View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by E-space: Manchester Metropolitan University's Research Repository

Ahmad, Jamil and Latif, Muhammad and Horejsi, Petr (2018)Investigation of an elevator dispatcher system. MM Science Journal, 12 (2018). pp. 2593-2600. ISSN 1803-1269

Downloaded from: http://e-space.mmu.ac.uk/624980/

Version: Accepted Version

Publisher: MM Publishing, s.r.o.

DOI: https://doi.org/10.17973/mmsj.2018_12_2018028

Please cite the published version

https://e-space.mmu.ac.uk

INVESTIGATION OF AN ELEVATOR DISPATCHER SYSTEM

JAMIL AHMAD[1], MUHAMMAD LATIF[1], AND PETR HOREJSI[2]

[1] School Of Engineering, Manchester Metropolitan University, Manchester, United Kingdom

[2] Faculty Of Mechanical Engineering, University Of West Bohemia, Pilsen, Czech Republic

DOI: 10.17973/MMSJ.2018_12_2018028

e-mail: tucnak@kpv.zcu.cz

To provide an efficient elevator system, a variety of diverse and sometimes conflicting constraints have to be solved. This paper focuses on using discrete event simulation as a means to model and explore elevator dispatching strategies. Witness simulation software has been used as a test-bed for model building, simulation and some experimentation. Model building using Witness model elements is described in detail followed by two different elevator call strategies. The main steps in the methodology are described with reference to a single elevator servicing a five-storey office building. The elevator call strategies are simulated and results compared. It was found that the data set significantly skews the results and overshadows any efficiency gains that might be possible from the different dispatching strategies. The paper concludes with the need to carefully select the data set as the basis for simulation comparisons and outlines future work required.

KEYWORDS

Elevator Dispatcher System, Simulation, Witness, Sequence Dispatcher, Shortest Travel Distance, Most Different Destinations

1 INTRODUCTION

Major cities in the UK have experienced a national programme of redevelopments often resulting in tall buildings with small footprints [Yuan 2008]. Existing buildings in many cases have been affected by urban generation programmes, often with a change of use from warehousing to commercial/residential. All these buildings new or old require an efficient floor transportation system, which traditionally is an elevator. Buildings often experience elevator congestion because of their heavy traffic, complex user types, and relatively slow-moving elevators (due to safety concerns) [Al-Sharif 2018, Nagatani 2003]. Yet waiting for an elevator can be one of the main annoyances in one's experience with tall buildings [Berbeglia 2010, Sutton 1998].

How long we wait depends on the dispatching strategy the elevators use to decide where to go. Not surprisingly, the times of greatest traffic and the greatest challenge to the dispatching algorithm are the morning and evening rush hours [Lee 2009]. Dispatchers are generally designed primarily for these difficult periods. Despite good designs, dispatchers have not achieved the efficiency levels that society expects, often resulting in the most common complaint, that the "waiting time was far too high" [Tebbenhof 2000].

Research into elevator dispatching is quite recent and has followed the development of technology. The late eighties and the nineties can be considered as the starting point, especially in the USA and Japan [Robert 1988, Thangavelu 1989]. The focus of research during the last two decades has been on controls, mechanisms, safety, etc. whilst using simple dispatching algorithms. However, in more recent times some researchers have been focusing on utilising artificial intelligence in elevators [Zheng 2013, Tanaka 2005].

Although some researchers have explored the use of simulation [Cortes 2004, Ahn 2017] or simulation optimization [Zhang 2013] in modelling an elevator, they have not directly addressed the effectiveness of dispatching algorithms within in-service elevators. Without an appropriate computer model, it is difficult to develop and test the performance of an elevator dispatcher algorithm.

2 AIM OF RESEARCH

The aim of the research was to design and develop a computerized Elevator Dispatcher System (EDS). To build a model of an EDS in Witness - discrete event simulation software. To monitor the performance of different elevator dispatching algorithms and explore strategies to reduce the average waiting time, the average system time and to increase the efficiency of the current operational system, while maintaining an acceptable level of operating ease and convenience.

3 CASE STUDY: OFFICE BUILDING, MANCHESTER, UK

An office building was selected for this case study comprising of five floors serviced by a single elevator. The floors have the notation: Ground (G), Floor 1(F1), Floor 2 (F2), Floor 3 (F3) and Floor 4 (F4). The elevator travelling time per floor was 20 seconds, comprising 5 seconds for opening, 5 seconds for closing the carriage doors and 10 seconds for carriage elevation. The simulation tool selected for this work was the Lanner Group's Witness software [Lanner 2018], which is part of a new generation of visually interactive simulation software.

DATA COLLECTION

Elevator traffic data was obtained through rigorous observations for a full working day as shown in tables 1 and 2. The elevator traffic data was converted into a series of schedules with half hourly-based durations to represent each travel direction. Note – a full inter-floor movement is not considered at this stage but is the subject of a later publication. Passengers leaving the elevator on the ground floor from all floors is considered at this stage.

Data Collection

Monday	Ground Floor Inbound Data				
Time	**G - F1**	**G - F2**	**G - F3**	**G - F4**	**Total**
08:30 - 09:00	1	2	8	15	**26**
09:00 - 09:30	1	3	9	16	**29**
09:30 - 10:00	2	2	6	16	**26**
10:00 - 10:30	2	1	7	15	**25**
10:30 - 11:00	0	1	2	4	**7**
11:00 - 11:30	0	2	4	5	**11**
11:30 - 12:00	0	2	3	4	**9**
12:00 - 12:30	0	1	2	4	**7**
12:30 - 13:00	3	4	7	5	**19**
13:00 - 13:30	3	3	6	5	**17**
13:30 - 14:00	2	1	7	8	**18**
14:00 - 14:30	1	1	2	4	**8**
14:30 - 15:00	1	1	3	3	**8**
15:00 - 15:30	0	1	1	2	**4**
15:30 - 16:00	1	2	2	2	**7**
16:00 - 16:30	0	0	1	3	**4**
16:30 - 17:00	0	0	0	2	**2**
	17	**27**	**70**	**113**	**227**

Table 1. Inbound elevator traffic data

Monday	Outbound Data From Other Floors				
Time	**F1 - G**	**F2 - G**	**F3 - G**	**F4 - G**	**Total**
08:30 - 09:00	0	0	2	0	**2**
09:00 - 09:30	0	2	4	3	**9**
09:30 - 10:00	2	2	3	3	**10**
10:00 - 10:30	1	3	3	3	**10**
10:30 - 11:00	0	0	1	2	**3**
11:00 - 11:30	1	1	0	1	**3**
11:30 - 12:00	0	0	1	2	**3**
12:00 - 12:30	1	1	1	1	**4**
12:30 - 13:00	2	5	7	14	**28**
13:00 - 13:30	2	3	5	14	**24**
13:30 - 14:00	2	4	7	19	**32**
14:00 - 14:30	1	1	0	3	**5**
14:30 - 15:00	1	1	1	2	**5**
15:00 - 15:30	0	1	3	2	**6**
15:30 - 16:00	1	3	4	14	**22**
16:00 - 16:30	2	3	3	15	**23**
16:30 - 17:00	2	2	5	25	**34**
	18	**32**	**50**	**123**	**223**

Table 2. Outbound elevator traffic data

MODEL BUILDING

Three elevator simulation models have been developed; Sequence Dispatcher (SD) model and two different scenario cases. Scenario 1- Shortest Travel Distance (STD) and Scenario 2- Most Different Destination (MDD). A few model elements and their working details are described and illustrated in this paper.

Assumptions for Model

- Before creating a model, the following points are considered:
- What logic to use to move the Elevator?
- How should the Elevator respond to floor calls?
- How to ensure the Elevator will answer the passenger waiting at a specific floor?
- How to find out how many passengers are already in the elevator?
- How to ensure the Elevator takes people only to its capacity?
- How to ensure the Elevator drops people at their desired destination floors?
- How to ensure the Elevator will control the passengers using FCFS (First Come First Served) logic?
- How to ensure the Elevator will stay on the floor if people are getting in or out?
- How to find out the exact state of the Elevator from different possible states (Loading, Loaded, Parked, In Demand, Calling, and Free, Out of Model, etc.)?
- How to change direction and control the Elevator when it reaches either the top or ground floor?

Some built-in functions like Istate, SetVehicleDestination, TrackNumber, TrackName, Match, Nents And some user defined functions like ChangeDirection, FreeTravel, EmptyTravel, NextFloor, NeedLoad, NeedUnload, Desti, and some built-in decision making statement If, Match and some loop statements For, and While are used to overcome the above issues. Table 3 shows the possible states of a vehicle model element which is used as an elevator in this model.

State no.	State	Colour
0.	Off-shift	White
1.	Free	Yellow
2.	Demanded	Cyan
3.	Blocked	Magenta
4.	Loaded	Green
5.	Loading	Blue
6.	Unloading	Blue
7.	Stopped	Red
8.	Parked	Dark Green
9.	Outside	Not yet entered the simulation

Table 3. Possible values of ISTATE function for a vehicle element

In all models the elevator operates (moves) if there is a demand for the elevator or if passengers are inside (the elevator is loaded). The rest of the time the elevator waits (Free/Waiting/Parked State) where the last passenger was dropped off and there is no other call at that dropped off floor. The elevator will start moving only when it is called by passengers waiting on other floors. If the elevator is in a loaded mode and some passengers are waiting on other floors (in the direction of movement), the elevator will pick up these passengers until its capacity is reached (Elevator capacity is max 8 people). If the elevator is in loading mode and some passengers are waiting in stations in the direction of movement and some passengers are dropped off who again want to use the Elevator, those passengers who were dropped off at that specific floor have higher priority over passengers waiting to join the elevator for the first time. The next passengers who are waiting on other floors will be picked up in the direction of movement (max 8 people). How passengers are dropped off depends on how they arrived in the actual model using the (FIFO) First In First Out principle.

Figure 1 illustrates the Witness modelling screen displaying the graphics elements and animation of the SD model as described earlier. The developed model (Fig 1) consists of the following discrete stages:

- Passengers entering the model
- Passengers entering the elevator
- Passengers leaving the model after reaching their destination floor
- Making an elevator call
- Elevator control logic start and stop
- Elevator movement between floors
- Changing elevator direction once it has reached top or ground floor

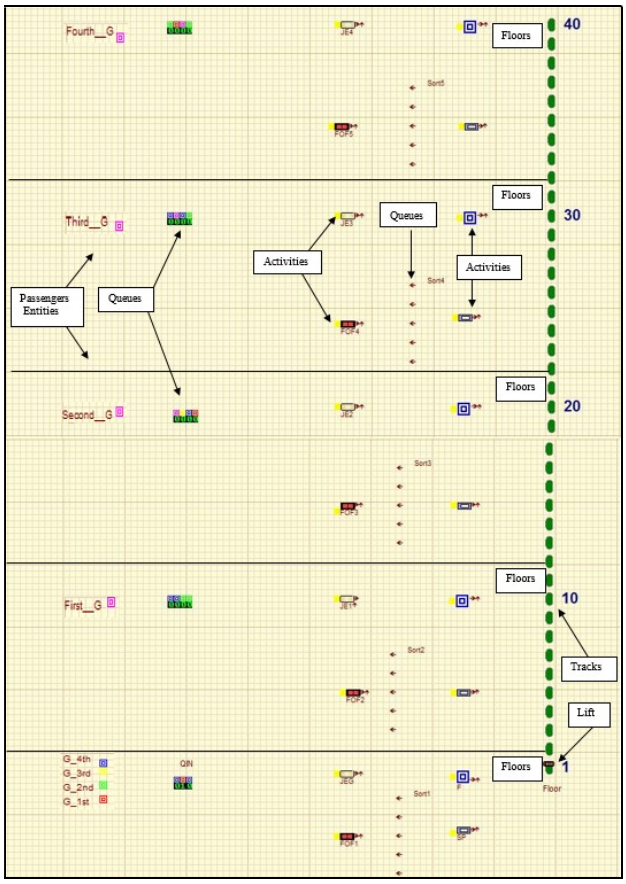

Figure 1. Witness Elevator Model

Passengers Entering the Model

Figure 2 illustrates the control logic enabling entities (passengers) to enter the model and initiate an elevator call at a specific floor. Whilst the Witness modules shown are designated for the ground floor, the control logic is generic and applicable to all floors. The logic is captured in Witness using a multiple instance of one modelling element called Entity. All passengers who want to use the elevator are identified with a different colour scheme as shown in Figure 3.

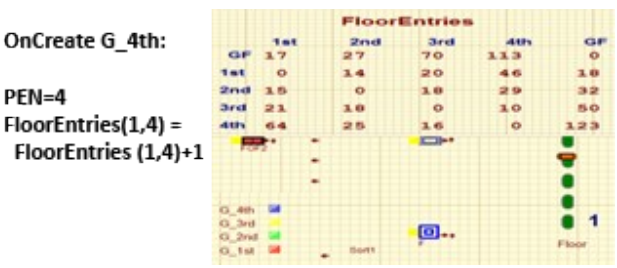

Figure 2. Control logic enabling passengers to enter the model

The arrival profile is based on a schedule derived for each travelling direction depicted in Table 1 and Table 2. The **Qin** model element in Figure 2 is used to hold the passengers in a queue (i.e. QIN) and wait for a specified condition to become true before the people can continue to move through the modelled system. The condition used at **Qin** is shown in the following code:

```
IF ISTATE (Lift) = 8 AND NENTS (QIN(1)) > 0 AND NENTS (F(1)) = 0 AND T
() = 1 AND NENTS (SP(1)) = 0 AND NENTS (FOF1) = 0 AND NENTS (Sort1)
= 0
        PULL from QIN(1)
ELSE
        Wait
ENDIF
```

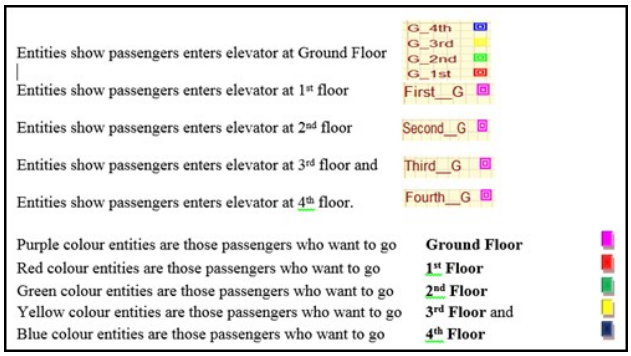

Figure 3. Passenger colour scheme

Elevator Control Logic

Figure 4 illustrates the control logic for moving the elevator. The Witness modules shown are designated for the Ground floor and the control logic is generic and applicable to all floors.

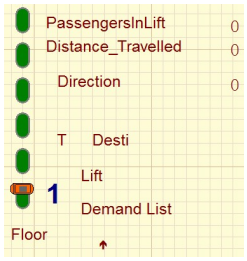

Figure 4. Elevator control logic

The control logic starts with the creation of a single but important model element (Figure 4) to represent the elevator carriage labelled as lift vehicle; it is the main control element and it is generated just once. The lift vehicle element has the following attributes:

- Floor – the current floor, where the elevator is now using the Track element. A user defined function T is created to find the elevator's current location at any time.
- Lift Capacity – the elevator capacity (currently set to 8 people)
- Speed – loading and unloading and transportation between floors speed
- Direction – has a value of 1 if the elevator is moving Up and a value of 0 when the elevator is moving Down
- Passengers In Lift – integer variable is the count of the number of people in the elevator at any time
- Desti – where to go next - a function which sets and tells the elevator the destination of a passenger
- Set Vehicle Destination – a built-in Witness function which changes the destination of the elevator during run time after

making a decision based on the results of some if statements used in the model

- Distance Travelled – a built-in variable which returns an integer value
- Demand list – which stores the list of calls made for the elevator and the next service point

In all three developed elevator simulation models two different control logics have been developed to move the elevator. The first logic is used when there are some passengers waiting for the elevator at a specific floor and there are no other passengers created in the model. This means the rest of the floors have no waiting passengers. While the second control logic is used to move the elevator after serving all the passengers at a specific floor (i.e. Second floor when no more passengers are at that floor level) and when other floors have passengers waiting to use the elevator. Note – Both logics use and check the state of the elevator by using the built-in function ISTATE. If the istate value is equal to 8, it means the elevator is parked. Figure 5 and Figure 6 illustrate the first control logic for moving the elevator. A model element activity called **MonitorArrival** executes every 0.1 second and after checking the result of an If statement it pulls another model element Signal. Whenever this condition becomes true this will then call a number of user defined functions to determine the next loading point for the elevator and it also changes its direction if needed. Figure 6 illustrates the detail and user defined function **ChangeDirection**. The last action of the activity is to eliminate the pulled entity **Signal**.

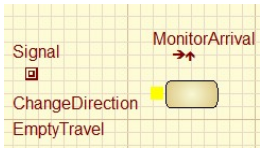

Figure 5. Elevator control 1st logic

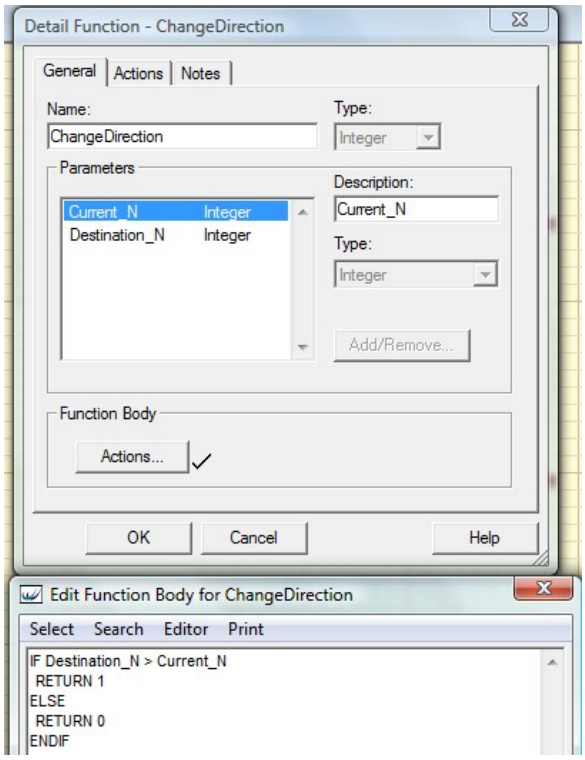

Figure 6. Elevator control 1st logic

The second elevator control logic also uses a number of user-defined functions. A set of different functions are used when the elevator is going up, and a different set of functions when the elevator is going down. The difference between the two control logics is that the first logic is executed every 0.1 second and it acts like a continuous monitoring of the changing situation of the model. The second control logic functions are executed after all the passengers on that floor have been served and there are no passengers waiting at that floor. Figure 7 illustrates those functions used in the model.

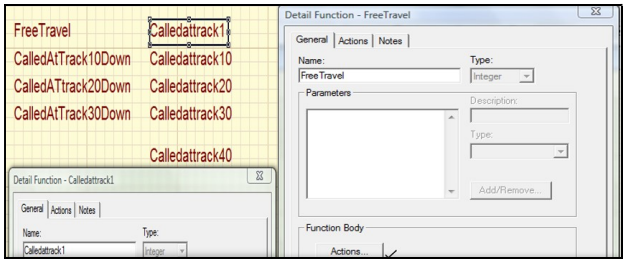

Figure 7. Elevator control 2nd logic

Making an Elevator call

Model elements activities **F** are used as a loading point for a passenger into the elevator. Figure 8 shows the detail behind the single instance of activity **F**. It uses the PEN system attribute which tells the elevator the next unloading points of different passengers. It is an array of activities which we can see from the field **Quantity** set to 5, one activity on each floor. Another field useful for the model is **Type** which lets you set the activity up to join several passengers together to create a new passenger entity, or to join several passenger entities into the first entity that enters into the activity. To ensure **FIFO** is applied to passengers, the box **Join to Entity** is checked which joins passenger entities into the first entity in the activity. By default, the entities in the activity are joined to the first entity that arrives in the activity. **Actions on Output** that only one statement there **De = UnLo (Desti ())** a function **Desti ()** is called which return a possible value between 0 - 5. The value returned by the function is used as the index number for array **UnLo,** which can return a possible value between 1,10,20,30 and 40. The value returned by the **UnLo** array is then stored in variable **De.** The value of **De** is used for unloading the condition on the track **Floor.** The whole purpose of this statement **De = UnLo (Desti ())** is to find the next unloading point for the Elevator by using the **PEN** attribute of the passenger entity. Function FreeTravel triggers the call for the elevator whenever a passenger arrives at activity **F** which moves the elevator as a result to serve the floor.

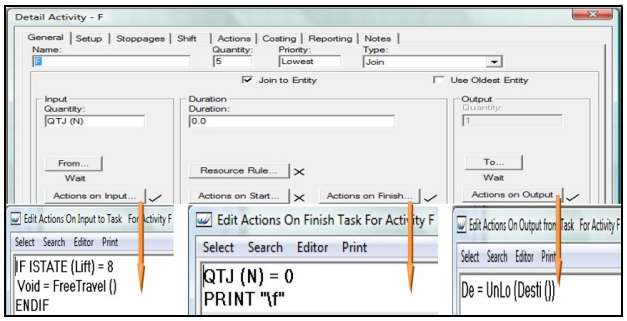

Figure 8. Control logic for making an elevator call

Elevator Movement between Floors

Figure 9 depicts a view of the control logic governing the elevator movement between floors. Model element tracks were used to model the elevator shaft to enable movement of the elevator between floors. A user defined function NextFloor is used which moves the elevator from one-track position to the next. If the elevator direction is up it adds 1 to the current value of the system attribute N until it reaches 40, which is the last unloading point for the elevator (i.e. 4th floor). Similarly, if the direction is down, the NextFloor function subtracts 1 from the current value of N until it reaches the track number 1, which is the first loading point for the elevator (i.e. Ground Floor). A change in direction of the elevator was also made possible once it reaches either the top or bottom of the tracks through function code.

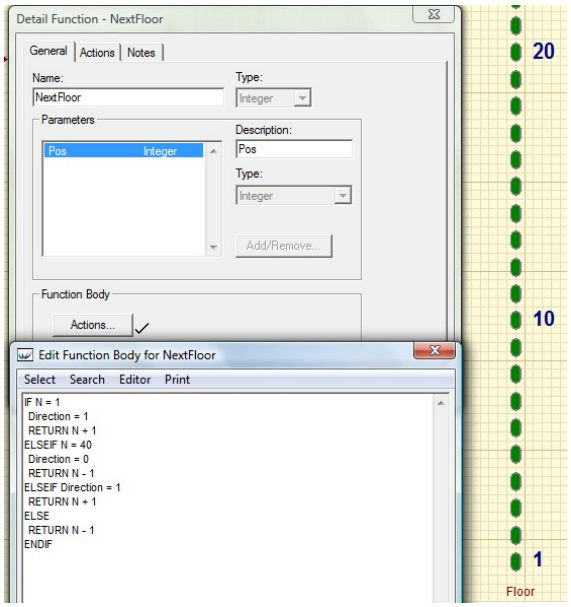

Figure 9. Control Logic of elevator movement between floors

The simulation was set with run duration of 8.5 hours i.e. 0830-1700 hrs.

4 SIMULATION RESULTS FOR SCENARIO SD

The results obtained after the model was simulated for 8.5 hours are shown in tables 4, 5 and 6. The time units are minutes. The waiting time at call points, which is an array of QIN, are depicted in Table 4. Essentially how long the people wait at the floor call points is important and is considered a measure of the service levels and system efficiency. Clearly the shorter the waiting time, the better the experience. Table 4 indicates that the maximum waiting time (1.67 minutes) was experienced on the ground floor at QIN(1) and the minimum waiting time (0.23 minutes) was experienced on the third floor at QIN(4) .

Name	QIN(1)	QIN(2)	QIN(3)	QIN(4)	QIN(5)
Total In	227	72	90	125	164
Total Out	227	72	90	125	164
Now In	0	0	0	0	0
Max	17	8	6	6	5
Min	0	0	0	0	0
Avg Size	0.75	0.05	0.07	0.06	0.29
Avg Time	1.67	0.38	0.41	0.23	0.89

Table 4. Scenario 1 Sequence Dispatcher Elevator Call Waiting Time (units in minutes)

Waiting time for all passengers arriving at different floors is depicted in Table 5. It is important to check how long passengers have to wait in the system before they get served at their desired destination floor. Table 5 indicates that passengers travelling to the second floor from the ground floor have a shorter flow time (i.e. 1.71 minutes on average) than any other inbound destinations in the system. While passengers going to the fourth floor from the ground floor have a higher flow time (i.e. 2.93 minutes on average).

Another useful measurement to consider is how long the elevator was free and not on call during the simulation period. How many loads an elevator made and what was the physical distance travelled by the elevator. From Table 6, it is evident that the elevator is parked for significant amounts of time on the upper floors.

Name	Signal	Check LiftState	G_4th	G_3rd	G_2nd	G_1st	First_G	Second G	Third_G	Fourth G
No. Entered	2158	50947	113	70	27	17	18	32	50	123
No. Served	0	0	113	70	27	17	18	32	50	123
No. Lost	2157	50947	0	0	0	0	0	0	0	0
No. Joined	0	0	0	0	0	0	0	0	0	0
No. Did Not Enter	0	0	0	0	0	0	0	0	0	0
No. In System	1	0	0	0	0	0	0	0	0	0
Avg Number In System	0.42	0	0.65	0.28	0.09	0.08	0.07	0.13	0.18	0.51
Avg time	0.1	0	2.93	2.03	1.71	2.27	2.06	2	1.79	2.13
Sigma Rating	0	0	6	6	6	6	6	6	6	6

Table 5. Scenario 1 Sequence Dispatcher Passenger Service Time (units in minutes)

Name	% Free	% Demand	% Transfer	% Loaded	% Stop	% Blocked	Distance	Loads
Lift	54.2	0	0	45.8	0	0	13067	360

Table 6. Scenario 1 Sequence Dispatcher Elevator Results (units in seconds)

Recognising opportunities for improvements, it was decided to explore alternative scenarios using the same data set shown in Table 1 and Table 2.

5 OTHER SCENARIOS

Two further scenarios were developed. The actual modifications to the model control logic are not detailed in this paper. Numbers of conditions and functions used to control the logic of elevator movement were introduced into two new scenarios and the model was simulated for the same duration, i.e. 8.5 hours.

Scenario 2 Shortest Travel Distance (STD)

The results obtained after the model was simulated for 8.5 hours are shown in tables 7, 8 and 9. The time units are minutes. In this scenario, the control logic governing the movement of the elevator from the SD model was modified so the elevator calculates its next loading point by checking the shortest distance to that floor. The elevator stops on the floors where the call was made or someone wants to be dropped off. Furthermore, a passenger cannot travel in the wrong direction, meaning that if any passenger wants to go from the second

floor to the third floor and the direction of the elevator is down towards the ground floor, that passenger will not be loaded into the elevator. Some new user defined functions are used to control the elevator movement and find the shortest distance after all the passengers are served at a specific floor and there are no more calls pending at that floor. Details of functions and logic are the subject of a later publication. Tables 7, 8 and 9 show the improvement in the results after the simulation is run for the same amount of time i.e. 8.5 hours.

Name	QIN(1)	QIN(2)	QIN(3)	QIN(4)	QIN(5)
Total In	227	72	90	125	164
Total Out	227	72	90	125	164
Now In	0	0	0	0	0
Max	17	8	6	6	5
Min	0	0	0	0	0
Avg Size	0.75	0.05	0.07	0.06	0.29
Avg Time	1.37	0.31	0.33	0.23	0.79

Table 7. Scenario 2 Shortest Travel Distance Elevator Call Waiting Time (units in minutes)

Name	Signal	Check LiftState	G_4th	G_3rd	G_2nd	G_1st	First_G	Second G	Third_G	Fourth G
No. Entered	2100	50247	113	70	27	17	18	32	50	123
No. Served	0	0	113	70	27	17	18	32	50	123
No. Lost	2101	50247	0	0	0	0	0	0	0	0
No. Joined	0	0	0	0	0	0	0	0	0	0
No. Did Not Enter	0	0	0	0	0	0	0	0	0	0
No. In System	1	0	0	0	0	0	0	0	0	0
Avg Number In System	0.42	0	0.59	0.18	0.09	0.08	0.07	0.10	0.18	0.21
Avg time	0.1	0	2.43	2.03	1.71	1.27	1.33	2	1.29	2.03
Sigma Rating	0	0	6	6	6	6	6	6	6	6

Table 8. Scenario 2 Shortest Travel Distance Passenger Service Time (units in minutes)

Name	% Free	% Demand	% Transfer	% Loaded	% Stop	% Blocked	Distace	Loads
Lift	54.2	0	0	45.8	0	0	12267	360

Table 9. Scenario 2 Shortest Travel Distance Elevator Results (units in minutes)

Scenario 3 Most Different Destinations (MDD)

In this scenario the control logic governing the movement of the elevator from the SD and STD models was modified so that after serving all the passengers at a floor and with no more calls for the elevator at the same floor, the elevator will go to the next nearest floor by finding the different destination calls from that floor. i.e. where most passengers want to go on different floors rather than going to the same floor. That scenario acts like a destination call elevator in which a passenger presses the destination button before getting into the carriage. At some point, the elevator can skip some floors due to fewer passengers found for different possible destinations. Again, a number of loops, logical if conditions and user defined functions are used to control the elevator movement in the model. All the working details of these functions are the subject of a later publication. Tables 10, 11 and 12 show slight improvements in the results after the simulation is run for the same amount of time, i.e. 8.5 hours.

Name	QIN(1)	QIN(2)	QIN(3)	QIN(4)	QIN(5)
Total In	227	72	90	125	164
Total Out	227	72	90	125	164
Now In	0	0	0	0	0
Max	17	8	6	6	5
Min	0	0	0	0	0
Avg Size	0.75	0.05	0.07	0.06	0.29
Avg Time	1.3	0.28	0.3	0.2	0.66

Table 10. Scenario 3 Most Different Destination Elevator Call Waiting Time (units in minutes)

Name	Signal	Check LiftState	G_4th	G_3rd	G_2nd	G_1st	First_G	Second G	Third_G	Fourth G
No. Entered	2100	50247	113	70	27	17	18	32	50	123
No. Served	0	0	113	70	27	17	18	32	50	123
No. Lost	2101	50247	0	0	0	0	0	0	0	0
No. Joined	0	0	0	0	0	0	0	0	0	0
No. Did Not Enter	0	0	0	0	0	0	0	0	0	0
No. In System	1	0	0	0	0	0	0	0	0	0
Avg Number In System	0.42	0	0.52	0.18	0.13	0.10	0.07	0.10	0.13	0.19
Avg time	0.1	0	2.40	2.00	1.65	1.17	1.30	2	1.22	1.93
Sigma Rating	0	0	6	6	6	6	6	6	6	6

Table 11. Scenario 3 Most Different Destination Passengers Service Time (units in minutes)

Name	% Free	% Demand	% Transfer	% Loaded	% Stop	% Blocked	Distace	Loads
Lift	54.2	0	0	45.8	0	0	12152	360

Table 12. Scenario 3 Most Different Destination Elevator Results (units in minutes)

6 DISCUSSION OF SCENARIO RESULTS

It is evident that the data set used to drive the simulation model and the logic that controls the elevator movements will strongly reflect the behaviour of the model and the subsequent results. However, recognising the short simulation runs and limited iterations it is probably inappropriate to form trends without good confidence levels. Further work will address these shortcomings.

Statistics show that average waiting times are smaller in scenario 3 in comparison with the other scenarios. If average waiting time is considered an important measure of performance, then scenario 3 has the best results.

The maximum waiting time is often the most important parameter, as this will lead to worst-case conditions that are essentially what simulation is all about. From Table 11, scenario 3 appears to have the lowest range of maximum waiting times. This is probably explained by the fact the data is skewed

towards more passengers making elevator calls on the ground floor. Overall, on balance, tables 11, 12 and 13 indicate that scenario 3 is more cost effective, and in reality this could also lead to minimum operating costs.

CONCLUSIONS

The use of a visually interactive simulator has been shown to effectively and dynamically simulate the behaviour of a single elevator system. The technique of representing passengers as a series of entities enables the use of high quality animation in the simulation, which improves the display at the human/computer interface. A simple elevator control strategy has been developed and simulated using Witness software with reasonable success. Variants of the simple SD were devised, developed and simulated producing interesting results. However, it became evident that the data set obtained which drives the model affects the simulation results in a very serious way.

The EDS became the test-bed for the experiment and enabled dispatching scenarios to be dynamically evaluated and compared. During this work, it became clear that using an elevator call strategy is very individual, user and purpose dependant. The dispatcher algorithm for the elevator decision-making would need to be chosen on the nature, passenger flow and kind of building the elevator is being used for.

In this work, we were unable to include artificial intelligence techniques for the decision making within the implemented scenarios. The decision making module for example could not consider the distances between (or among) the floors requiring calls.

Clearly, a well-developed simulation model for a particular building in the construction design phase can save many unexpected expenses, especially more when we can predict and alter the volume of traffic.

The developed simulation model is designed strictly for the specific building with one ground floor and four floors above. Therefore, the developed model has one disadvantage: it is not parametrically built. The number of floors could not be a variable quantity. One possible solution to this problem is to develop a universal parameterized floor in a sub model, or more properly to use the Witness template and design user-defined elements.

The performance of any elevator dispatching system operating in high volume buildings will be increasingly important to building management as passengers come to expect higher service levels in the facilities being provided by modern elevators, having little regard for the complexity of the tasks involved in optimising the decision making.

7 FUTURE WORK

The results and overall experience of this preliminary study have provided the necessary stimulus to continue the work to include high traffic volume, consider other scenarios, i.e. maximum floor calls, maximum passenger waiting, longest waiting time and to extend the experimentation to achieve 95% confidence levels.

Further work will involve optimisation for computational efficiency, incorporate intelligent decision making techniques, increase the problem domain to deal with multiple and banked elevators, and quantify the benefits gained from these methods.

Furthermore, some optimisation could be performed. The criteria function could consider the average waiting time and the time of the passenger within the system. For economic evaluation, we should also consider operation costs. The criteria function can balance both of these factors (maximize the transport quality and minimize the costs).

ACKNOWLEDGMENTS

This paper was prepared with the support of the Internal Science Foundation of the University of West Bohemia SGS-2018-031.

REFERENCES

[Ahn 2017] Ahn, S., Lee, S., Bahn, H., A smart elevator scheduler that considers dynamic changes of energy cost and user traffic, Integrated Computer-Aided Engineering, 2017, vol. 24, no. 2, pp. 187-202, ISSN 1069-2509

[Al-Sharif 2018] Al-Sharif, L., Yang, Z. S., Hakam, A., Abd Al-Raheem, A., Comprehensive analysis of elevator static sectoring control systems using Monte Carlo simulation. Building Services Engineering Research and Technology, January 2018, ISSN 0143-6244

[Berbeglia 2010] Berbeglia G., Cordeau J. F., and Laporte G., Dynamic pickup and delivery problems. European Journal of Operational Research, April 2010, Volume 202, Issue 1, pp 8-15, ISSN 0377-2217

[Cortes 2004] Cortes, P., Larraneta, J. and Onieva, L.: Genetic algorithm for controllers in elevator groups: analysis and simulation during lunch peak traffic. Applied Soft Computing, May 2004, Volume 4, Issue 2, pp. 159-174, ISSN 1568-4946

[Lanner 2018] Lanner Group, Technology Witness Horizon [1.6.2018]. Available from https://www.lanner.com/en-us/technology/witness-simulation-software.html

[Lee 2009] Lee, Y., Kim, T. K et al.: Performance analysis of an elevator system during up-peak. Mathematical and Computer Modelling, February 2009, Volume 49, Issues 3-4, pp. 423-431, ISSN 0895-7177

[Nagatani 2015] Nagatani, T., Complex motion induced by elevator choice in peak traffic. Physica A: Statistical Mechanics and its Applications, May 2015, Volume 436, pp 159-169, ISSN 0378-4371

[Robert 1988] Robert M. C., and Abrego E.: Coincident call optimization in an elevator dispatching system, 1988, Westinghouse Electric Corp. U.S. Patent No. 4 782 921.

[Sutton 1998] Sutton, R. S. and Barto, A. G.: Reinforce ment Learning: An Introduction: Elevator Dispatching. Massachusetts: The MIT Press, 1998, ISBN 0-262-19398-1

[Tanaka 2005] Tanaka, S., Uraguchi, Y., and Araki, M.: Dynamic optimization of the operation of single-car elevator systems with destination hall call registration. European Journal of Operational Research, December 2005, Volume 167, Issue 2, pp. 550-573, ISSN 0377-2217

[Tebbenhof 2000] Tebbenhof, A., and Dekker, R.: Econometrics in the elevator, International Journal: The Medium for Econometric Applications, 2000, Volume 9, Issue 3, pp. 26-30, ISSN 1389-9244

[Thangavelu 1989] Thangavelu and Kandasamy: Queue based elevator dispatching system using peak period traffic prediction, 1989, Otis Elevator Company. U.S. Patent No. 4 838 384.

[Yuan 2008] Yuan X., Busoniu L., and Babuska R., Reinforcement Learning for Elevator Control, 17th International Federation of Automatic Control, In: IFAC world conference, Seoul, South Korea, 2008, July 6-11
[Zahang 2013] Zhang, J., Zong, Q., Group Elevator Peak Scheduling Based on Robust Optimization Model, Advances in Electrical and Computer Engineering 2013, Volume 13, Number 3, 2013, pp. 51-58, ISSN 1582-7445
[Zheng 2013] Zheng, Y. P., Zhang, Z. T., Xu, H., (2013) A novel intelligent elevator group control algorithm based on corridor passenger detection and tracking, In: 2nd International Conference on Measurement, Instrumentation and Automation, ICMIA 2013, Guilin, China, 2013, Volume 336-338, 2013, pp 815-819, ISBN 978-303785751-9

CONTACTS:

Jamil Ahmad
Manchester Metropolitan University, School of Engineering
John Dalton Building, Chester St, Manchester M1 5GD, UK
Tel.: +44 (0)785393914, e-mail: Jamilahmad@hotmail.com

Dr. Muhammad Latif
Manchester Metropolitan University, School of Engineering
John Dalton Building, Chester Street, Manchester M1 5GD, UK
Tel.: +44 (0) 161 247 6264, e-mail: M.Latif@mmu.ac.uk

Ing. Petr Horejsi, Ph.D.
University of West Bohemia,
Department of Industrial Engineering and Management
Univerzitnií 8, 301 00 Pilsen
Tel.: +420 37763 8495, e-mail: tucnak@kpv.zcu.cz