Automata Based Railway Gate Control System at Level Crossing

Aniqa Rehman
Department of Computer Science
COMSATS University Islamabad
Sahiwal Campus, Pakistan
Email: aniqarehman@cuisahiwal.edu.pk

Saba Latif
Department of Computer Science
COMSATS University Islamabad
Sahiwal Campus, Pakistan
Email: sabalatif@cuisahiwal.edu.pk

Nazir Ahmad Zafar
Department of Computer Science
COMSATS University Islamabad
Sahiwal Campus, Pakistan
Email: nazafar@cuisahiwal.edu.pk

Abstract*— In recent studies, it is indicated that the rate of accidents at level crossings of Railway Gate Control System (RGCS) is increasing. The need of an automatic system emerges due to the poor management and negligence of railway staff, that may make us able to handle the situation by reducing the accidents. RGCS has gained much attention because of its accident reducing and time saving competence at the level crossing. To signify the entire automatic railway gate control system, one of the best approaches for RGCS modeling is Deterministic Finite Automata (DFA). This paper presents UML and formal model of RGCS using DFA. The behavior of the proposed model is explained using UML activity and sequence diagrams and then to prove its verification and correctness formal specification is described. Vienna Development Method Specification Language (VDM-SL), a formal specification language, is a mathematical based approach to implement the systems in a correct way and to verify the properties of software systems. For analysis purpose of the model, various functionalities of VDM-SL toolbox are used. The verification and validation of model is provided using VDM-SL toolbox.

Keywords—Railway Interlocking, Level Crossing, DFA, Formal Methods, VDM-SL.

I. INTRODUCTION

The safety critical systems are those systems whose malfunctioning may cause serious problems, e.g., loss of human lives and degradation of environment [1]. Many peoples are injured and disabled due to mishaps or negligence of railway staff. A lot of time is wasted during manual control of level crossing [2]. Manual RGCS has all of these problems described above. An automated RGCS provided a safe and secure way to reduce all accidents, injuries, mishaps and reduce time of passing the crossing efficiently. Therefore, manual systems are being replaced by automated systems to achieve more reliable and secure systems that may make us able to cope with the situation by reducing the railway accidents [3].

Modeling of systems provides an understanding and information about the system more clearly and depicting its behavior. It helps the users to understand visual flows of transactions from one state to another [4]. A more powerful tool in modeling the control flows is Deterministic Finite Automata (DFA). DFA has many applications in modelling of real time scenarios [5] including smart parking, sewerage, lightening, registration, campus, transport and other smart components. The execution time of these models may be different in different scenarios [6]. We have modeled a system in automata theory that must preserve the semantics of the system during the transformation from modeling to specification. The Integration of DFA with formal methods enhances its modeling power for complex and critical systems [7].

Ensuring the correctness of a system is more important than defining a system; this takes paramount importance when we verify safety-critical systems [8]. In these systems, incorrect functionality may cause many disasters and uncertain situations. Formal methods provide mathematical based statements to verify the correctness and validation of a system. It describes what a system should do and how to do. It provides design level testing so chances of occurrence of errors are reduced. In complex systems, formal methods provide a way to enhance the confidence, proof of correctness, validation and verification of the system [9].

The railway gate control system is assumed on the basis of microcontroller processing as shown in Figure 1. It takes data from sensors, which is then sent to the microcontroller that drives motor which opens or closes the gate. If the sensors send signal to close the gate the signal passes to the controller that estimates distance if the train enters in the critical region. Then gates are closed and parallel activities generating buzzer are performed. It also gives the light indication and drives the motor that closes the gate. When train leaves the critical region, then signal is sent to open the gate.

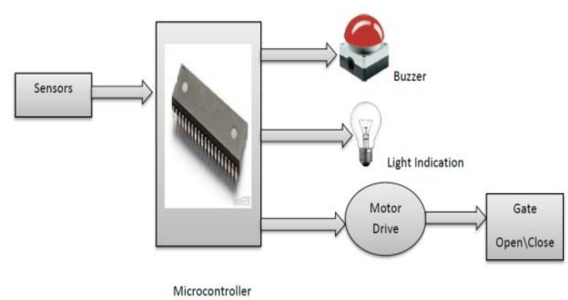

Fig.1: Visual representation of gate controller

The rest of the paper is structured as: In section II, Related work is discussed and in section III, UML based graphical model of railway gate control system is presented. Section IV describes the Automata based model of railway gate control system. Section V describes formal specification of the DFA based model. Formal analysis of the gate control is provided

978-1-5386-5106-3/19/$31.00 ©2019 IEEE

through VDM-SL in section VI. Finally, conclusion and future work are given in section VII.

II. RELATED WORK

Past studies show that different techniques have been applied in a railway safety system to avoid the accidents that occurs at level crossings. In [10-11], authors describes model that is based on cellular automata in which every railway line is considered as a cell. A list of rules describes about train speed limit, signal and train updates. The block section rules provide simulation of its work. The simulation is not a standardized approach and is a bit difficult task to interpret and validate the simulation result. In [12], authors have described failure propagation and state machine model to show the flow of information between different components of system. They have used some transformation notations which are not enough for expressing a system state statically and dynamically. The work carried in [13], author has described UML-based modeling of critical railway control system that is not enough because UML diagrams contains ambiguity. In another work [14], authors have presented petri-net model that is based on graph theoretic concepts. Due to the state space explosion, analysis of system may cause difficulty by applying petri-net model to the real level crossing. In [15], authors have provided formal verification using Z notation. Another work is listed in [16], graph theory is used to depict static model of components of the system. The authors have described an automatic railway gate control system, in which data from sensors passes to microcontroller that drives motor to open or close the gate. To analyse system's behavior, some mathematical techniques such as formal methods are used. These mathematical techniques contains some notations that are used for describing system effectively [17].

Our work is different from others because the approaches like graph theory and Z are used to describe system in abstract way. We have formalized RGCS using VDM-SL that provides a sequential way for developing a system as compared to the existing models and notations. For developing and describing the system in a sequential manner, VDM-SL is best formal method language which has mathematical foundation to implement the system in a correct way. For validation and verification of proposed solution, various existing facilities of VDM-SL toolbox are used [18]. It resolves ambiguity, provides design level testing, high- and low-level abstractions of the system [19-20].Other related work which is relevant to applications of UML, automata, graph theory and formal methods are discussed in [20- 36].

III. UML BASED GRAPHICAL MODEL OF RAILWAY GATE CONTROL SYSTEM

Modeling presents the visual analysis of the system and depicts the system view in multiple perspectives. Unified Modeling Language (UML) is an object-oriented language for demonstrating a system. UML provides the unique graphical notations which depicts the system at abstract level. It captures the system structural and behavioural properties. Sequence and Activity diagram is used to capture the behavior of the proposed system. Activity diagram depicts the flow of system in sequential or parallel manner. A method or functions can be an activity. Here, Figure 2 represents the activity diagram of the railway gate control system.

First of all, Sensors receive the input and send to the controller, then controller checks the distance of the train from destination. If the distance is less than and equal to the defined parameter then it will perform some parallel actions i.e., Light indication, generate buzzer and send command to the motor to close gate. If the distance is greater than the define parameter, then it will open the gate. A detailed interaction of the proposed system is defined through sequence diagram. Interaction is represented through message passing between objects of the system. Figure 3 represents the sequence diagram in detail. Some guard condition is mentioned in sequence diagram which represents that if the desired distance is less than the defined parameter then system will close the gate automatically, generate the buzzer for indication to the passengers and send some light indications as well. Otherwise the gate will remain open.

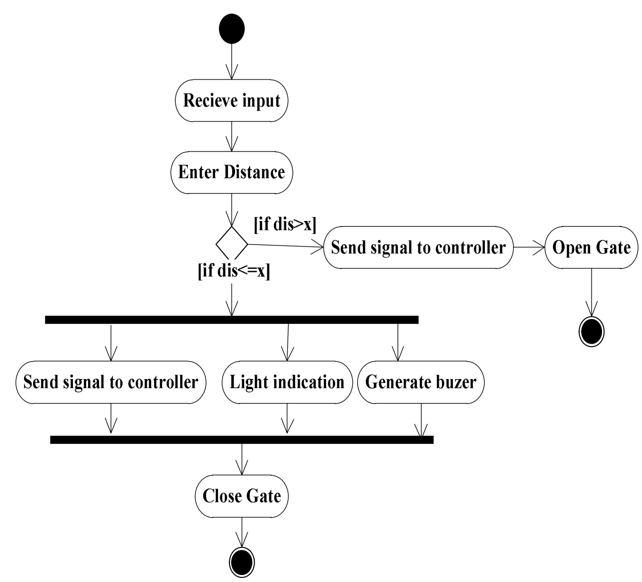

Fig. 2: Activity diagram of railway gate control system

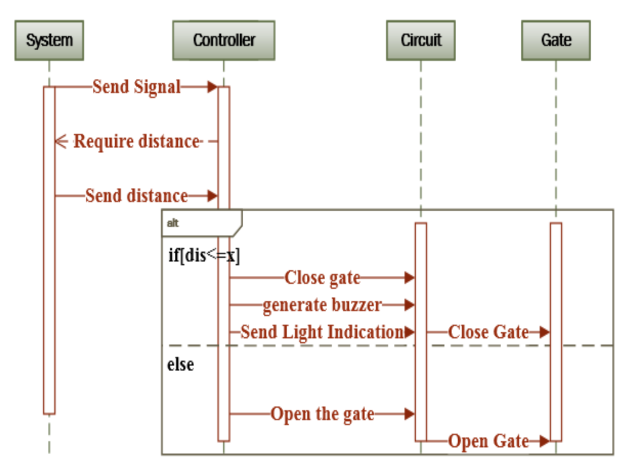

Fig. 3: Sequence diagram of railway gate control system

IV. AUTOMATA BASED MODEL OF RAILWAY GATE CONTROL SYSTEM

Automata theory deals with the mathematical properties of the system. Formal analysis of the system is presented by theory of automata model called Deterministic Finite Automat (DFA). It captures the system dynamic behavior in an efficient manner.
A Deterministic Finite Automata (DFA) consists of five components (Q, Σ, δ, q0, F) [15-16]. Description of each component is given below.

- i) Finite non-empty set of states is represented by Q.
- ii) Finite Set of alphabets is represented byΣ.
- iii) δ is transition function which takes inputs as one alphabet and one state and outputs the next state.
  $\delta : Q \times \Sigma \rightarrow Q$
- iv) q_0 is the initial state of the system.
- v) F is a finite set of final state.

For every state $q \in Q$ and for each alphabet $a \in \Sigma$, the transition function can be defined as:

$$\delta(q1, a) = q2,$$

Where q1 and a are inputs and q2 is the output. It is noted that q1 and q2 might be same.
States: Finite set of states Q = {Sen, Con, Cir, Dis, Cl, OP, Buz, Lig, Dead } i.e., Sensor, Controller, Circuit, Distance, Close, Open, Buzzer, Light and Dead state.
Alphabets: Finite set of alphabets Σ = {s, c, d, l, g, b, cl} i.e., sensor, check, distance, less, greater, buzzer and close.
Initial State: S
Final State: Finite set of final state. F = {Cl, OP}
Transition Function: Transition function that takes input as one alphabet and one state and outputs the next state.

$$\delta: Q \times \Sigma \rightarrow Q$$

One of the functions of automated Railway Gate Control System is given below.
It is described that real system is mapped into DFA, i.e., open the gate when microcontroller receive signal from sensors.

$$\delta: Sen \times s \rightarrow Con$$
$$\delta: Con \times c \rightarrow Dis$$
$$\delta: Dis \times d \rightarrow Cir$$
$$\delta: Cir \times g \rightarrow OP$$
$$\delta: Cir \times l \rightarrow Lig$$
$$\delta: Lig \times b \rightarrow Buz$$
$$\delta: Buz \times cl \rightarrow Cl$$
$$\delta: Cl \times cl \rightarrow Cl$$

Similarly, all functions are mapped that are shown in the Figure 4.

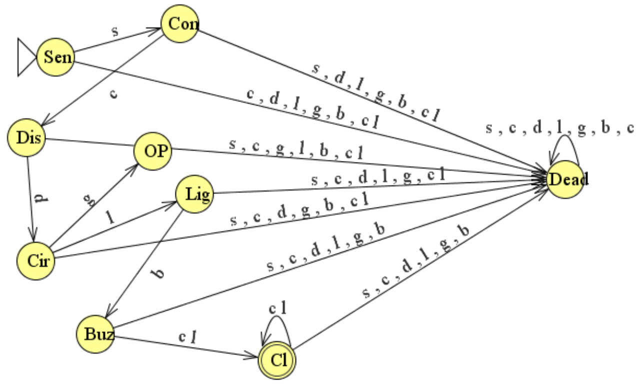

Fig. 4: Automata based model of railway gate control system

V. FORMAL SPECIFICATION USING VDM-SL

In this Section, the transformation of DFA model into VDM-SL is described. Firstly, DFA model is mapped into VDM-SL. VDM-SL specification describes definition, state, invariants, functions and operations of the proposed model. The types of system variables are defined in static model, we have used composite data type which is similar to structure data type in the programming languages for recording different data types into one variable.

```
types
Id = token;
States = seq of Id;
Signal = <s>|<c>|<d>|<g>|<l>|<b>|<cl>;
Signal1= seq of Signal;
Status = <open>|<close>;
Condition = <occupy>|<clear>;

ARS::
        Id:Id
        Sig:Signal;
```

The Id is defined as token data type, states variable is defined as sequence of Id in which there is no repetition of values and are stored in a specified manner. Signal have different values like s, c, d, l, g, b and cl. Signal1 is defined as sequence of signal. The status variable has two possible types that is open and close. ARS variable is composite type variable having many different types of variables which are considered as properties of one single variable. The composite type variables in VDM-SL are accessed using an object variable which is created and accessed through dot operator. ARS variable have Id and sig variables.

Another composite type variable ARGS is defined in which there are many variables are associated with each other. States, signal, initial_state, trans_fun and final_state. There are many invariants which are defined on the function. Invariants are the properties which must satisfied before the execution of the program. The invariant scope is dependent on the function or operation being defined. Invariants are the conditions that must be true otherwise system does not work properly. In the invariants, it is stated that initial state must belong to set of states of this system. Every element in set of final state is also in the set of total states that is final_state ⊆ states.

```
ARGS::
  states : set of ARS
  signal : set of Signal
  intial_state : ARS
  trans_fun : map (ARS * Signal) to ARS
  final_state : set of ARS
inv args == args.intial_state in set args.states and
forall a in set args.final_state &
  a in set args.states and
forall mk_(s1,signal) in set dom args.Trans_fun &
  exists s2 in set args.states &
  args.Trans_fun(mk_(s1,signal)) = s2;
```

In the above specification, Signal, Signal1, Status and Condition are user defined data types which are specified according to the nature of the system. ARS and ARGS are composite types that contain combination of different basic data types.

In this section of formal specification, the functions of the system are defined. Functions in VDM-SL are used to perform some action based on the conditions defined on the system. String acceptor is a function which is defined having input of signal and sequence of ARS. The output of the system is Boolean which returns true or false after the execution of that function.

```
functions
String_Acceptor : ARGS *Signal1 * set of (seq of ARS) -> bool
String_Acceptor(args,signal1,ars) ==
        forall signal in set elems signal1 &   signal in set args.signal and exists states in set ars &
  len states = len signal1 + 1 and
  states(1) = args.intial_state and
  states(len states) in set args.final_state and
forall i in set {2,...., len states} &
  args.Trans_fun(mk_(states(i-1), signal1(i)) = states(i));
```

The nature of the function is defined as every signal must belong to the alphabets set. There must be many states in the system from which one should be initial and at least one state should be final state. Other remaining states are non-final, non-initial states which provide us way towards the final state from initial. The last state of the proposed system must belong to the set of final states.

The state of the system is used to define system variables, types, constants and composite type. In the state of the system the system variables are declared and initialized also invariants on the system are defined. State, of and end are keywords used in the VDM-SL toolbox.

```
state Railway of
tid: nat
dis: nat
sta: Status
con: Condition
inv mk_Railway(-,-,s,c) == s = <open> and c = <clear>
init mk_Railway(t,d,s,c) == t = 1 and d = 1 and s = <open> and c = <clear>
end
```

State is a permanent data type that must be stored in a system. It consists of variable with their type, invariants and initialization of all variables. The initialization of the system variables is also done in the state of the system.

```
operations
enter_critical_region(d:nat)c1: Condition
ext wr tid: nat
   rd dis: nat
   rd con: Condition
pre dis> = d and con=<clear>
post c1= <occupy>;
```

The operations in VDM-SL are used to change the state of the system. First operation in our proposed system is enter critical region. This operation senses if there is incoming of train in the critical region of the system

```
leave_critical_region()c1:Condition
ext rd tid: nat
   rd dis: nat
   rd con: Condition
pre con = <occupy>
post c1 = <clear>;
```

Another operation in our proposed system is leave critical region. This operation senses if there is outgoing of train in the critical region of the system. The con variable will be occupied and c1 will be clear.

```
open()s1: Status
ext rd tid: nat
   rd dis: nat
   rd con: Condition
pre sta = <close>
post con = <clear> => s1=<open>;
```

The open operation in VDM-SL is used to open the barried also it changes the state of the system. The value of sta system is changed, con will be in clear position and s1 will be opened.

```
close()s1:Status
ext rd tid:nat
   rd dis:nat
   rd con:Condition
pre sta = <open>
post con = <occupy> => s1 = <close>;
```

VI. RESULTS AND ANALYSIS

Formal modeling and analysis of the Railway Gate Control System (RGCS) is done through VDM-SL toolbox. The formal analysis of the system is used to validate and verify system and its properties. The formal model is constructed in VDM-SL using the guidelines available in the manual of VDM-SL [37]. The proposed system model is analyzed using facilities available in VDM-SL toolbox. The syntax check is a facility in VDM-SL which is used to check the syntax of the specification against definitions of the variables, constants, composite types available in VDM-SL toolbox. Formal analysis using VDM-SL toolbox shows that there are no syntax and type error. Type error in the toolbox is used to validate that every variable which is defined is properly and logically used in correct way in the specification. Further, there are no warnings found in the specification analysis.

Figure 4 and 5 depicts the screenshots that provides the proof of the correctness of the formal model. Figure 4 shows the project of Railway Gate Control System as loaded in the VDM-SL toolbox. In Figure 5, S shows that there is no syntax and T represents that there is no type error in the model of the Automata Based Railway Gate Control System at Level Crossing (RGCS).

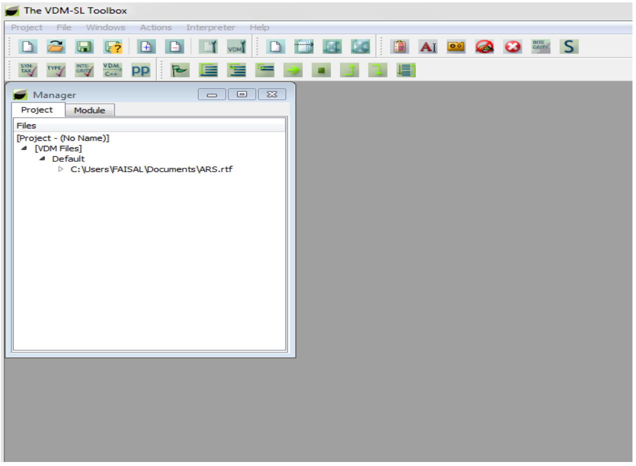

Fig. 5: Load project screenshot

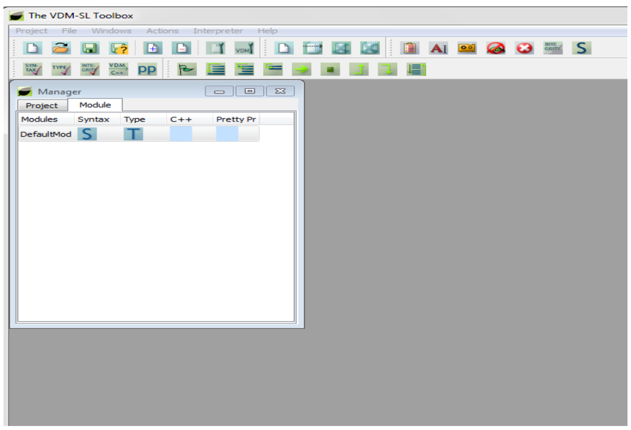

Fig. 6: Syntax and type check

VII. CONCLUSION

A comprehensive survey about safety critical systems is discussed in this paper. There exists some research work in this area, but we have proposed a different model of Railway Gate Control System (RGCS) using integration of different approaches. Firstly, the model is designed using UML activity and sequence diagrams for capturing the functionality of the system and to depict system behavior. After UML sequence diagram is transferred into Deterministic Finite Automata (DFA) realizing the functions of the system. Further behavior of the system is presented using DFA supporting automation of the system. Finally, automata-based model is transferred into formal model using VDM-SL which is an ideal language for specification of a system. At the end, it is concluded that this research is beneficial for industrial and academic purposes as paper focus was on applications of basic concepts and principles. In future, we would extend this work by using Specification Description Language (SDL) that support object-oriented design of the system.

ACKNOWLEDGMENT

We would like to thank to Miss Amna Pir and Miss Madiha Fatima for improving the quality of our paper.

REFERENCES

[1] J. Banuchandar, V. Kalian, P. Balasubramanian, S. Deepa, N. Thamilarasi, “Automated Unmanned Railway Level Crossing System,” International Journal of Modern Engineering Research (IJMER), 2, pp. 458-468, 2012.
[2] N. A. Zafar, “Formal dynamic operational model of RIS components,” IJCSNS, vol 11(9), pp. 91-96, 2011.
[3] N. A Zafar, S. A. Khan, K. Araki, “Towards the safety properties of moving block railway interlocking system,” Int. J. Innovative Comput., Info & Control, vol 8(7), pp. 5677-5690, 2012.
[4] J. Rushby, “Formal Methods and their role in the certification of critical systems,” Safety and Reliability of Software Based Systems, pp. 1-42, 1997
[5] P. Antoniou, J. Holub, C. S. Iliopoulos, B. Melichar, P. Peterlongo, “Finding common motifs with gaps using finite automata,” International Conference on Implementation and Application of Automata, pp. 69-77, 2006.
[6] K. Brouwer, W. Gellerich, E. Plödereder, “Myths and facts about the efficient implementation of finite automata and lexical analysis,” International Conference on Compiler Construction, 1998.
[7] N. Sabir, A. Ali, N.A. Zafar, “Linking finite automata and formal methods enhancing modeling power for complex systems,” International Conference on Computer Science and Information Technology, ICCSIT'08, 2008.
[8] V. Hartonas-Garmhausen, S. Campos, A. Cimatti, E. Clarke, F. Giunchiglia, “Verification of a safety-critical railway interlocking system with real-time constraints,” In Fault-Tolerant Computing Digest of Papers. Twenty-Eighth Annual International Symposium, pp. 458-463, 1998.
[9] N. A. Zafar, “Modeling and formal specification of automated train control system using Z notation,” IEEE International Multitopic Conference, 2006.
[10] N. A. Zafar, “Formal specification and validation of railway network components using Z notation,” IET Software, vol 3(4): pp. 312-320, 2009.
[11] H. Zhou, L. Jia, Y. Qin, “Cellar Automaton Model for Railway Transportation Safety System,” Knowledge Acquisition and Modeling Workshop (KAM) Workshop, 2008.
[12] H. Dong, B. Ning, B. Cai, Z. Hou, “Automatic train control system development and simulation for high-speed railways,” IEEE Circuits and Systems Magazine, vol 10(2), p. 6-18.
[13] A. Pataricza, I. Majzik, G. Huszerl, G. Várnai, “UML-based design and formal analysis of a safety-critical railway control,” Software Module Safety, 2003.
[14] J. Guo, Z. Huang, M. Liu, “Research on the railway safety critical system with petri nets,” IEEE 6th international Conference on ITS Telecommunications, 2006
[15] M. Sipser, “Introduction to the Theory of Computation,” Thomson Course Technology Boston, vol 2. 2006.
[16] J. E. Hopcroft, “Introduction to the Automata Theory, Languages and Computation,” Pearson Education India, 1979.
[17] J. M. Wing, “A specifier’s introduction to the formal methods,” Computer, vol 23(9): pp. 8-22, 1990.
[18] C. Quentin, A. Kans, “Formal Software Development from VDM to Java,” Palgrave Macmillian, 2004.
[19] W. Xishi, N. Bin, C. Yinhang, “A new microprocessor-based approach to an automatic control system for railway safety,” Proceedings of the IEEE International Symposium Industrial Electronics, 1992.
[20] H. Zhengqun, C. Junxia, Z. Jie, Z. Lirong, L. Chang, “Railway Safety Monitoring System Based on CAPS,” IEEE Fifth International Conference on Measuring Technology and Mechatronics Automation, pp. 838-840, 2013.
[21] N. A. Zafar, F. Alhumaidan, “Scenarios Verification in Sequence Diagram,” The Journal of American Science, vol 9, pp. 287-293, 2013.
[22] S. A. Khan, N. A. Zafar, F. Ahmad,” Petri net modeling of railway crossing system using fuzzy brakes”, International Journal of Physical Sciences, vol 6(14), pp. 3389-3397, 2011.
[23] S. Yousaf, N. A. Zafar, S. A. Khan,” Formal analysis of departure procedure of air traffic control system”, 2nd International Conference on In Software Technology and Engineering (ICSTE), vol. 2, pp. V2-301, 2010

[24] M. Imran, N. A. Zafar, M. A., Alnuem, M. S. Aksoy, A. V. Vasilakos," Formal verification and validation of a movement control actor relocation algorithm for safety–critical applications" Wireless Networks, vol. 22(1), pp. 247-265, 2016.
[25] S. Latif, H. Afzaal, N. A. Zafar, "Modeling of Sewerage System Using Internet of Things for Smart City", International Conference on In Frontiers of Information Technology (FIT), pp. 46-5, 2017.
[26] H. Afzaal, N. A. Zafar, F. Alhumaidan, "Hybrid subnet-based node failure recovery formal procedure in wireless sensor and actor networks," International Journal of Distributed Sensor Networks, vol. 13(4), 2017.
[27] H. Afzaal, N. A. Zafar, "Formal analysis of subnet-based failure recovery algorithm in wireless sensor and actor and network," Complex Adaptive Systems Modeling, vol. 27(4), pp. 1-27, 2016.
[28] H. Afzaal and N. A. Zafar, "Formalism of network restructuring in flood control system," International Conference on Innovations in Electrical Engineering and Computational Technologies, 2017.
[29] H. Afzaal and N. A. Zafar, "Formal localized reactive subnet-based failure recovery model for sparsely connected wireless sensor and actor networks, 9th IEEE International Conference on Open-Source Systems and Technologies (ICOSST), 2015.
[30] H. Afzaal, N. A. Zafar, "Implementing partitioning detection and connectivity restoration in WSAN using VDM-SL, 13th International Conference on Frontiers of Information Technology (FIT'15), 2015.
[31] A. Shingare, A. Pendole, N. Chaudhari, P. Deshpande, S. Sonavane, "GPS supported city bus tracking & smart ticketing system," IEEE International Conference on Green Computing and Internet of Things, pp. 93-98, 2015.
[32] S. Latif, H. Afzaal, N.A Zafar, "Intelligent traffic monitoring and guidance system for smart city", IEEE International Conference on In Computing, Mathematics and Engineering Technologies (iCoMET), pp. 1-6, 2018.
[33] S. Latif, N.A. Zafar, "A survey of security and privacy issues in IoT for smart cities", Fifth International Conference on Aerospace Science & Engineering (ICASE), pp. 1-5, 2017.
[34] H. Afzaal, N. A. Zafar, "Formal modeling and algorithm of subnet-based backup assigning in WSAN", International Conference on In Information and Communication Technologies (ICICT), pp. 1-6, 2015.
[35] N. A. Zafar, S. A. Khan, K. Araki, "Towards the safety properties of moving block railway interlocking system", Int. J. Innovative Comput., Info & Control, vol 8(7), pp. 5677-5690, 2012.
[36] N. A. Zafar, "Modeling and formal specification of automated train control system using Z notation", IEEE In Multitopic Conference, INMIC'06, pp. 438-443, 2006.
[37] SCSK Corporation, "VDM tools, User Manual," Version 9.0.2, 2013.