The Z Specification for Exam Scheduling System (ESS) thru Genetic Algorithm

Rosziati Ibrahim
Dept. of Software Engineering
Universiti Tun Hussein Onn Malaysia
Parit Raja, Malaysia
rosziati@uthm.edu.my

Ammar Aminuddin Bani Amin
Dept. of Software Engineering
Universiti Tun Hussein Onn Malaysia
Parit Raja, Malaysia
ammar.aminuddin.bani.amin@gmail.com

Mohd Zainuri Saringat
Dept. of Software Engineering
Universiti Tun Hussein Onn Malaysia
Parit Raja, Malaysia
zainuri@uthm.edu.my

***Abstract*— Formal methods are widely used by using mathematical notations to precisely express requirements specification. In this paper, we discuss the formal specification using a case study of Examination Scheduling System (ESS). ESS is a system that generates exam schedule by using genetic algorithm. ESS is developed to solve the exam scheduling problem in order to reduce time-consuming for conventional manually managing the exam venue. By using genetic algorithm, it helps to generate exam schedule very fast and reduce time taken on processing the exam schedule manually. The formal specification of ESS using Z language is also presented in order to show the mapping of the implementation of ESS with its design by using the formal specification. The Z schema can effectively remove ambiguity presents in natural language specification. Hence, this will help to reduce defect in developing the system during implementation phase.**

Keywords*— *scheduling system, genetic algorithm, UML specification, formal method, class diagram, use-case diagram

I. Introduction

Implementation phase in software development lifecycle (SDLC) is crucial part where the requirements analysis and design are put together to produce a good system software. Because of many elements are involved in the implementation phase, system software is prone to error and incomplete. Apart from that, requirements analysis can have ambiguity if presented in the natural language specification.

Because of the ambiguity in requirements of the system, the formal specification is often used to specify the system's requirements [1]. Based on mathematical symbols, Z language can be used to represent the formal specification of any system [2].

This paper discusses the requirements of Examination Scheduling System (ESS) in order to establish the formal specification by using genetic algorithm. The formal specification is represented by the Z language. This paper consists of 7 Sections. Related works are discussed in Section 2 and Section 3 demonstrates the case study of examination scheduling system. Section 4 proposes the Z schema based from the UML class diagram. The implementation of ESS using Java Programming language is discussed in Section 5. Section 6 shows the results after the implementation phase and the concluding remarks are in Section 7.

II. Algorithms for Examination Scheduling System

Some of the algorithms regarding examination scheduling system are discussed in this section. Z language has been used for specifying requirements in formal notations. Most researchers are using Z language to overcome the ambiguity of the requirements such as [3], [4] and [5]. Some researchers are using formal specification for the mapping purposes such as [6], [7], [8] and [9].

Hosny & Al-Olayan [10] discussed a mutation-based genetic algorithm for solving the examination scheduling problem. The proposed solution will find the best room assignment for a number of exams and then the schedule will be assigned to a proctor for supervision. The algorithm has been successfully implemented on a female college. They have save time on the planning for the college.

Elsaka [11] proposed Examination timetable (ETT) using constraint satisfaction modelling (CSP) for a rich data such as large numbers of student enrolment related to many department and lack of examination hall and short period of the exam. CSP has been chosen for optimization programming language to model the examination timetable dataset. The results of using CSP have manage to solve the generating of examination timetable manually..

Wu *et al.* [12] discussed the examination scheduling using NP-complete and presented the so-called graph coloring reduction of the examination timetabling minimum timeslot used. They investigate on the implantation of a heuristic hybridized with simulated annealing-based heuristic using three different neighborhood structure. They manage to schedule the examination timetable without any repetition. The main goal of their paper is to find a feasible solution for timetabling with minimum time slot and the student exam shouldn't crash at the same time with different subjects.

Wahaishi & Aburukba [13] presented an Agent-Based Personal Assistant architecture that provides an innovative approach to automate the scheduling processes as well as allocating different proctors and exam rooms and provides automated personal assistance activities. Their approach resolves any possible exam conflicts and hence delivers a complete examination timetable.

Shatnawi *et al.* [14] discussed the process of the exam schedule. They mentioned that the large number of students, teachers, and the location of the rooms for the manual scheduling are time consuming. Therefore, their approach on exam scheduling is using two algorithms, which are greedy algorithm and genetic algorithm. Their approach reduces the number of conflict, venue and exam days required. They implemented their system using C++ language.

III. The Examination Scheduling System (ESS)

The examination scheduling system is presented in this paper to demonstrate how the requirements are captured. Let say if we want to develop a simple examination scheduling system (ESS). The system receives the input data (name of the subject, time and duration of the exam) from the user, searches the availabilities of the rooms and schedules the exam slot accordingly. To ensure the search algorithm is working properly, the predefined database must already have preset slots for examinations.

978-1-7281-2680-7/20/$31.00 ©2020 IEEE

The first phase in SDLC is the requirements analysis phase. For this simple ESS, the requirements of the system is as follows:

- To login the system
- If success on Login for Admin, then do the following:
  - Add Room
  - Add Subject
  - Add Lecturer
  - Add faculty
  - Add Course
  - Generate Schedule
  - Display Schedule
- If success on Login for Student, then:
  - Display schedule

For ESS, genetic algorithm is used to automatically arrange the scheduling slot. The genetic algorithm is as follows:

- Initialization
- Do fitness assignment
- Do Selection
- Do Crossover
- Do Mutation
- If the criteria is success (where no conflict on each slot scheduling),
  - schedule generated
- Else, re-loop generation to fitness

Unified Modelling Language (UML) specification is used for the requirements analysis. In this paper we only show two diagrams, which are use case and class. UML is a semi language that can specify the requirements of any system. It consists of 13 diagrams [15]. Fig. 1 shows the diagram of use case for the ESS.

Based from Fig. 1, there are two actors which are Admin and Student. An Admin can login, add room, add subject, add lecturer, add course, generate schedule and display schedule. Meanwhile, a student can only view the display schedule generated by the admin.

Based from Fig.1, the class diagram can be drawn from the use case diagram. Fig. 2 shows the class diagram identified by the use cases from Fig. 1. Class diagram consist of class name, attributes and operations. Based on the ESS there are six classes. The classes are User, Student, Room, Admin, Course, Subject and Lecturer. Each of the classes have their own respective operations.

IV. From UML Specification to Z Specification

Z language is then used from the UML specification. Based from Fig. 2, the class diagram is used to map the necessary information to Z Schema prior to implementation phase in SDLC. The Z Schema includes the schema for each classes in Fig. 2. They are Admin, Room, Subject, Course, Lecturer and Student. The ESS provides login, add room, add subject, add lecturer, add course and generate schedule. Fig. 3 shows the Z schema for login method. The username and password are required in order to login into the system.

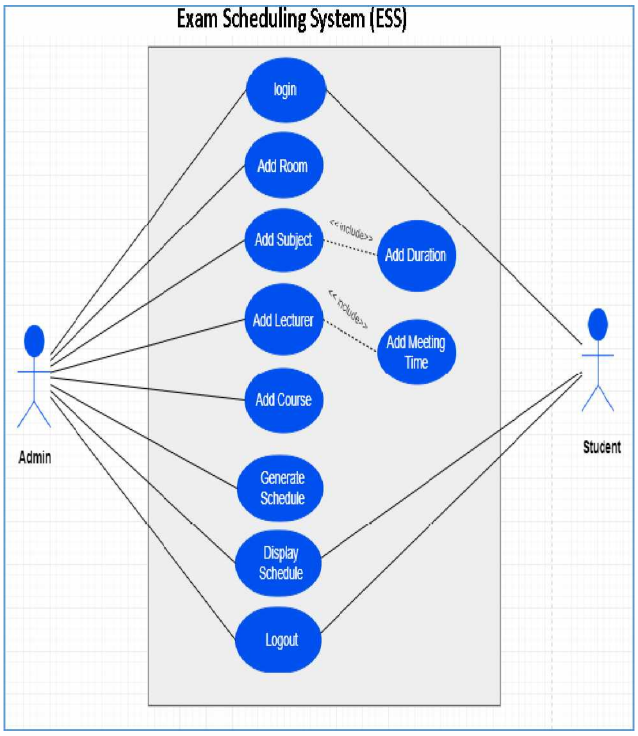
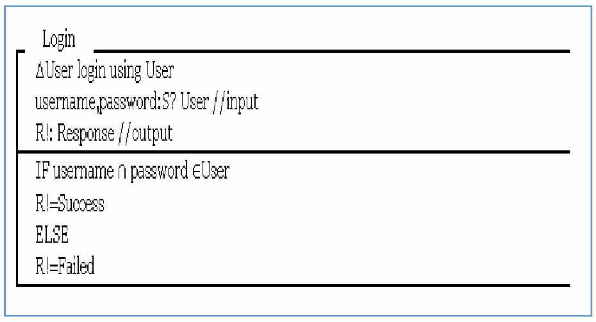


Fig. 1. Use-Case Diagram for EES



```
Login
ΔUser login using User
username,password:S? User //input
R!: Response //output
IF username ∩ password ∈User
R!=Success
ELSE
R!=Failed
```

Fig. 3. Z Schema for Login

Once login is successful, Admin is able to add room, add lecturer, add courses, add subject and generate schedule. Fig. 4 shows that Z schema for generating schedule for the Admin. The system will get all the input required in order to implement the genetic algorithm such as room, subject and course. Then the system will check for consistency either the schedule have cost conflict or not. If there is no conflict between the schedules, the system will generate and display the schedule as shown in Fig. 4.

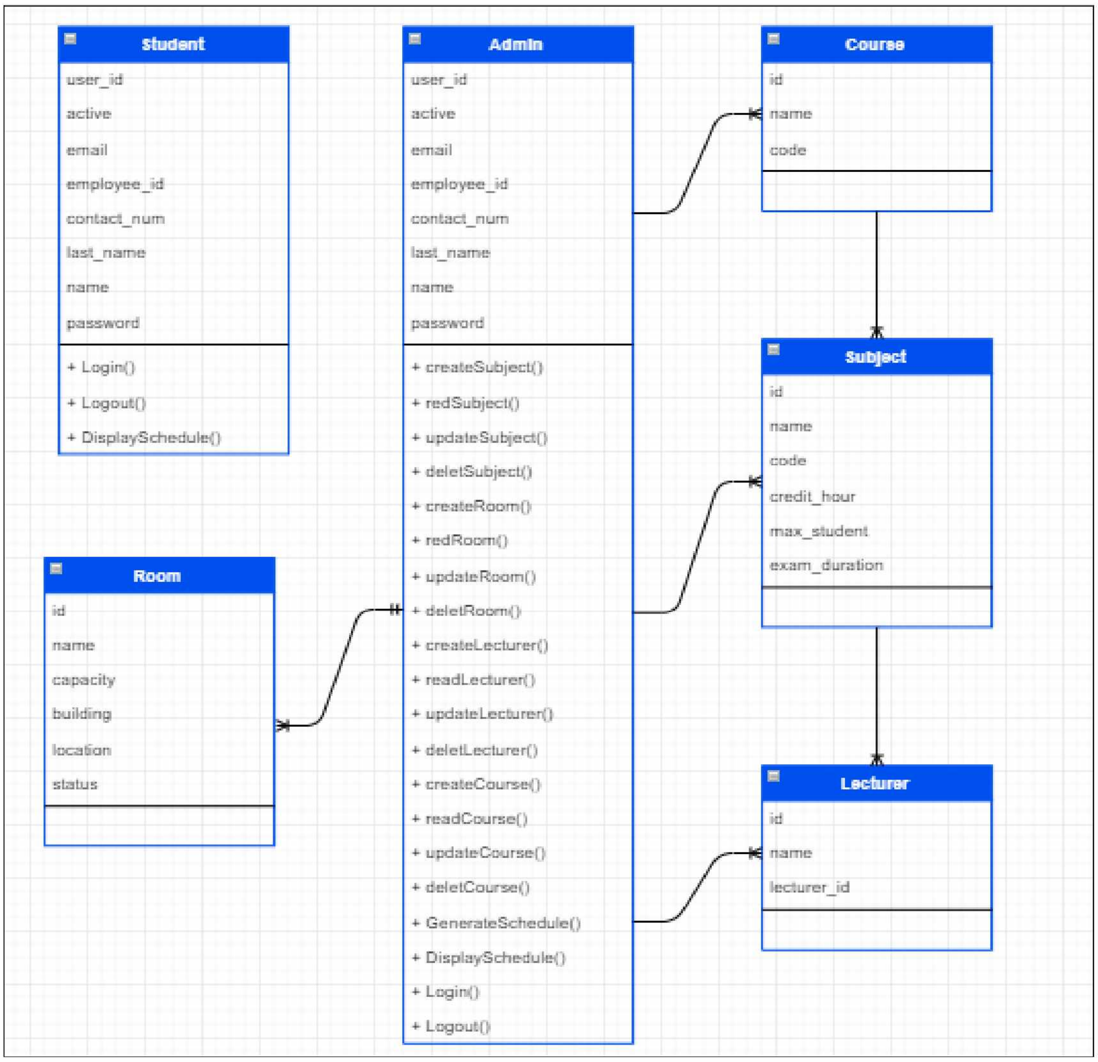


Fig. 2. Class Diagram for EES

```
GenerateSchedule
ΔSchedule //generate schedule using Schedule
id,room_name,capacity: R? Room //input
id,code,name,max_student,exam_duration,exam_time: S? Subject
id,code,subjectId : C? Course
R! = Response
IF Conflict=0 ∧ Schedule ∈ Room,Subject,Course
R! = Success
ELSE
R! Failed
```

Fig. 4. Z Schema for Generate Schedule

For add room, add lecturer, add courses, and add subject, the Z Schemas are almost the same. For that, only add room method is shown as in Fig. 5. Fig. 6 shows the Z schema for the display schedule method.

```
CreateRoom
ΔRoom //add new room using Room
id,name,capacity,building,location,status : S? Room //input
R! : Response //output
IF S? ∉ Room
S! = S?
R! = Success
ELSE
R!=Failed
```

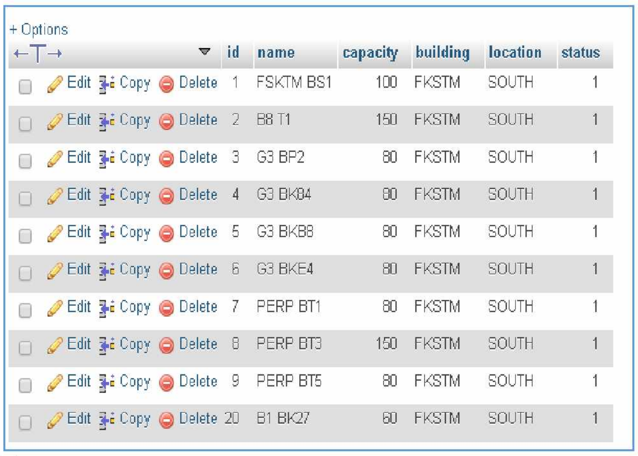
Fig. 5. Schema for Add Room

```
DisplaySchedule
ΔSchedule //display schedule using Schedule
id : S? Schedule //input
R! = Response //output
IF id∈Schedule
R! = Success
ELSE
R! = Failed
```

Fig. 6. Z Schema for Display Schedule

V. Implementation of EES

In this section, we discuss how to generate a scheduling system using Java language. The front-end of the system is using HTML and database is using MySQL. Based on the case study, we are generating the scheduling system by using real data from Fakulti Sains Komputer dan Teknologi Maklumat (FSKTM). The data consists of 57 lecturers, 66 subjects being offered and 20 rooms with its capacity. Fig. 7 shows record from room table, where all the details are listed such as capacity, name of the room and location. All the record in each table can be insert by the Admin using the front end of the ESS.


+ Options

←T→	id	name	capacity	building	location	status
Edit Copy Delete	1	FSKTM BS1	100	FKSTM	SOUTH	1
Edit Copy Delete	2	B8 T1	150	FKSTM	SOUTH	1
Edit Copy Delete	3	G3 BP2	80	FKSTM	SOUTH	1
Edit Copy Delete	4	G3 BK84	80	FKSTM	SOUTH	1
Edit Copy Delete	5	G3 BKB8	80	FKSTM	SOUTH	1
Edit Copy Delete	6	G3 BKE4	80	FKSTM	SOUTH	1
Edit Copy Delete	7	PERP BT1	80	FKSTM	SOUTH	1
Edit Copy Delete	8	PERP BT3	150	FKSTM	SOUTH	1
Edit Copy Delete	9	PERP BT5	80	FKSTM	SOUTH	1
Edit Copy Delete	20	B1 BK27	60	FKSTM	SOUTH	1

FFig. 7. Room Table Record for EES

In order to generate the schedule for the exam, all the records must have all the information such as lecturer records, room records, subject records and others. We are implementing genetic algorithm using java language in order to generate the scheduling. Fig. 8 shows the genetic algorithm flowchart where several processes start from population, selection, crossover and mutation. Fig. 9 shows the code for genetic algorithm in java programming language. For each process, the system will check its consistency on each slot time either is redundant or not. If there is a redundancy, the system will flag as conflict. The iteration of each generation will be increased until such conflict is return zero. If the conflict is zero then the stopping criteria is true and the genetic algorithm will end. Finally the scheduling will be displayed. The results of the scheduling will be discussed in next Section.

VI. Simulation Results

We have implemented the EES using Java programming language. The simulation results for EES are based on genetic algorithm. Based on Fig. 10, we found out that ESS generates fifteen (15) iterations of generation to finalize the scheduling. We also calculate the total amount of time taken for the whole process. Using the data given, the system can process and complete the scheduling within one second. Table 1 shows comparison of the time taken between small and medium data on using the genetic algorithm in order to produce the exam scheduling system.

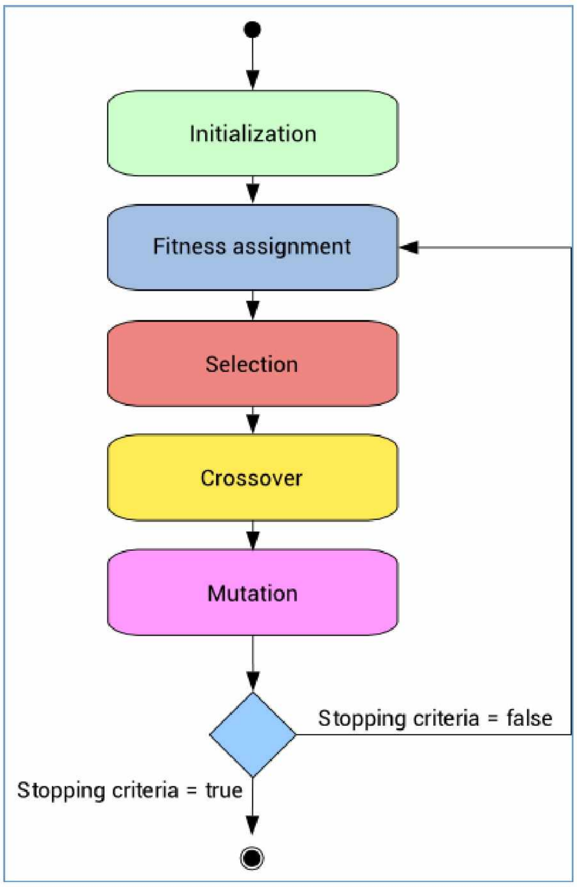
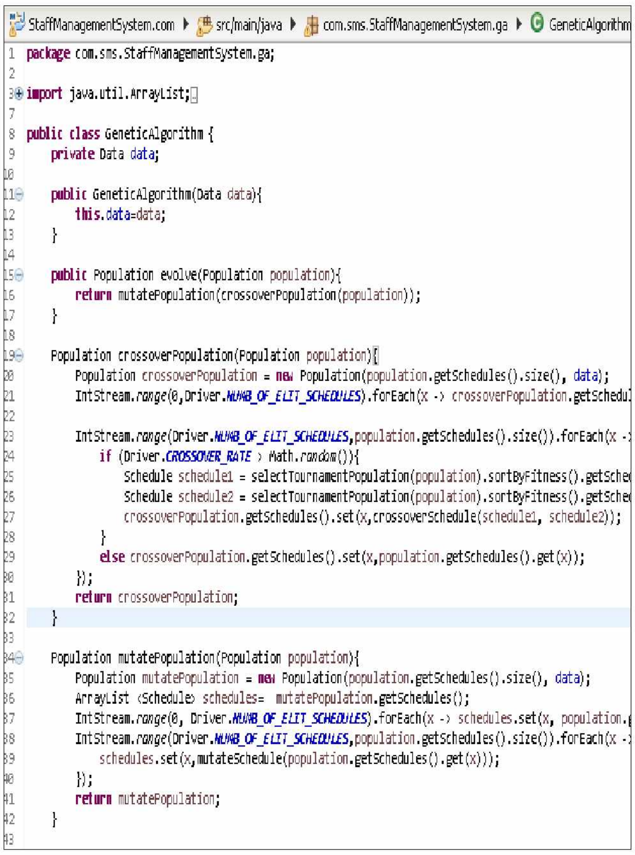
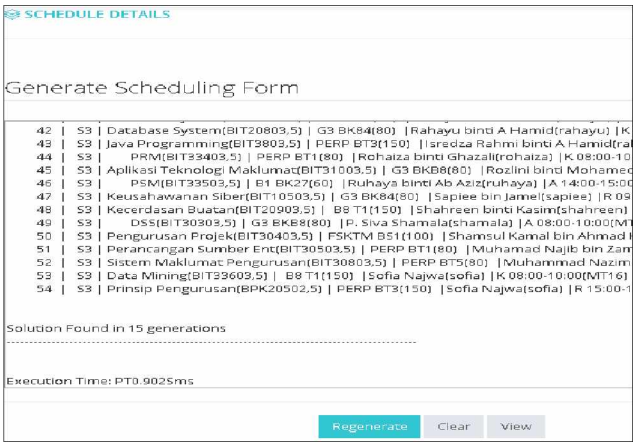


Fig. 8. Genetic Algorithm Flowchart


StaffManagementSystem.com ▸ src/main/java ▸ com.sms.StaffManagementSystem.ga ▸ GeneticAlgorithm

```java
package com.sms.StaffManagementSystem.ga;

import java.util.ArrayList;

public class GeneticAlgorithm {
    private Data data;

    public GeneticAlgorithm(Data data){
        this.data=data;
    }

    public Population evolve(Population population){
        return mutatePopulation(crossoverPopulation(population));
    }

    Population crossoverPopulation(Population population){
        Population crossoverPopulation = new Population(population.getSchedules().size(), data);
        IntStream.range(0,Driver.NUMB_OF_ELIT_SCHEDULES).forEach(x -> crossoverPopulation.getSchedul

        IntStream.range(Driver.NUMB_OF_ELIT_SCHEDULES,population.getSchedules().size()).forEach(x -
            if (Driver.CROSSOVER_RATE > Math.random()){
                Schedule schedule1 = selectTournamentPopulation(population).sortByFitness().getSche
                Schedule schedule2 = selectTournamentPopulation(population).sortByFitness().getSche
                crossoverPopulation.getSchedules().set(x,crossoverSchedule(schedule1, schedule2));
            }
            else crossoverPopulation.getSchedules().set(x,population.getSchedules().get(x));
        });
        return crossoverPopulation;
    }

    Population mutatePopulation(Population population){
        Population mutatePopulation = new Population(population.getSchedules().size(), data);
        ArrayList <Schedule> schedules=  mutatePopulation.getSchedules();
        IntStream.range(0, Driver.NUMB_OF_ELIT_SCHEDULES).forEach(x -> schedules.set(x, population.
        IntStream.range(Driver.NUMB_OF_ELIT_SCHEDULES,population.getSchedules().size()).forEach(x -
            schedules.set(x,mutateSchedule(population.getSchedules().get(x)));
        });
        return mutatePopulation;
    }
```

Fig. 9. Implementation of Genetic Algorithm



SCHEDULE DETAILS

Generate Scheduling Form

```
   42  |   S3 | Database System(BIT20803,5) | G3 BK84(80)  |Rahayu binti A Hamid(rahayu) |K
   43  |   S3 | Java Programming(BIT3803,5) | PERP BT3(150)  |Isredza Rahmi binti A Hamid(ra
   44  |   S3 |      PRM(BIT33403,5) | PERP BT1(80)  |Rohaiza binti Ghazali(rohaiza) |K 08:00-10
   45  |   S3 | Aplikasi Teknologi Maklumat(BIT31003,5) | G3 BKB8(80)  |Rozlini binti Mohamed
   46  |   S3 |      PSM(BIT33503,5) | B1 BK27(60)  |Ruhaya binti Ab Aziz(ruhaya) |A 14:00-15:0
   47  |   S3 | Keusahawanan Siber(BIT10503,5) | G3 BK84(80)  |Sapiee bin Jamel(sapiee) |R 09
   48  |   S3 | Kecerdasan Buatan(BIT20903,5) |  B8 T1(150)  |Shahreen binti Kasim(shahreen)
   49  |   S3 |      DSS(BIT30303,5) | G3 BKB8(80)  |P. Siva Shamala(shamala) |A 08:00-10:00(M
   50  |   S3 | Pengurusan Projek(BIT30403,5) | FSKTM BS1(100)  |Shamsul Kamal bin Ahmad
   51  |   S3 | Perancangan Sumber Ent(BIT30503,5) | PERP BT1(80)  |Muhamad Najib bin Za
   52  |   S3 | Sistem Maklumat Pengurusan(BIT30803,5) | PERP BT5(80)  |Muhammad Nazim
   53  |   S3 | Data Mining(BIT33603,5) |  B8 T1(150)  |Sofia Najwa(sofia) |K 08:00-10:00(MT16)
   54  |   S3 | Prinsip Pengurusan(BPK20502,5) | PERP BT3(150)  |Sofia Najwa(sofia) |R 15:00-1

Solution Found in 15 generations
-----------------------------------------------------------------------

Execution Time: PT0.9025ms
```

Fig. 10. Generate Exam Schedule

TABLE 1. COMPARISON TIME BETWEEN DATA SIZE

Premises	Time in seconds
Small Data Set (consists of 20 records)	0.0480ms
Medium Data Set (consist of 60 records)	0.9025ms

Based from Table 1, based on small dataset, the ESS manage to generate the exam schedule within 0.048ms while for medium dataset, ESS still manage to generate exam schedule less than 1.00ms.

VII. Conclusion

This paper discussed the development of examination scheduling system (ESS) thru genetic algorithm. The formal specification of EES has also been presented using Z language. The Z schema can effectively remove ambiguity presents in natural language specification. Hence, this will help to reduce ambiguity in system's requirements as well as reducing the cost in implementing the system automatically.

Acknowledgment

The authors would like to thanks Universiti Tun Hussein Onn Malaysia (UTHM) for supporting this study.

References

[1] J. M. Spivey, "Understanding Z: A Specification Language and Its Formal Semantics," Cambridge University Press, 1988.

[2] J. M. Spivey, "The Z notation: A Reference Manual," Second Edition. Prentice Hall, 1992.

[3] P. Saratha, G. V. Uma, and B. Santhosh, "Formal Specification for Online Food Ordering System using Z Language," 2nd International Conference on Recent Trends and Challenges in Computational Models (ICRTCCM'2017), 2017.

[4] M. W. Azeem, M. Ahsan, M. Minhas, K. Noreen, "Specification of e-Health System using Z: A Motivation to Formal Methods," Proceedings of International Conference for Convergence for Technology (ITCCT'2014), 2014.

[5] S. H. Bakri, H. Harun, A. H. M. Alzoubi, and R. Ibrahim, "The formal specification for the inventory system using Z language," Proceedings of ICOCIE, 2013.

[6] S. Y. Wong, E. Mit, and J. Sidi, "Integration of use case formal template using mapping rules," Proceedings of 3rd International Conference on Information Retrieval and Knowledge Management (CAMP'2016), 2016.

[7] H. Aman, and R. Ibrahim, "Formalization of Transformation Rules from XML Schema to UML Class Diagram." Int. J. Softw. Eng. Appl., Vol. 8, No.12, pp.75 – 90, 2014.

[8] B. Mondal, B. Das, and P. Banerjee, "Formal Specification of UML Use Case Diagram – A CASL based Approach," Int. J. Comput. Sci. Inf. Tech., Vol. 5, No. 3, pp. 2713 – 2717, 2014.

[9] N. Ibrahim, R. Ibrahim, M. Z. Saringat, D. Mansor, and T. Herawan, "Consistency rules between UML use case and activity diagrams using logical approach," Int. J. Softw. Eng. Appl., Vol. 5, No. 3, pp. 119 – 134, 2011.

[10] M. Hosny, and M. Al-Olayan, "A mutation-based genetic algorithm for room and proctor assignment in examination scheduling," Proceedings of 2014 Science and Information Conference, SAI 2014, pp. 260 – 268, 2014.

[11] T. Elsaka, "Autonomous generation of conflict-free examination timetable using constraint satisfaction modelling" IDAP 2017 - International Artificial Intelligence and Data Processing Symposium, 2017.

[12] X. Wu, J. Li, R. Xu, and T. Yu, "A simulation study of appointment scheduling for multi-class MRI examination," 2016 13th International Conference on Service Systems and Service Management, pp. 1 – 6, 2016.

[13] A. M. Wahaishi, and R. O. Aburukba, "An agent-based personal assistant for exam scheduling," 2013 World Congress on Computer and Information Technology, 2013.

[14] A. Shatnawi, M. Fraiwan, and H. S. Al-Qahtani, "Exam scheduling: A case study," 9th International Conference on Advanced Computational Intelligence, ICACI, pp. 137 – 142, 2017.

[15] UML2.5.1 (2019). OMG UML Specification. [Online]. Available: http://www.omg.org/spec/UML/2.5.1/