Hindawi Publishing Corporation
Abstract and Applied Analysis
Volume 2014, Article ID 530619, 9 pages
http://dx.doi.org/10.1155/2014/530619

Research Article

Z Specification of Gate and Apron Control Management at Airport

Nazir Ahmad Zafar,[1] Fahad Alhumaidan,[1] and Sher Afzal Khan[2]

[1]*College of Computer Sciences and IT, King Faisal University, Alahsa 31982, Saudi Arabia*
[2]*Department of Computer Sciences, Abdul Wali Khan University, Mardan 23200, Pakistan*

Correspondence should be addressed to Nazir Ahmad Zafar; nazafar@gmail.com

Received 4 March 2014; Revised 15 April 2014; Accepted 15 April 2014; Published 31 December 2014

Academic Editor: Saeed Islam

Modelling of an air traffic control (ATC) system is an open issue and has become a challenging problem due to its complexity and increase of traffic at airports and in airspace. Consequently, automated ATC systems are suggested to improve efficiency ensuring the safety standards. It is reported that the number of collisions that occurred at airports surface is three times larger than in airspace. Further, it is observed that gates and aprons congestions cause significant delays at airports; hence, effective monitoring and guidance mechanisms are required to control ground air traffic. In this paper, formal procedure of managing air traffic from gate to enter in the active area of airport for taxiing is provided using Z notation. An integration of gate and apron controllers is described to manipulate the information for correct decision making and flow management. Graph theory is used for representation of airport topology and appropriate routs. In static part of the model, safety properties are described in terms of invariants over the critical data types. In dynamic model, the state space is updated by defining pre- and postconditions ensuring the safety. Formal specification is analysed using Z/Eves tool.

1. Introduction

Air traffic control (ATC) system is a highly complex and safety critical system because its failure may cause a huge loss in terms of deaths or financial losses. It requires state-of-the-art techniques for development of ATC systems. Because of a large increase in movement of population and consequently a significant increase in capacity of air traffic [1], next generation ATC systems are suggested to improve efficiency by not compromising safety standards [2–5]. Although an automated support to ATC system is available nowadays, still it is heavily dependent upon human interaction causing delays and accidents due to failure of communication in decision making [6, 7]. Therefore, developing an ATC system enabling aircraft to move at airport and freely fly in the air is a current issue [8]. Further, we believe that modelling of safe and efficient ATC system will remain an open research problem because of its complexity.

The ATC control can be divided into two categories, that is, in-air and airport control systems. The airport surface environment has historically been more dangerous than the airspace. For example, the number of collisions that occurred at the airport surface is three times more than the collisions in the airspace [9]. It means we need effective monitoring and automated guiding systems to control ground air traffic at gates, apron area, taxiways, and runway intersections. Apron area is used for preflight activities, for example, parking, waiting, and maintenance. Gate, apron, and ground controllers are main components for airport surface management; however these are very less focussed on by the scientific community addressing ATC system [10]. Objective of apron controller is to share information, communication constraints, and priorities of aircrafts among the various operators and controllers in addition to providing an active decision support functions for route predictions [11].

In this paper, formal procedure of managing air traffic at airport from gate to taxiway is provided using Z notation. The Z is applied for formal description of the system under hand because of its abstract mathematical features and rigorous computer tool support [12]. It is noted that according to European electrotechnical standards, the use of formal methods is recommended to achieve a required level of confidence in

development of safety critical system [13]. In the algorithm, an integration of gate and apron controllers is defined by sharing the information required for managing the air traffic from gate to taxiway. The airport surface is divided into small enough blocks. Graph theory is used for the description of airport and routs. For mapping of real world airport to graph model, block is represented as a node and connectivity between the blocks is represented as an edge. The safety properties are described in terms of invariants over the data types carrying the critical information in the static model. For example, it is assured that there must exit, at most, one aircraft in one region preventing collision at the airport surface. The traffic sequence is updated by defining pre-/postconditions for defining safety properties in the dynamic part of the model. Formal specification is analysed using Z/Eves tool.

There exists much work on modelling of ATC system; some important work is discussed in Section 2. In most of the work safety criteria are developed by testing through simulation but unfortunately this approach is lacking in verifying the correctness of ATC systems. This is because the number of simulations increases exponentially to provide a required level of confidence in complex systems. Moreover, when a modification is needed the complete set of simulations must be reconducted to ensure that the change did not compromise with the defined safety and reliability. Therefore, it is required to apply formal approaches for modelling ATC system which has motivated us for embarking research in this direction.

The rest of the paper is organized as follows. In Section 2, most relevant work is critically analysed. In Section 3, problem statement and formulation is presented. Formal algorithm is described in Section 4. Model analysis is done in Section 5. Finally, conclusion and future work are discussed in Section 6.

2. Related Work

There exists much work on ATC system; most relevant work is discussed here. For example, in the existent work, a modified minimum cost and maximum-flow genetic algorithm is developed to maximize the ground airport capacity [14]. The work is interesting in which the online scheduling of aircrafts considering airport capacity is analysed but results are tested and not fully verified. NASA has developed the surface management system that provides information to federal aviation authority controllers and air carriers to manage the airport [15]. In this paper, it is focussed on prediction of future airport surface characteristics, for example, taxiing and take-off time for allocation of departure runways. In another work, traffic limitations enhancement is developed by advanced modelling capability for simulating airport surface aircraft movement as a case study [16]. In this work, a probabilistic timed automata model is developed under certain assumptions for describing operators' behaviours. A limited number of system's properties are expressed in probabilistic real time computation tree logic. Taxi operation and real-time planning function is developed to address taxi operation uncertainties based on real data in [17]. However, in this work, only a part of the planning function is evaluated using scenarios based approach. In [18], unnecessary taxi time is reduced by reducing taxiing process uncertainties and aircraft queuing at the runway. Planning function is developed for optimizing the traffic flow by real time sequencing the departure traffic at the gate. In this work, although a process is described, no proper algorithm is provided. In another interesting and most relevant work, a model for estimating the ramp congestion delay is reduced by employing managed gate operation (MGO) tool [19]. Again a detailed procedure is described in this work; however, no validation or verification approach is provided to prove its correctness. In [20], PRISM tool is used to verify and analyse the properties of ATC system using timed automata. Multiagent approach is applied in modelling of air traffic flow management (ATFM) in [21]. The objective of this research is to show applicability of agent-based simulation. In another work of NASA, a collaborative air traffic flow is developed using multiagent simulation technique. Several strategies were used to select various routes increasing complexity of the system [22]. A fusion of intelligent computing is studied for development of a new tactical system for ATFM using advantages of the meta-level control approach [23]. In another application, intelligent computing models are claimed for ATFM as presented in [24]. Due to the increase of air traffic density and relatively limited number of airways, the future solution for optimal airspace and safe air traffic control is proposed in [25]. A protocol for aircraft conflict resolution is proposed in [26–28] for information horizon in which the communication range of an aircraft is finite. A predictable system is developed to achieve free flight to choose an optimal path minimizing fuel consumption and delay time rather than using predefined flight schedules in [29–31]. This is very good effort for developing free flight ATC systems. The performance of conflict detection and resolution is presented in [32] on estimation of aircraft state space. Further, satellite based communication systems have been suggested in current advances to consider the free flight concept for the future ATC systems [33, 34]. Preliminary results of our research are reported in [35–39]. Other similar work can be found in [40–45].

3. Problem Statement and Formulation

Ensuring safety and increasing efficiency of an ATC system have become a central issue due to increase of air-traffic and introduction of new technologies [46]. The primary objective of ATC system is to provide a safe and efficient flow of air traffic [47]. The safe operation is made possible by sharing information and developing effective communication through various systems to keep standard separation between aircrafts. There are various ATCs responsible for monitoring aircrafts from gate (departing) to gate (destination).

The departing of aircraft begins with the pushback procedure. After checking the availability of the apron area, the aircraft is taxied to the active area of the airport. Air traffic control tower is responsible for giving permission to enter from apron area to taxiway. The route sequence from gate to taxiway is issued by the apron controller having communication with tower controller. The control tower is responsible for assigning the departure runway and the taxi route to reach the assigned runway. The runway assignment

decision is based on various factors, for example, gate, pilot preference, and traffic size. After reaching the runway, the aircraft is put into the departure queue and then takes off after the final permission. To complete this whole procedure the surface management system collects information from various sources and then flight plan including gate leaving, apron area occupation, taxiway entering, departure time, and runway assignment route is prepared.

It is noted that the departure runway might be assigned after pushback request is received. This is because the departure runway can only be predicted if we are able to predict the time between pushback and the takeoff. Various factors are involved assigning departing aircraft to a runway. For example, how the airport runways are configured for arriving and departing traffic which totally depends on the airport. From gate to take-off, few major reasons for delay in a flight are as follows: (i) unoptimized calculation of departure sequence because of various airport states causing state space explosion, (ii) inefficient push back procedures because of dynamic change at airport, (iii) revisiting route sequences because of accommodating priorities, and (iv) apron controllers do not have full support to accommodate airline priorities because of lack of automated functioning support. As a result, it causes an irregular operation of apron and gate controllers.

In our model, a formal approach is developed to expedite airport traffic from gate to apron and taxiways through integration of various controllers. It is noted that detailed information, for example, weather conditions, wind speed, and direction, which may change the runway configuration in reality is not considered in our model in defining route sequence. Further, aircraft type and weight are also not considered. In this way, same length of route is assumed for every type of an aircraft in our model. In the model, two separate queues are maintained, each one for entering apron area and taxiways. Aircraft position, taxiway location, and runway are provided by the surface surveillance system which is usually integrated with GIS. Such integration issues are also out of the scope of this research.

In static part of the model, the airport surface is divided into different regions and then transformed into a graph. In the transformation, a small unit (block) of airport surface is represented as a node and connectivity between two blocks is assumed as an edge in the graph. As we have supposed different routes for take-off and landing procedures in our model, hence, the resultant model is a directed graph relation. However, a gate can be used for both incoming and departing flights. The objective is to find and assign an appropriate route from gate to taxiing an aircraft. Assigning gates and defining aprons and connectivity from apron to active area for taxiing are three main activities addressed in the dynamic part of system.

In the operational system, initially an aircraft sends a pushback request to the gate controller. The pushback operation is executed after having a clearance from the apron controller. Next, the aircraft sends a request for taxi clearance. The clearance is awarded to the aircraft for taxiing after communication of apron controller and the air traffic control tower.

4. Formal Modelling Using Z Notation

Safety and efficiency are two core requirements in safe and normal operation of an ATC system. Safety requires a well-defined sequence of patterns whereas efficiency needs expeditious movement of aircrafts. In this section, formal procedure of aircrafts movement from gate to active area for taxiing is described. Formal rules are defined to prevent collisions and expedite the flow of traffic by maintaining a queue of aircrafts using Z notation.

4.1. Static Model. First of all, formal specification of airport surface is described based on the graph relation. The smallest surface unit of the airport is represented by a *Block* which is node in the graph relation. The connectivity of two blocks is represented by a link which is an edge in the graph. The ordered pair (u, v) in the edge-set means an aircraft can move from node u to node v:

$$[Block]; Links == \{x, y : Block \mid x \neq y \bullet (x, y)\}.$$

Formal specification of the graph relation is described by the schema *Graph*. The schema consists of two parts divided in horizontal form, definition, and predicate parts. In first part of the schema, variables definitions are given and invariants are described in predicate part of the schema. The schema consists of two components, that is, block-set and link-set. The block-set is defined as a finite power set of *Block*. The link-set is a finite power set of *Links* which is in fact the set of all the possible edges of the graph relation.

In predicate part, it is stated that both ends of any edge are nodes which is a natural constraint in graph relation. Further, every block is an end of an edge; that is, there is no isolated block. Finally, for any two blocks, there is a path in the graph relation because it is supposed that it is possible to move from one block to any other block at the airport surface.

Graph

$blocks: \mathbb{F}\, Block$

$links: \mathbb{F}\, Links$

$\forall b1, b2: Block \mid (b1, b2) \in links \bullet b1 \in blocks \land b2 \in blocks$

$\forall b: blocks \bullet \exists b1, b2: Block \mid (b1, b2) \in links \bullet b = b1 \lor b = b2$

$\forall b1, b2: blocks \bullet \exists path: \text{seq } Block \bullet \forall i: \mathbb{N} \mid i \geq 1 \land i < \#path \bullet (path\ i, path\ (i + 1)) \in links$

Passenger gate is represented by the *Gate* schema constituted by two components, that is, gate identifier and gate state. The state variable has values, clear or occupied. The set of gates is defined by the partial function *gates* from gate identifier to *Gate* schema. The *Gates* schema is a set of all the gates at airports which can be assigned to an aircraft.

$State ::= CLEAR \mid OCCUPIED$

Gate

$gateid: Block$

$gatestate: State$

Gates

$gates$: $Block \nrightarrow Gate$

$\forall gid$: $Block$; $gate$: $Gate \mid (gid, gate) \in gates$
$\bullet\ gid = gate \cdot gateid$

Apron area is used for preflight activities including parking, waiting, and maintenance. As there is an association relationship between apron area and gate, hence, apron is needed to be defined as a separate entity which consists of apron identifier and its state. The identifier is assumed as a block. The set of aprons is a partial function from apron identifier to apron schema. In predicate part, it is stated that for every apron identifier *apid* and schema *apron* the ordered pair *(apid, apron)* is in the domain of *aprons* function.

Apron

$apid$: $Block$
$state$: $State$

Aprons

$aprons$: $Block \nrightarrow Apron$

$\forall apid$: $Block$; $apron$: $Apron \mid (apid, apron) \in aprons$
$\bullet\ apid = apron \cdot apid$

Taxiway is a path on an airport connecting an apron area to a runway through various other services. In our model, taxiway is defined as a schema consisting of taxiway identifier and a sequence of blocks defining a well-defined path.

[TaxiwayId]

Taxiway

$taxiwayid$: $TaxiwayId$
$path$: seq $Block$

The *Taxiways* schema contains two components, namely, *taxiways* and *taxiingA*. The first one, *taxiways,* is a function from taxiway identifier to *Taxiway*. The second one, *taxiingA*, is a partial function from taxiway identifier to aircraft identifier occupying the taxiway. In predicate part, it is stated that the domain of *taxiingA* is contained in the domain of *taxiways* function.

[AircraftId]

Taxiways

$taxiways$: $TaxiwayId \rightarrow Taxiway$

$taxiingA$: $TaxiwayId \nrightarrow AircraftId$

$\forall tid$: $TaxiwayId$; $taxiway$: $Taxiway \mid (tid, taxiway) \in taxiways$
$\bullet\ tid = taxiway \cdot taxiwayid$
dom $taxiingA \subseteq$ dom $taxiways$

The airport topology consists of four schemas defining graph, gates, aprons, and taxiways. In predicate part, it is stated that every block in the domain of gates and aprons functions belongs to the node-set. The intersection of domains of gates and aprons functions is empty. Further, it is stated that every block of gate and apron area is connected to a taxiway. The paths in taxiways are represented as a sequence of blocks satisfying the invariants of the connectivity relation. It is stated that every element of a path sequence is a block in the graph relation. Any two consecutive elements in path sequence constitute an edge in the graph relation.

AirportTopology

$Graph$; $Gates$; $Aprons$; $Taxiways$

$\forall$b: $Block$; $gate$: $Gate \mid (b, gate) \in gates$
$\bullet\ gate \cdot gateid \in blocks$

$\forall b$: $Block$; $apron$: $Apron \mid (b, apron) \in aprons$
$\bullet\ apron \cdot apid \in blocks$

dom $gates \cap$ dom $aprons = \varnothing$

$\forall tw$: ran $taxiways$

$\bullet\ \exists path1$: seq $Block \mid tw \cdot path = path1$

$\bullet\ \forall i$: $\mathbb{N} \mid i \in$ dom $path1 \wedge i \in 1, \ldots, \#path1 - 1$

$\bullet\ (tw \cdot path\ i, tw \cdot path\ (i + 1)) \in links$

$\forall b1$: dom $gates \cup$ dom $aprons$

$\bullet\ \exists tw$: ran $taxiways \bullet \exists b2$: ran $tw \cdot path$

$\bullet\ \exists route$: seq $Block \bullet \forall i$: $\mathbb{N} \mid i \geq 1 \wedge i <$ # $route$

$\bullet\ (route\ i, route\ (i + 1)) \in links$

An aircraft is specified by a schema *Aircraft* which consists of two components, that is, aircraft identifier and its safe area. The set of all permissible aircrafts at the airport is defined as a mapping from aircraft identifier to *Aircraft*. In predicate part of the *Aircrafts* schema, it is stated that an intersection of safe areas of any two aircrafts is always empty.

Aircraft

$aircraftid$: $AircraftId$
$safeArea$: seq $Block$

Aircrafts

$aircrafts$: $AircraftId \rightarrow Aircraft$

$\forall aid$: $AircraftId$; acr: $Aircraft \mid (aid, acr) \in aircrafts$
$\bullet\ aid = acr \cdot aircraftid$
$\forall ac1, ac2$: ran $aircrafts \bullet ac1 \cdot safeArea \cap ac2 \cdot safeArea = \varnothing$

The gate controller defined below consists of four components. The first one is *Gates* schema which is already defined. The second component is *gatesR* representing the aircrafts which have requested a gate. The *gatesR* component is defined as a sequence type to provide the service on first come and first serve basis. The *gatesA* is the third component representing mapping from aircraft identifier to gate. The last

one is *pushbackR* which is the set of aircrafts which have requested for pushback from the gate.

It is mentioned that relationship among all the components of the gate controller is defined in terms of properties. To capture invariants for completeness of the specification, each component of the gate controller was selected and then it identified any relationship, if exists, with the rest of the components.

GateController

$Gates$

$gatesR$: seq $AircraftId$

$gatesA$: $AircraftId \nrightarrow Gate$

$pushbackR$: $\mathbb{F}\ AircraftId$

$\forall gate$: ran $gates$

$\bullet\ gate \in \text{ran}\ gatesA \Rightarrow gate \cdot gatestate = OCCUPIED \wedge$

$gate \notin \text{ran}\ gatesA \Rightarrow gate \cdot gatestate = CLEAR$

ran $gatesR \cap$ dom $gatesA = \varnothing$

$pushbackR \subseteq$ dom $gatesA$

Invariants are as follows: (i) if a gate is assigned to an aircraft, it must be in the occupied state otherwise it is in the clear state; (ii) if a gate is assigned to an aircraft, the aircraft cannot be in the list of aircrafts which have requested a gate. If an aircraft has requested a gate, it cannot be in the list of aircrafts which are assigned a gate; (iii) if an aircraft has requested pushback, then aircraft must be in the list of aircrafts which are assigned the gates.

It was possible to specify gate and apron controllers using the same schema; however, we have defined it separately because of the simplicity of the model. The apron controller consists of *aprons*, set of aircrafts which have requested for pushback clearance *pushbackC*, sequence of aircrafts which are in apron area *apronQ*, and sequence of aircrafts which have requested for taxiing *taxiingR*. Formal specification of the apron controller is described below following the invariants.

ApronController

$Aprons$

$pushbackC$: $\mathbb{F} AircraftId$

$apronQ$: seq $AircraftId$

$taxiingR$: seq $AircraftId$

$\forall aid$: $pushbackC \bullet aid \notin \text{ran}\ apronQ \cup \text{ran}\ taxiingR$

$\forall aid$: ran $apronQ \bullet aid \notin pushbackC$

$\forall aid$: ran $taxiingR \bullet aid \in \text{ran}\ apronQ \wedge aid \notin pushbackC$

Invariants are as follows: (i) any aircraft which has requested pushback clearance cannot be in the list of aircrafts in the apron area; (ii) if an aircraft is in the apron area, it has not requested the pushback clearance; (iii) if an aircraft has requested taxiing, it is in the apron area but not in the list of aircrafts which has requested pushback clearance.

4.2. Dynamic Model. Formal specification of operations required for moving aircrafts from gate to taxiways is described in this section. The model is a part of the take-off procedure from gate to taxiing for updating state space of the airport. There are three main facilities, namely, gates, aprons, and taxiways, which are managed by gate and apron controllers. At first, an aircraft is entered from gate to apron area by communication of gate and apron controller. After an aircraft is entered from apron area to taxiway, it is controlled by the local controller.

First of all, an operation for gate request is defined below. An aircraft sends a request for gate to the gate controller by showing its identity. After verifying the identity, the gate controller accepts the request and adds the aircraft in the waiting list *gatesR*. The operation is described by the schema *RequestGate* which contains $\Xi AproncController$, $\Delta GateController$, and aircraft identifier *aid?* as inputs. The state of gate controller is updated by verifying the properties as pre- and postconditions. It is noted that postcondition must be satisfied after the successful execution of the operation. The symbol Ξ used in the schema shows that state of apron controller is not changed. The symbol Δ shows that state of gate controller is changed. The symbol *?* after *aid* variable represents that it is an input variable. The schema components are put in first part and pre-/postconditions are described in second part of the schema.

RequestGate

$\Xi ApronController$

$\Delta GateController$

$aid?$: $AircraftId$

$aid? \in \text{ran}\ apronQ$

$aid? \notin \text{ran}\ gatesR$

$gatesR' = gatesR^\frown\langle aid?\rangle$

$gatesA' = gatesA$

$pushbackR' = pushbackR$

Pre-/postconditions are as follows: (i) the requesting aircraft must be in the apron area (*apronQ*); (ii) the aircraft is not in the list of waiting list (*gatesR*); (iii) if the above conditions are satisfied, then aircraft is added to the waiting list; (iv) the other two variables of gate controller are unchanged. The symbol "$'$" decorating a variable is used for its new state.

Formal definition of the gate assignment operation is provided by the *AssignGate* schema. The schema contains three components, namely, $\Delta GateController$, aircraft identifier, and gate, as inputs in first part of the schema. The gate is assigned by the gate controller in terms of pre- and postconditions in the predicate part of the schema.

AssignGate

$\Delta GateController$
$aid?: AircraftId$
$gate?: Gate$

$aid? \in \text{ran } gatesR$
$aid? \notin \text{dom } gatesA$
$gate? \in \text{ran } gates$
$gate? \notin \text{ran } gatesA$
$gatesA' = gatesA \cup \{(aid? \mapsto gate?)\}$
$gatesR' = gatesR$
$pushbackR' = pushbackR$

Pre-/postconditions are as follows: (i) the aircraft must be in the waiting list of aircrafts; (ii) the aircraft is not assigned a gate; (iii) the input gate belongs to the valid list of gates; (iv) the gate is not assigned to any other aircraft; (v) if the above conditions are satisfied, then aircraft is assigned the gate; (vi) the other two variables *gateR* and *pushbackR* of gate controller are unchanged.

The pushback request procedure is denoted by the *PushbackRequest* schema. The schema consists of $\Delta ApronController$, $\Delta GateController$, and aircraft identifier. The schema definition is given below following pre-/postconditions for updating state space of gates.

PushbackRequest

$\Delta ApronController$
$\Delta GateController$
$aid?: AircraftId$

$aid? \in \text{dom } gatesA$
$aid? \notin pushbackR \cap pushbackC$
$pushbackC' = pushbackC \cup \{aid?\}$
$pushbackR' = pushbackR \cup \{aid?\}$
$gatesA' = gatesA$
$gatesR' = gatesR$
$apronQ' = apronQ$
$taxiingR' = taxiingR$

Pre-/postconditions are as follows: (i) the aircraft must be assigned a gate before sending a pushback request; (ii) the aircraft neither has requested for pushback nor has a pushback clearance; (iii) the aircraft is added in the pushback request of gate controller and pushback clearance list of apron controller; (iv) the other variables *gatesA* and *gatesR* of gates controller and *apronQ* and *taxiingR* of apron controller are unchanged.

The pushback procedure is denoted by *Pushback* schema consisting of five components, namely, $\Delta GateController$, $\Delta ApronController$, aircraft identifier, apron identifier, and apron, as given below. The schema definition is given below following the pre-/postconditions.

Pushback

$\Delta GateController$
$\Delta ApronController$
$aid?: AircraftId$
$apid?: Block$
$apron?: Apron$

$aid? \in \text{dom } gatesA$
$aid? \in pushbackR \cap pushbackC$
$(apid?, apron?) \in aprons$
$gatesA' = \{aid?\} \lhd gatesA$
$pushbackR' = pushbackR \setminus \{aid?\}$
$pushbackC' = pushbackC \setminus \{aid?\}$
$aprons' = aprons \cup \{(apid? \mapsto apron?)\}$
$apronQ' = apronQ^\frown \langle aid? \rangle$
$gatesR' = gatesR$
$taxiingR' = taxiingR$

Pre-/postconditions are as follows: (i) the aircraft must be assigned a gate; (ii) it is in the lists of pushback requests and pushback clearance; (iii) the aircraft is removed from the gate list; (iv) the aircraft is removed from the pushback and clearance lists; (v) the aircraft is allowed to enter the apron area; (vi) the rest of the variables of gate and apron controllers are unchanged.

The taxi request procedure is defined below by using *TaxiRequest* schema consisting of $\Delta ApronController$ and aircrafts identifier. The schema definition is given below following the informal description.

TaxiRequest

$\Delta ApronController$
$aid?: AircraftId$

$aid? = apronQ\ 1$
$aid? \notin \text{ran } taxiingR$
$taxiingR' = taxiingR^\frown \langle aid? \rangle$
$apronQ' = apronQ$
$pushbackC' = pushbackC$

Pre-/postconditions are as follows: (i) the aircraft which has requested for taxiing is the first one in the queue in apron area; (ii) the aircraft does not belong to the list of aircrafts waiting for taxiing; (iii) the aircraft is added in the list of aircrafts waiting for taxiing; (iv) the other two variables *apronQ* and *pushbackC* of apron controller remained unchanged.

Finally, formal procedure of leaving the apron area for an aircraft and entering into taxiway is described using *EnterTaxi*

schema which consists of $\Delta ApronController$, $\Delta Taxiways$, aircraft, taxiway identifier, and taxiway. The aircraft is removed from the waiting list of aircrafts by using the filter "$\upharpoonright$" operation.

EnterTaxi

$\Delta ApronController$

$\Delta Taxiways$

$acr?: Aircraft$

$tid?: TaxiwayId$

$taxiway?: Taxiway$

$acr? \cdot aircraftid \in \text{ran } taxiingR$

$(tid?, taxiway?) \in taxiways$

$acr? \cdot aircraftid = apronQ\ 1$

$taxiingR'$

$= \{i: \mathbb{N}; aid: AircraftId \mid i \in \text{dom } taxiingR \wedge (i, acr? \cdot aircraftid) \notin taxiingR \wedge aid \neq acr? \cdot aircraftid \bullet i\} \upharpoonright taxiingR$

$apronQ'$

$= \{i: \mathbb{N}; aid: AircraftId \mid i \in \text{dom } apronQ \wedge (i, acr? \cdot aircraftid) \notin apronQ \wedge aid \neq acr? \cdot aircraftid \bullet i\} \upharpoonright apronQ$

$pushbackC' = pushbackC$

$taxiingA' = taxiingA \cup \{(tid? \mapsto acr? \cdot aircraftid)\}$

$taxiways' = taxiways$

Pre-/postconditions are as follows: (i) the aircraft must have taxiing permission; (ii) the aircraft which has requested for taxiing is the first one in the queue in the apron area; (iii) after the aircraft has taxied, it is removed from the list of aircrafts having permission for taxiing and from the apron area; (iv) the rest of the variables of apron controller remained unchanged.

5. Model Analysis

In this section, formal analysis of the specification is provided using Z/Eves toolset. As we know, there does not exist any real computer tool which may assure complete correctness of formal specification. That means even if the formal specification is written well, it may cause potential errors. Hence, an art of writing formal specification does not provide any guarantee about correctness of the model. If the formal specification of a system is analysed with a computer tool, it improves the confidence by identifying errors if it exists in the model.

The Z/Eves is a powerful tool used here for analysing the formal specification of a part of the air traffic control system responsible for aircraft movement from gate to the active area for taxiing. Some schemas of the formal model are checked to be correct while the others are proved by reduction technique available in the tool.

Table 1: Results of model analysis.

Schema Name	Syntax type check	Domain check	Reduction	Proof by reduction
Graph	Y	Y	Y*	Y
Gate, Gates	Y	Y	NA	Y
Apron, Aprons	Y	Y	NA	Y
Taxiway, Taxiways	Y	Y	NA	Y
AirportTopology	Y	Y	Y*	Y
Aircraft, Aircrafts	Y	Y	NA	Y
GateController	Y	Y	NA	Y
ApronController	Y	Y	NA	Y
RequestGate	Y	Y	Y*	Y
AssignGate	Y	Y	NA	Y
PushbackRequest	Y	Y	NA	Y
Pushback	Y	Y	Y*	Y
TaxiRequest	Y	Y	Y*	Y
EnterTaxi	Y	Y	Y*	Y

Summary of the results is provided in Table 1. In first column of the table, name of the schema is provided. The second column is used for syntax and type checking. The domain checks proofs in the tool guarantees the consistency of the formal specifications for axiomatic declarations. Domain checking is done in column 3. Proof by reduction is a technique in which equivalent simpler combinations of tactics is substituted. Reduction and proof by reduction are represented in columns 4 and 5, respectively. The symbol "Y" in the table indicates that all schemas are proved to be correct automatically. The symbol "Y" annotated with "*" shows that the schema is proved to be correct by reduction technique. The symbol "NA" in 4th column is used to mean that reduction is not required on the predicates and, hence, the formal specification is proved to be written well and meaningful.

6. Conclusion

In this paper, we have described a formal procedure for air traffic flow management from gate to taxiing in air traffic control (ATC) system. Initially, we have described fundamental components for description of the required system. The airport surface is represented using graph theory as a part of static model. We observed that graph model was an effective one for defining connectivity relation and appropriate taxing routs. Dynamic model is described for manipulating critical information based on the static model. Safety properties are described in terms of invariants over the components in the static model. Pre- and postconditions are used to define safety criteria in the operational system to avoid any unwanted situation. Z notation is applied, because of its rigorous and abstract nature, for formal analysis of this critical system.

We observed that the complexity of the ATC system was reduced by decomposing into its components. The use of schema structure in Z notation facilitated both in the static

and dynamic parts of the model. Systematic development from abstraction to detailed model made it easy to propose a simple and abstract model.

There exists much work on modelling of ATC system; however it needs more research to address next generation automated systems achieving the required level of safety and efficiency. The work of Michael and Steven is close to this in which gate management and ramp operations are analysed for reducing delay time, fuel burning, and other costs [48]. In their work, the approach is fairly conservative based on observations and results are not fully verified and established.

Various benefits describing formal specification of the system were observed. For example, modelling of component-based system provided us with a complete characterization at a higher level of abstraction. On the other hand, if the system was specified at a more detailed level, intuition may have been lost. Compositional approach enabled us to give reasoning about the components and subsequently the entire system. Further advantages of a formal model can be observed after refinement. The detailed model can be achieved after a series of refinements while guaranteeing the transformation of syntax and semantics rules.

A clear scope and set of assumptions were defined before producing a mathematical model of the system. It is mentioned that this formal model can be applied to an ATC system after a further refinement and analysis. This is because we have defined the properties based on the requirements of a real ATC system.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] J. Villiers, *ERASMUS—A Friendly Way for Breaking the Capacity Barrier*, ITA, 2004.

[2] H. Erzberger, "Transforming the NAS: the next generation air traffic control system," in *Proceedings of the International Congress of the Aeronautical Sciences*, 2004.

[3] H. Erzberger, "Automated conflict resolution for air traffic control," in *Proceedings of the 25th International Congress of the Aeronautical Sciences*, 2006.

[4] H. Erzberger and K. Heere, "Algorithm and operational concept for resolving short range conflicts," *Journal of Aerospace Engineering*, vol. 224, pp. 225–243, 2009.

[5] T. Farley and H. Erzberger, "Fast time air traffic simulation of a conflict resolution algorithm under high air traffic demand," in *Proceedings of the USA Europe ATM Seminar*, 2007.

[6] J. Hu, M. Prandini, and S. Sastry, "Optimal maneuver for multiple aircraft conflict resolution: a braid point of view," in *Proceedings of the 39th IEEE Confernce on Decision and Control*, vol. 4, pp. 4164–4169, December 2000.

[7] S. T. Shorrock and B. Kirwan, "Development and application of a human error identification tool for air traffic control," *Applied Ergonomics*, vol. 33, no. 4, pp. 319–336, 2002.

[8] N. E. Debbache, "Toward a new organization for air traffic control," *Aircraft Engineering and Aerospace Technology*, vol. 73, no. 6, pp. 561–567, 2001.

[9] W. Marshall and W. I. Joseph, "Airport movement area safety system," in *Proceedings of the IEEE Digital Avionics Systems Conference*, pp. 549–552, 1992.

[10] Y. Guo, X. Cao, and J. Zhang, "Constraint handling based multiobjective evolutionary algorithm for aircraft landing scheduling," *International Journal of Innovative Computing, Information and Control*, vol. 5, no. 8, pp. 2229–2238, 2009.

[11] G. J. Couluris, R. K. Fong, M. B. Downs et al., "A new modeling capability for airport surface traffic analysis," in *Proceedings of the IEEE/AIAA 27th Digital Avionics Systems Conference (DASC '08)*, pp. E41–E411, October 2008.

[12] J. M. Spivey, *The Z Notation: A Reference manual*, Prentice Hall, London, UK, 1992.

[13] European Electro-Technical Standardization, "Railway applications communications, signaling and processing systems software for railway control and protection systems," The European Standard BS EN 50128, 2001.

[14] J. García, A. Berlanga, J. M. Molina, J. A. Besada, and J. R. Casar, "Planning techniques for airport ground operations," in *Proceedings of the 21st Digital Avionics Systems Conference*, 2002.

[15] P. M. Moertl, J. M. Hitt II, S. Atkins, C. Brinton, and D. H. Walton, "Factors for predicting airport surface characteristics and prediction accuracy of the surface management system," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pp. 3798–3803, October 2003.

[16] T. T. B. Hanh and D. V. Hung, "Verification of an air traffic control system with probabilistic real-time model checking," Tech. Rep. 355, UNU-IIST, 2007.

[17] G. J. M. Koeners, E. P. Stout, and R. M. Rademaker, "Improving taxi traffic flow by real-time runway sequence optimization using dynamic taxi route planning," in *Proceedings of the 30th IEEE/AIAA Digital Avionics Systems Conference (DASC '11)*, October 2011.

[18] G. J. M. Koeners and R. M. Rademaker, "Analyze possible benefits of real-time taxi flow optimization using actual data," in *Proceedings of the 30th Digital Avionics Systems Conference (DASC '11)*, Fremont, Calif, USA, October 2011.

[19] S. Amy, J. S. Philip, and B. Charles, *Ramp Control Issues in the Design of a Surface Management System*, Cognitive Systems Engineering Laboratory, The Ohio State University, 2002.

[20] M. Kwiatkowska, G. Norman, J. Sproston, and F. Wang, "Symbolic model checking for probabilistic timed automata," in *Joint Conference on Formal Modeling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault Tolerant Systems*, vol. 3253 of *Lecture Notes in Computer Science*, pp. 293–208, Springer, 2004.

[21] M. Nguyen-Duc, J.-P. Briot, A. Drogoul, and V. Duong, "An application of multi-agent coordination techniques in air traffic management," in *Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology*, pp. 622–628, October 2003.

[22] L. C. Yang and J. K. Kuchar, "Prototype conflict alerting system for free flight," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 4, pp. 768–773, 1997.

[23] D. P. Alves, L. Weigang, B. Bueno, and B. B. Souza, "Reinforcement learning to support meta-level control in air traffic management," in *Reinforcement Learning: Theory and Applications*, pp. 357–372, ARS Publishing, 2008.

[24] L. Weigang, M. V. P. Dib, D. P. Alves, and A. M. F. Crespo, "Intelligent computing methods in air traffic flow management," *Transportation Research C: Emerging Technologies*, vol. 18, no. 5, pp. 781–793, 2010.

[25] A. Cavcar and M. Cavcar, "Impact of aircraft performance differences on fuel consumption of aircraft in air of management environment," *Aircraft Engineering and Aerospace Technology*, vol. 76, no. 5, pp. 502–515, 2004.

[26] I. Hwang and C. Tomlin, "Protocol-based conflict resolution for finite information horizon," in *Proceedings of the IEEE American Control Conference (ACC '02)*, pp. 748–753, Piscataway, NJ, USA, May 2002.

[27] I. Hwang, J. Hwang, and C. Tomlin, "Flight-mode-based aircraft conflict detection using a residual-mean interacting multiple model algorithm," in *Proceedings of the AIAA Guidance Navigation, and Control Conference*, 2003.

[28] I. Hwang, H. Balakrishnan, K. Roy, and C. Tomlin, "Target tracking and identity management in clutter for air traffic control," in *Proceedings of the American Control Conference (AAC '04)*, 2004.

[29] K. Bousson, "Waypoint-constrained free flight collision avoidance," in *Proceedings of the SAE Advances in Aviation Safety Conference*, 2003.

[30] S. Kahne and I. Frolow, "Air traffic management: evolution with technology," *IEEE Control Systems Magazine*, vol. 16, no. 4, pp. 12–21, 1996.

[31] M. S. Nolan, *Fundamentals of Air Traffic Control*, Brooks/Cole, Wadsworth, Ohio, USA, 3rd edition, 1998.

[32] J. K. Kuchar and L. C. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 4, pp. 179–189, 2000.

[33] J. Hu, M. Pradini, and S. Sastry, "Optimal coordinated maneuvers for three-dimensional aircraft conflict resolution," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 5, pp. 888–900, 2002.

[34] S. R. Wolfe, F. Y. Enomoto, P. A. Jarvis, and M. Sierhuis, "Comparing route selection strategies in collaborative traffic flow management," in *Proceedings of the IEEE Computer Society Technical Committee on Intelligent Informatics (TCII '07)*, pp. 59–62, November 2007.

[35] S. Yousaf, N. A. Zafar, and S. A. Khan, "Formal analysis of departure procedure of air traffic control system," in *Proceedings of the 2nd International Conference on Software Technology and Engineering (ICSTE '10)*, pp. 301–305, October 2010.

[36] N. A. Zafar and K. Araki, "Formalizing moving block railway interlocking system for directed network," *Research Reports on Information Science and Electrical Engineering of Kyushu University*, vol. 8, no. 2, pp. 109–114, 2003.

[37] N. A. Zafar, "Modeling and formal specification of automated train control system using Z notation," in *Proceedings of the IEEE International Multitopic Conference (INMIC '06)*, pp. 438–443, December 2006.

[38] N. A. Zafar, S. A. Khan, and K. Araki, "Towards the safety properties of moving block railway interlocking system," *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 8, pp. 5677–5690, 2012.

[39] N. A. Zafar, "Safety control management at airport taxiing to take-off procedure," *The Arab Journal of Science and Engineering*. In press.

[40] R. Banach, C. Jeske, A. Hall, and S. Stepney, "Retrenchment and the atomicity pattern," in *Proceedings of the 5th IEEE International Conference on Software Engineering and Formal Methods (SEFM '07)*, pp. 37–46, September 2007.

[41] A. C. Garcia, H. Idris, R. Vivona, and S. Green, "Common aircraft performance modeling evaluation tools and experiment results," in *Proceedings of the 24th Digital Avionics Systems Conference (DASC '05)*, pp. 51–59, 2005.

[42] M. Jamal and N. A. Zafar, "Formal model of computer-based air traffic control system using Z notation," in *Proceedings of the 17th International Conference on Computer Theory and Applications*, 2007.

[43] M. Jamal and N. A. Zafar, "Requirements analysis of air traffic control system using formal methods," in *Proceedings of the International Conference on Information and Emerging Technologies (ICIET '07)*, pp. 216–222, July 2007.

[44] M. Medina, L. Sherry, and M. Feary, "Automation for task analysis of next generation air traffic management systems," *Transportation Research C: Emerging Technologies*, vol. 18, no. 6, pp. 921–929, 2010.

[45] S. Pickin, C. Jard, T. Jéron, J.-M. Jézéquel, and Y. Le Traon, "Test synthesis from UML models of distributed software," *IEEE Transactions on Software Engineering*, vol. 33, no. 4, pp. 252–269, 2007.

[46] A. M. F. Crespo, C. V. Aquino, B. B. Souza, L. Weigang, A. C. M. A. Melo A, and D. P. Alves, "Distributed decision support system applied to tactical air traffic flow management in case of CINDACTA I," *Journal of the Brazilian Air Transportation Research Society*, vol. 4, no. 1, pp. 47–60, 2008.

[47] C. Livadas, J. Lygeros, and N. A. Lynch, "High-level modeling and analysis of the Traffic Alert and Collision Avoidance System (TCAS)," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 926–947, 2000.

[48] C. Michael and S. Steven, "Managing gate and ramp operations to reduce delay, fuel burn, and costs," in *Proceedings of the Integrated Communications, Navigation and Surveillance Conference (ICNS '12)*, 2012.