one function which maps Vehicle Owner name with registration number. Since it is a one-to-one function, therefore every Vehicle Owner has a unique registration number and consequently, would be no ambiguity. The schema of Vehicle Owner is further verified by Z/EVES tool for syntax & type checking in **Figure 5**. The left most columns' value "Y" shows that the schema is implemented using correct syntax. If there would be any syntax error, it shows "N" instead of "Y" in syntax column [9].

2) Schema Expansion

The schema expansion facility enables to extend the functionality of system and helps in understanding the complex schema structure in detail. Initially, the list of registered vehicle owner in RTMS is empty which is depicted by the "Init Vehicle Owner" schema in **Figure 6**.

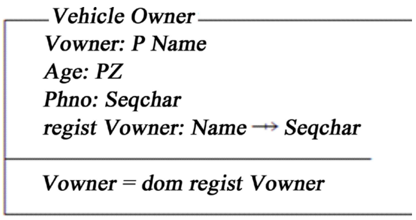
Since the lower part of the schema explain the relation between the variables, the function *Regist Vowner* is assigned a value "*ϕ*", and means initially there is no registered vehicle owner in RTMS. **Figure 7** shows the Z/EVES result of "Init Vehicle Owner".

Now, the Vehicle Owner may perform a list of tasks like: Login. If the Vehicle Owner is Login first time, he/she has to register him/her; otherwise he/she will sign in. In **Figure 8**, the schemas of Login operation is implemented.



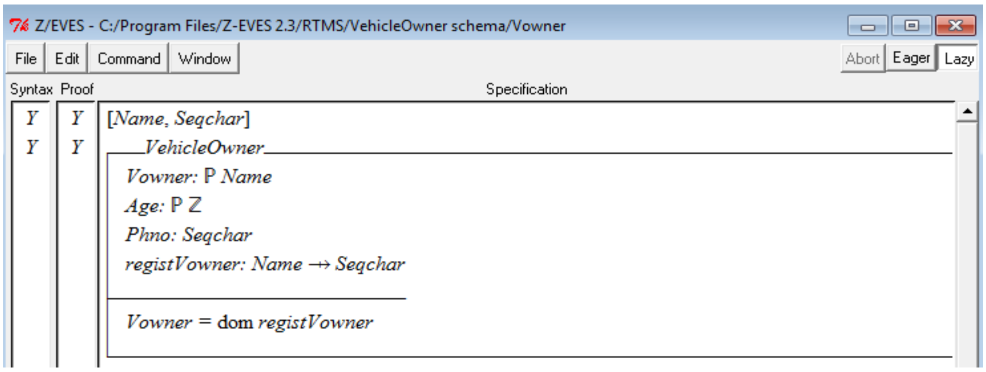
Schema Name
Variables Declared
Invariant
(*relationship bewteen the values of variables*)

Figure 3. State space of schema.



Vehicle Owner
Vowner: P Name
Age: PZ
Phno: Seqchar
regist Vowner: Name ⇸ Seqchar
Vowner = dom regist Vowner

Figure 4. Vehicle Owner schema with invariants.


Z/EVES - C:/Program Files/Z-EVES 2.3/RTMS/VehicleOwner schema/Vowner
File
Edit
Command
Window
Abort
Eager
Lazy
Syntax
Proof
Specification
Y
Y
Y
Y

$[Name, Seqchar]$

VehicleOwner

$Vowner: \mathbb{P}\ Name$

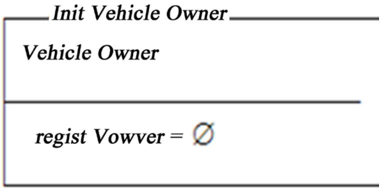
$Age: \mathbb{P}\ \mathbb{Z}$

$Phno: Seqchar$

$registVowner: Name \rightarrowtail Seqchar$

$Vowner = \mathrm{dom}\ registVowner$

Figure 5. Syntax checking of Vehicle Owner schema by Z/EVES.

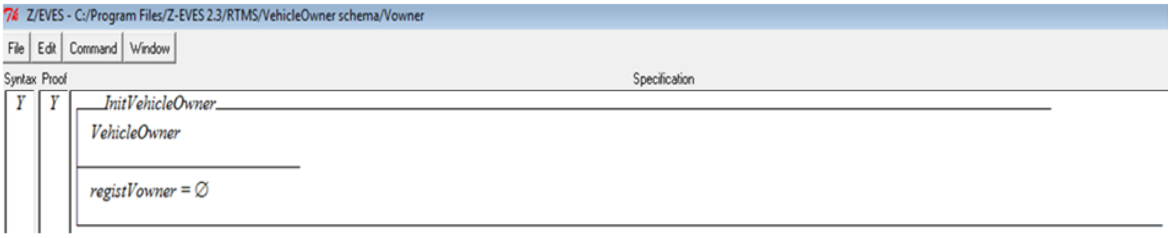


Init Vehicle Owner

Vehicle Owner

regist Vowver = ∅

Figure 6. Initial state space of schema Vehicle Owner.

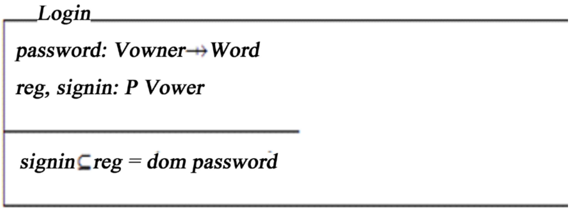

Z/EVES - C:/Program Files/Z-EVES 2.3/RTMS/VehicleOwner schema/Vowner
File
Edit
Command
Window
Syntax
Proof
Specification
Y
Y

InitVehicleOwner

VehicleOwner

$registVowner = \varnothing$

Figure 7. Initial Vehicle Owner schema.



Login

password: Vowner⇸Word

reg, signin: P Vower

signin⊆reg = dom password

Figure 8. State space of schema Login.

In this schema:

Password**:** ***Vowner⤔Word***

"Password" is a function which associates a username to password. Nevertheless, it is a one-to-one function which in turn provides accuracy and correctness to system. Now Signin set and registered set both is the member of power set of Vehicle Owner which is mathematically shown by using set theory as following.

Signin, Reg**:** ***ℙ Vowner***

Also the Signin set is a subset of registered set and the registered set having the values which are there in domain of "password" function *i.e.*

Signin ⊆ ***Reg*** **=** ***Dom Password***

Initially, Login schema is empty which is here explained by assigning a value "ϕ" to both the set whether it's a registers one or a new one *i.e.*

Reg **=** ***ϕ*****;** ***Signin*** **=** ***ϕ***

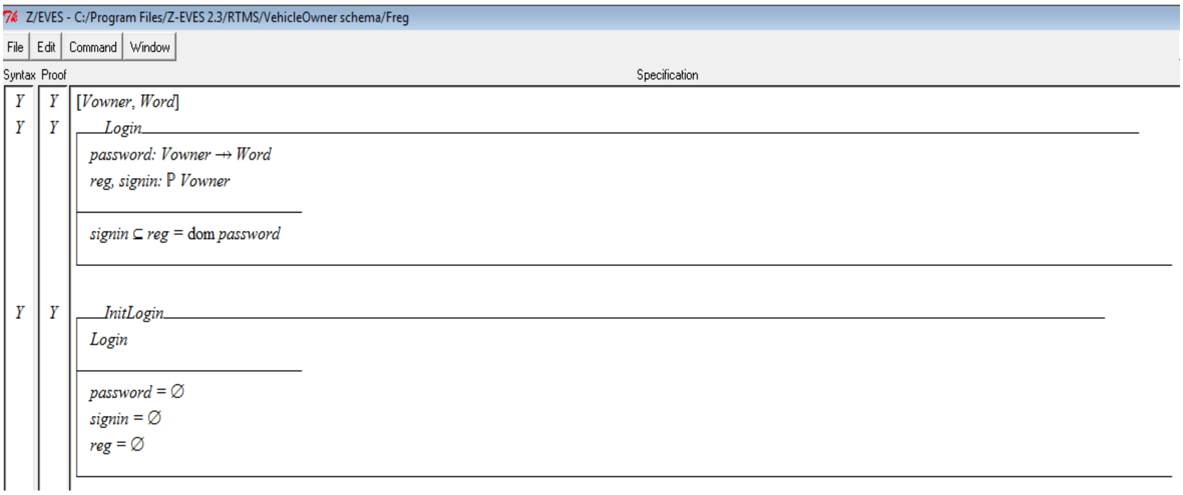
This is called schema expansion which is one of the key features of Z/EVES tool *i.e.* from "Init Login" schema to "Login" schema.

In **Figure 9**, the schema expansion is shown and verified by Z/EVES as follow.

3) Domain Checking

Domain checking feature of Z/EVES tool enables us to write the statements which are meaningful and in finding the domain errors. However, it has been found that as compared to syntax & type checking, domain checking is more crucial because where syntax and type checking is done automatically, one needs to work together with theorem prover to accomplish the domain checking. We also observed that proof "by reduce" in the proof window of the tool was sufficient for our formal specifications for domain checking. Now if you are already registered, you will opt for the sigin option. By investigating **Figure 10**, the value for syntax column is "Y", means no error, but the value in proof column is "N". This is related to domain checking. The proof can be initiated by selecting the theorem in the ***Specification window***, right clicking, and selecting "***Show proof***" which is shown in **Figure 9**.

The proof can be done by various mean in Z/EVES by choosing "Action Point" by Reduction, Cases, Quantifiers, Normal Norms and Equality. In our case, we use the option "**prove by reduction**". **Figure 11** describes the proof by reduce action point in case of "Signin" schema.


Z/EVES - C:/Program Files/Z-EVES 2.3/RTMS/VehicleOwner schema/Freg
File
Edit
Command
Window
Syntax
Proof
Specification

Y Y [Vowner, Word]

Y Y Login

password: Vowner ⇸ Word
reg, signin: ℙ Vowner

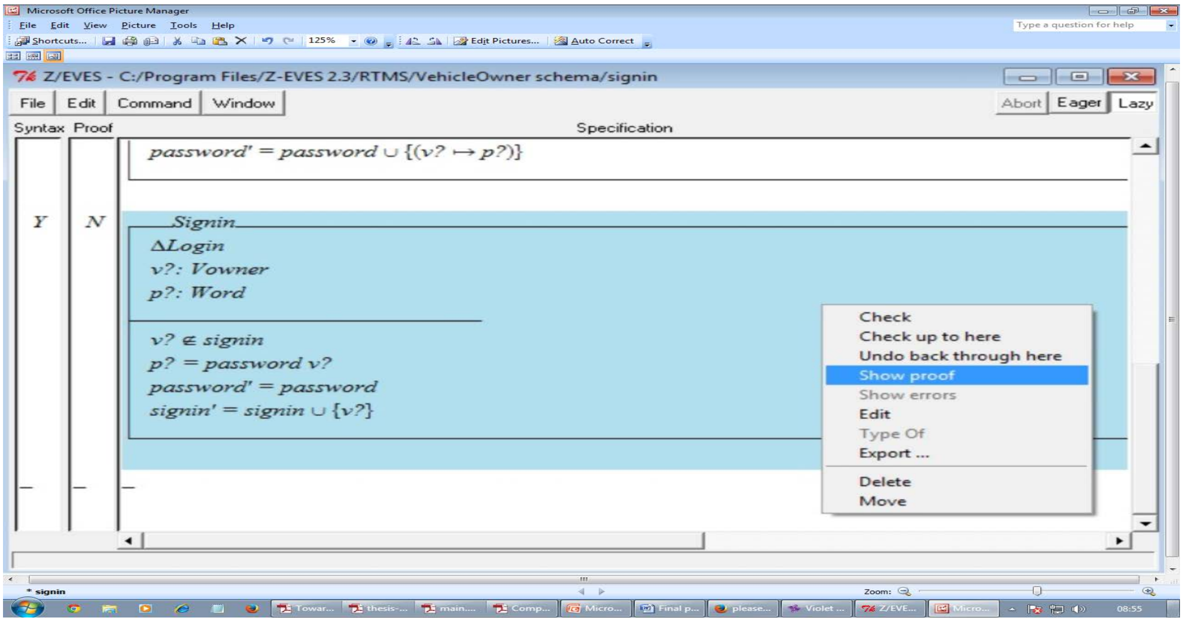
signin ⊆ reg = dom password

Y Y InitLogin

Login

password = ∅
signin = ∅
reg = ∅

Figure 9. Z/EVES Schema expansion of Initial Login to Login schema.



password′ = password ∪ {(v? ↦ p?)}

Y N Signin

ΔLogin
v?: Vowner
p?: Word

v? ∉ signin
p? = password v?
password′ = password
signin′ = signin ∪ {v?}

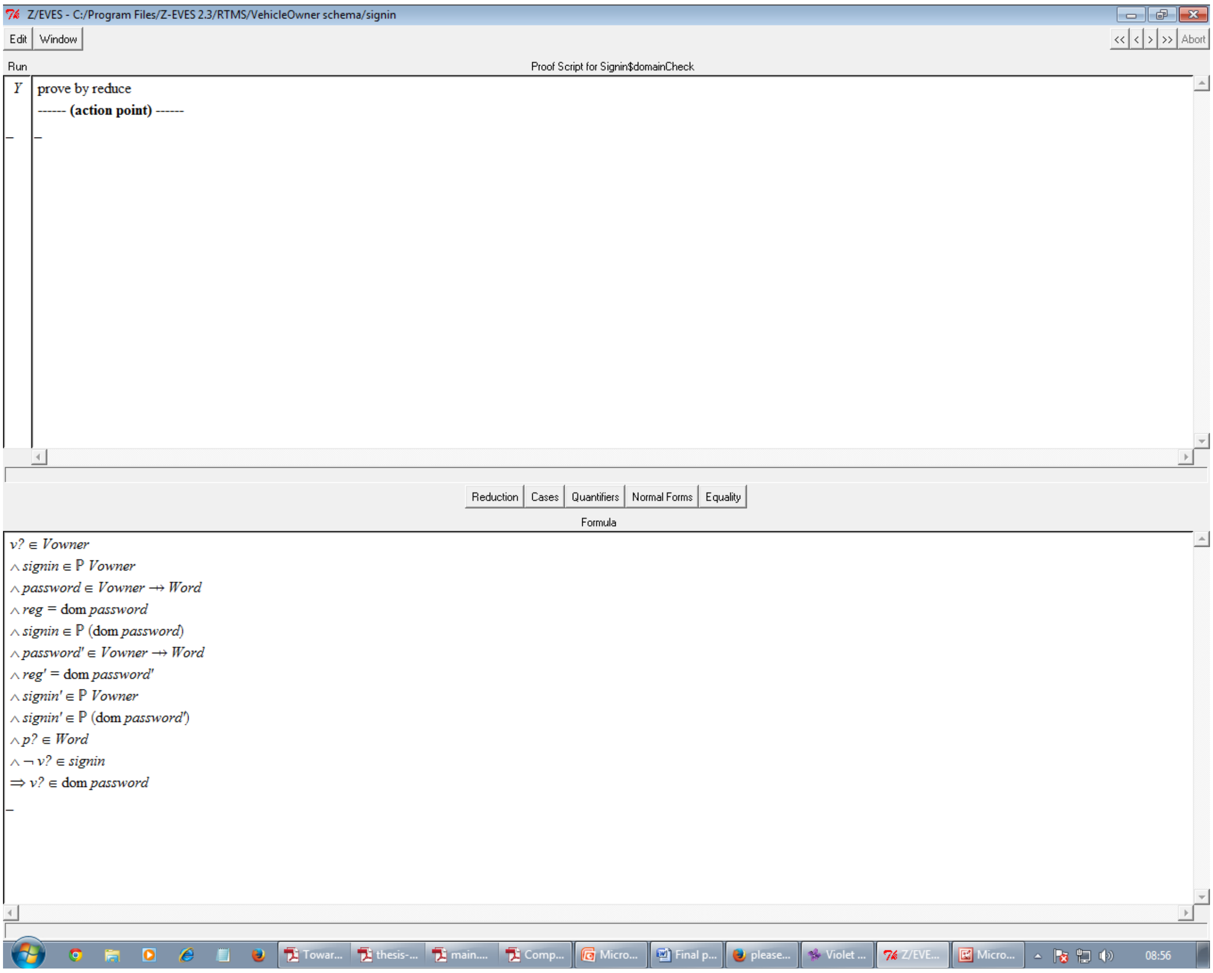
Figure 10. Domain checking with Z/EVES.

4. Result Analysis

Any proposed model is incomplete without tool support. Nevertheless, use of formal language adequately increases the accuracy and completeness but, the use of computer tool indeed increases the level of confidence significantly for the system to be developed by fingering out the potential errors in syntax and semantics of formal narration. **Table 1** depicts the result of formal analysis of proposed schemas of road traffic management system using Z/EVES. The attributes in the table are name of the schema followed by syntax & type checking, domain checking, proof and reduction. The second row in table, having status Y for all columns indicating that the schema named "Vehicle Owner" is correct with respect to syntax & type check errors, domain check and having correct proof by performing reduction on the set of predicates for making specification meaningful. The Y¹ symbol shows that the action point in proof window is chosen as "**prove by reduce**".

5. Conclusion

The use of formal methods in safety critical application increases quality in terms of accuracy, consistency, and



Y prove by reduce

------ **(action point)** ------

$v? \in Vowner$
$\wedge\ signin \in \mathbb{P}\ Vowner$
$\wedge\ password \in Vowner \nrightarrow Word$
$\wedge\ reg = \mathrm{dom}\ password$
$\wedge\ signin \in \mathbb{P}\ (\mathrm{dom}\ password)$
$\wedge\ password' \in Vowner \nrightarrow Word$
$\wedge\ reg' = \mathrm{dom}\ password'$
$\wedge\ signin' \in \mathbb{P}\ Vowner$
$\wedge\ signin' \in \mathbb{P}\ (\mathrm{dom}\ password')$
$\wedge\ p? \in Word$
$\wedge\ \neg\ v? \in signin$
$\Rightarrow v? \in \mathrm{dom}\ password$

Figure 11. Proof script by using action point "**proof by reduce**" for "Signin" schema.

Table 1. Result analysis by Z/EVES.

Schema Name	Syntax & Type Checking	Domain Checking	Schema Expansion	Proof by Reduction
Vehicle Owner	Y	Y	Y	Y¹
Login	Y	Y	Y	Y¹
Signin	Y	Y	Y	Y¹

in completeness. This paper describes the use of Z notation, a formal method for Vehicle Owner, an actor of Road Traffic Management System; which will be further verified by Z/EVES, a typechecker tool for Z notation specification. In Future, the schema of Traffic Police, Admin, and Traffic Manager will be implemented and verified by Z/EVES theorem prover.

Acknowledgements

Authors are thankful to faculty of Engineering & Technology (FET), Mody University of Science & Technology for providing the facility to carry out the research work.

References

[1] Woodcock, J.C.P. (1989) Structuring Specifications in Z. *IEE/BCS Software Engineering Journal*, **4**, 51-66. http://dx.doi.org/10.1049/sej.1989.0007

[2] Hall, A. (2002) Correctness by Construction: Integrating Formality into a Commercial Development Process. *Proceedings of International Symposium of Formal Methods Europe*, **2391**, 139-157. http://dx.doi.org/10.1007/3-540-45614-7_13

[3] Spivey, J.M. (1989) The Z Notation: A Reference Manual. Prentice-Hall, Englewood Cliffs.

[4] Hamdy, K.E., Elsoud, M.A. and El-Halawany, A.M. (2011) UML-Based Web Engineering Framework for Modeling Web Application. *Journal of Software Engineering*, **5**, 49-63. http://dx.doi.org/10.3923/jse.2011.49.63

[5] Hasan, O. and Tahar, S. (2007) Verification of Probabilistic Properties in the HOL Theorem Prover. *Proceedings of the Integrated Formal Methods*, **4591**, 333-352. http://dx.doi.org/10.1007/978-3-540-73210-5_18

[6] He, X. (2000) Formalizing UML Class Diagrams: A Hierarchical Predicate Transition Net Approach. *Proceedings of* 24*th Annual International Computer Software and Applications Conference*, Taipei, 25-28 October 2000, 217-222.

[7] Zafar, N.A., Khan, S.A. and Araki, K. (2012) Towards the Safety Properties of Moving Block Railway Interlocking System. *International Journal of Innovative Computing*, *Information and Control* (*ICIC International*), 5677-5690.

[8] Heiner, M. and Heisel, M. (1999) Modeling Safety Critical Systems with Z and Petri-Nets. *Proceedings of International Conference on Computer Safety*, *Reliability and Security*, London, 26-28 October 1999, 361-374. http://dx.doi.org/10.1007/3-540-48249-0_31

[9] The Z/EVES 2.0 User's Guide: Mark Saaltink. October 1999 ORA Canada.

[10] Mostafa, A.M., Manal, A.I., Hatem, E.B. and Saad, E.M. (2007) Toward a Formalization of UML2.0 Meta-Model Using Z Specifications. *Proceedings of* 8*th ACIS International Conference on Software Engineering*, *Artificial Intelligence*, *Networking and Parallel*/*Distributed Computing*, **3**, 694-701. http://dx.doi.org/10.1109/SNPD.2007.508

[11] Jacobson, R.I. and Booch, G. (2006) The Unified Modeling Language Reference Manual. 2nd Edition.

[12] Selic, B. and Rumbaugh, J. (1998) UML for Modeling Complex Real-Time Systems. Technical Report, Object Time.