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, Saudi Arabia

[2] Department of Computer Sciences, Abdul Wali Khan University, Mardan, Pakistan

Abstract: *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 occurred at airports surface are 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 post conditions ensuring the safety. Formal specification is analysed using Z/Eves tool.*

1. Introduction

Air traffic control (ATC) system is highly a 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 [40], next generation ATC systems are suggested to improve efficiency by not compromising at safety standards [8–11]. Although an automated support to ATC system is available now a day but still it is heavily dependent upon human interaction causing delays and accidents due to failure of communication in decision making [18], [37]. Therefore, developing an ATC system enabling aircraft to move at airport and freely fly in air is a current issue [7]. 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 occurred at the airport surface are three time more than the collisions in the airspace [29]. 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 pre-flight 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 by the scientific community addressing ATC system [15]. 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 [14].

In this paper, formal procedure of managing air traffic on the airport surface from start of gate to enter in a 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 [38]. It is noted that according to European electro-technical standards, the use of formal methods is recommended to achieve a required level of confidence in development of safety critical system [39]. 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 exits, at most, one aircraft in one region preventing collision at the airport surface. The traffic sequence is updated by defining pre/post conditions 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 is discussed in section 2. In most of the work safety criteria are developed by testing through simulation but unfortunately this approach is lacking of 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 re-conducted 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.

Rest of the paper is organized as: 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 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 [12]. The work is interesting in which the airport capacity and online scheduling the airport surface 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 [32]. In this article, 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 [25]. However, in this work, only a part of the planning function is evaluated using scenarios based approach. In [36], 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, however, 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 [2]. Again a detailed procedure is described in this work, however, no validation or verification approach is provided to prove its correctness. In [27], PRISM tool is used to verify and analyse the properties of ATC system using timed automata. Multi-agent approach is applied in modelling of air traffic flow management (ATFM) in [33]. The objective of this research is to show applicability of agent-based simulation. In another work of NASA, it is developed a collaborative air traffic flow using multi-agent simulation technique. Several strategies were used to select various routes increasing complexity of the system [43]. A fusion of intelligent computing is studied for development of a new tactical system for ATFM using advantages of the meta-level control approach [1]. In another application, intelligent computing models are claimed for ATFM as presented in [41]. Due to increasing of air traffic density and relatively limited number of airways, the future solution for optimal airspace and safe air traffic control are proposed in [5]. A protocol for aircraft conflict resolution is proposed in [19–21] 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 pre-defined flight schedules in [4] [24] [34]. This is very good effort for developing free flight ATC systems. The performance of conflict detection and resolution is presented in [26] 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 [17] [42]. Preliminary results of our research are reported in [45-49]. Other similar work can be found in [3] [13] [22] [23] [30] [35].

3. Problem Statement and Formulation

Ensuring safety and increasing efficiency of an ATC system has become a central issue due to increase of air-traffic and introduction of new technologies [6]. The primary objective ATC system is to provide a safe and efficient flow of air traffic [28]. 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, 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 is totally depends on the airport. From gate to take-off, few major reasons for delay in a flight are: (i) un-optimized 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 are 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 scope of this research.

In static part of the model, the airport surface is divided into different regions and then transformed into a graph as in the Figure 1. 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, defining aprons and connectivity from apron to active area for taxiing are three main activities addressed in dynamic part of system.
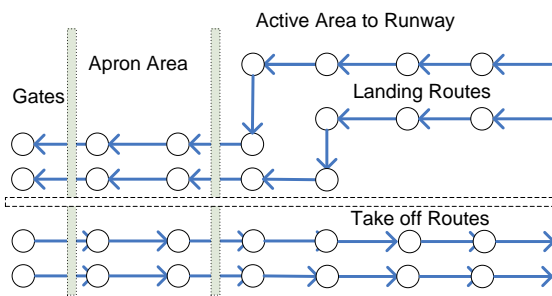

Figure 1: Airport model represented by a directed graph.

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. In the 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 requires to and from movement of aircrafts expeditiously. 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 expediting 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 as in Figure 1. A 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 that an aircraft can move from node u to node v.

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

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, i.e., 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
_____

$blocks: \mathbb{F} \; Block$
$links: \mathbb{F} \; Links$

_____

$\forall b1, b2: Block \mid (b1, b2) \in links \cdot b1 \in blocks \wedge b2 \in blocks$
$\forall b: blocks \cdot \exists b1, b2: Block \mid (b1, b2) \in links \cdot b = b1 \vee b = b2$
$\forall b1, b2: blocks \cdot \exists path: \text{seq } Block$
$\cdot \forall i: \mathbb{N} \mid i \geqslant 1 \wedge i < \# path \cdot (path \; i, path \; (i+1)) \in links$

_____

Passenger gate is represented by the *Gate* schema constituted by two components, i.e., 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 ⇸ Gate
│─────────────────────────────
│ ∀gid: Block; gate: Gate |(gid, gate) ∈ gates • gid = gate .
│ gateid
└────────────────────────────────────────
```

Apron area is used for pre-flight 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 ⇸ Apron
│─────────────────────────────
│ ∀apid: Block; apron: Apron | (apid, apron) ∈ aprons
│  • apid = apron . 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 → Taxiway
│ taxiingA: TaxiwayId ⇸ AircraftId
│─────────────────────────────
│ ∀tid: TaxiwayId; taxiway: Taxiway | (tid, taxiway) ∈
│ taxiways
│   • tid = taxiway . taxiwayid
│ dom taxiingA ⊆ 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
│─────────────────────────────
│ ∀b: Block; gate: Gate | (b, gate) ∈ gates • gate . gateid ∈
│ blocks
│ ∀b: Block; apron: Apron | (b, apron) ∈ aprons • apron .
│ apid ∈ blocks
│ dom gates ∩ dom aprons = ∅
│ ∀tw: ran taxiways
│   • ∃path1: seq Block | tw . path = path1
│     • ∀i: ℕ | i ∈ dom path1 ∧ i ∈ 1 .. # path1 - 1
│       • (tw . path i, tw . path (i + 1)) ∈ links
│ ∀b1: dom gates ∪ dom aprons
│   • ∃tw: ran taxiways • ∃b2: ran tw . path
│     • ∃route: seq Block • ∀i: ℕ | i ⩾ 1 ∧ i < # route
│       • (route i, route (i + 1)) ∈ 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 → Aircraft
│─────────────────────────────
│ ∀aid: AircraftId; acr: Aircraft | (aid, acr) ∈ aircrafts
│   • aid = acr . aircraftid
│ ∀ac1, ac2: ran aircrafts • ac1 . safeArea ∩ ac2 . safeArea
│ = ∅
└────────────────────────────────────────
```

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 for 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 identified any relationship, if exists, with the rest of the components.

GateController
───────────────────────────────

Gates
gatesR: seq *AircraftId*
gatesA: AircraftId $\rightarrow$ *Gate*
pushbackR: $\mathbb{F}$ *AircraftId*
─────────────────────────────

$\forall$*gate:* ran *gates*
　• *gate* $\in$ ran *gatesA* $\Rightarrow$ *gate . gatestate = OCCUPIED* $\wedge$
　　gate $\notin$ ran *gatesA* $\Rightarrow$ *gate . gatestate = CLEAR*
ran *gatesR* $\cap$ dom *gatesA* $= \varnothing$
pushbackR $\subseteq$ dom *gatesA*
───────────────────────────────

Invariants: (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 for a gate. If an aircraft has requested for a gate, it cannot be in the list of aircrafts which are assigned a gate. (iii) If an aircraft has requested for 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 • aid* $\notin$ ran *apronQ* $\cup$ ran *taxiingR*
$\forall$*aid:* ran *apronQ • aid* $\notin$ *pushbackC*
$\forall$*aid:* ran *taxiingR • aid* $\in$ ran *apronQ* $\wedge$ *aid* $\notin$ *pushbackC*
───────────────────────────────

Invariants: (i) Any aircraft which has requested for pushback clearance, it cannot be in the list of aircrafts in the apron area. (ii) If an aircraft is in the apron area, it has not requested for the pushback clearance. (iii) If an aircraft has requested for taxiing, it is in the apron area but not in the list of aircrafts which has requested for 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 Ξ*ApronController*, Δ*GateController* and aircraft identifier *aid?* as inputs. The state of gate controller is updated by verifying the properties as pre and post conditions. It is noted that post condition 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/post conditions are described in second part of the schema.

RequestGate
───────────────────────────────

Ξ*ApronController*
Δ*GateController*
aid?: AircraftId
─────────────────────────────

aid? $\in$ ran *apronQ*
aid? $\notin$ ran *gatesR*
gatesR' = gatesR $^\frown$ $\langle$*aid?*$\rangle$
gatesA' = gatesA
pushbackR' = pushbackR
───────────────────────────────

Pre/post conditions: (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, Δ*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 post conditions in the predicate part of the schema.

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

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

Pre/post conditions: (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/post conditions for updating state space of gates.

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

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

Pre/post conditions: (i) The aircraft must be assigned a gate before sending a pushback request. (ii) The aircraft has neither 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/post conditions.

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

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

Pre/post conditions: (i) The aircraft must be assigned a gate. (ii) It is in the lists of pushback requests and pushback clearance. (iii) The aircraft is un-assigned the gate. (iv) The aircraft is removed from the pushback and clearance lists. (v) The aircraft is allowed to enter the apron area. (vi) 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_____
$\Delta ApronController$
$aid?: AircraftId$
_____

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

Pre/post conditions: (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 are 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 "↑" operation.

$$
\begin{array}{l}
\underline{EnterTaxi}\\
\Delta ApronController\\
\Delta Taxiways\\
acr?: Aircraft\\
tid?: TaxiwayId\\
taxiway?: Taxiway\\
\hline
acr? . aircraftid \in \text{ran } taxiingR\\
(tid?, taxiway?) \in taxiways\\
acr? . aircraftid = apronQ\ 1\\
taxiingR'\\
\quad = \{\ i\colon \mathbb{N};\ aid\colon AircraftId \mid i \in \text{dom } taxiingR \wedge (i, acr? .\\
\quad\quad aircraftid) \notin taxiingR \wedge aid \neq acr? . aircraftid \cdot i\ \}\\
\quad\quad \upharpoonleft taxiingR\\
apronQ'\\
\quad = \{\ i\colon \mathbb{N};\ aid\colon AircraftId \mid i \in \text{dom } apronQ \wedge (i, acr? .\\
\quad\quad aircraftid) \notin apronQ \wedge aid \neq acr? . aircraftid \cdot i\ \}\\
\quad\quad \upharpoonleft apronQ\\
pushbackC' = pushbackC\\
taxiingA' = taxiingA \cup \{(tid? \mapsto acr? . aircraftid)\}\\
taxiways' = taxiways
\end{array}
$$

Pre/post conditions: (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) Rest of the variables of apron controller are 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 about complete correctness of formal specification. That means even 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. 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. Prove by reduce is a techniques in which equivalent simpler combinations of tactics is substituted. Reduction and proof by reduction are represented in columns 4 & 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 that reduction is not required on the predicates and, hence, the formal specification is proved to be written well and meaningful.

Table 1: Results of model analysis

Schema Name	Syntax Type Check	Domain Check	Reduction	Proof by Reduce
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

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 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 post conditions 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 us both in static and dynamic parts of the model increasing reusability. Systematic development from abstraction to detailed model made it easy to purpose 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 Carpenter et. al is close to this in which gate management and ramp operations are analysed for reducing delay time, fuel burning and other costs [31]. 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 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 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 like a real ATC system.

References

[1] Alves, D. P., Weigang, L., Bueno, B., Souza, B. B.: Reinforcement learning to support meta-level control in air traffic management, Reinforcement Learning: Theory and Applications, ARS Publishing, pp. 357-72, 2008.

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

[3] Banach, R., Hall, A., Stepney, S.: Retrenchment and the atomicity pattern, Fifth IEEE International Conference on Software Engineering and Formal Methods, pp. 37-46, 2007.

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

[5] Cavcar, A., Cavcar, M.: Impact of aircraft performance differences on fuel consumption of aircraft in air traffic management environment, Aircraft Engineering Aerospace Technology, 76 (5), pp. 502-15, 2004.

[6] Crespo, A. M. F., Aquino, C. V., Souza, B. B., Weigang, L., Melo, A. C. M. A., Alves, D. P.: Distributed decision support system applied to tactical air traffic flow management in case of CINDACTA, I. Journal of the Brazilian Air Transportation Research Society, 4 (1), pp. 47-60, 2008.

[7] Debbache, N. E.: Toward a new organization for air traffic control, Aircraft Engineering Aerospace Technology, 73 (6), pp. 561-67, 2001.

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

[9] Erzberger, H.: Automated conflict resolution for air traffic control, Proceedings of the 25th International Congress of the Aeronautical Sciences, 2006.

[10] Erzberger, H., Heere, K.: Algorithm and operational concept for resolving short range conflicts, Journal of Aerospace Engineering, 224, pp. 225-43, 2009.

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

[12] Garcia, J., Berlanga, A., Molina, J. M., Besada, J. A., Casar, J. R.: Planning techniques for airport ground operations, Proceedings The 21st Digital Avionics Systems Conference, 2002.

[13] Garcia, A. C., Idris, H., Vivona, R., Green, S.: Common aircraft performance modeling evaluation tools and experiment results, Proceedings of the 24th DASC, pp. 51-59, 2005.

[14] George, J. C., Robert, K. F., Michael, B. D., Nathan, M., D. Signor, Ari, S., Thomas H.: A new modeling capability for airport surface traffic analysis, 27th IEEE /AIAA Digital Avionics Systems Conference, 2008.

[15] Guo, Y., Cao X., Zhang, J.: Constraint handling based multi-objective evolutionary algorithm for aircraft landing scheduling, International Journal of Innovative Computing, Information and Control, 5(8), pp. 2229-38, 2009.

[16] Hanh, T. T. B., Hung, D. V.: Verification of an air traffic control system with probabilistic real-time model checking, UNU-IIST, Report No. 355, 2007.

[17] Hu, J., Prandini, M., Sastry, S.: Optimal coordinated maneuvers for three-dimensional aircraft conflict resolution, Journal of Guidance, Control, and Dynamics, 25 (5), pp. 888-00, 2002.

[18] Hu, J., Prandini, M., Sastry, S.: Optimal maneuver for multiple aircraft conflict resolution, A braid point of view, Proceedings of the 39th IEEE Conference on Decision and Control, (4), pp. 4164-69, 2000.

[19] Hwang, I., Tomlin, C.: Protocol-based Conflict Resolution for Finite Information Horizon, Proceedings of the AACC American Control Conference, IEEE Publ., Piscataway, NJ, 2002.

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

[21] Hwang, I., Balakrishnan, H., Roy, K., Tomlin, C.: Target tracking and identity management in clutter for air traffic control, Proceedings of the AACC American Control Conference, 2004.

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

[23] Jamal, M., Zafar, N. A.: Requirements analysis of air traffic control system using formal methods, Proceedings of IEEE International Conference on Information and Emerging Technologies, pp. 216-22, 2007.

[24] Kahne, S., Frolow, I.: Air traffic management: Evolution with technology, IEEE Control Systems Magazine, 16 (4), 1996.

[25] Koeners, G. J. M., Stout, E. P., Rademaker, R. M.: Improving taxi traffic flow by real-time runway sequence optimization using dynamic taxi route planning, 30th IEEE/AIAA Digital Avionics Systems Conference, 2011.

[26] Kuchar, J. K., Yang, L. C.: A review of conflict detection and resolution modeling methods, IEEE Transactions on Intelligent Transportation Systems, 1 (4), pp. 179-89, 2000.

[27] Kwiatkowska, M., Norman, G., Sproston, J., Wang, F.: Symbolic model checking for probabilistic timed automata, Joint Conference on Formal Modeling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault Tolerant Systems, LNCS, Springer, 3253, pp. 293-08, 2004.

[28] Livadas, C., Lygeros, J., Lynch, N. A.: High level modeling and analysis of the traffic alert and collision avoidance system, Proceedings of The IEEE, 88 (7), pp. 926-48, 2000.

[29] Marshall, W., Joseph, W. I.: Airport Movement Area Safety System, IEEE Proceedings of Digital Avionics Systems Conference, pp. 549 - 52 1992.

[30] Medina, M., Sherry, L., Feary, M.: Automation for task analysis of next generation air traffic management systems, Transportation Research Part C, 18, pp. 921-29, 2010.

[31] Michael, C., Steven, S.: Managing gate and ramp operations to reduce delay, fuel burn, and costs, Integrated Communications, Navigation and Surveillance Conference (ICNS), 2012.

[32] Moertl, P. M., Atkins, S., Hitt, J. M., Brinton, C., Walton, D. H.: Factors for Predicting Airport Surface Characteristics and Prediction Accuracy of the Surface Management System, IEEE International Conference on Systems, Man and Cybernetics, 2003.

[33] Nguyen-Duc, M., Briot, J. P., Drogoul, A., Duong, V.: An application of multi-agent coordination techniques in air traffic management, Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology, pp. 622-28, 2003.

[34] Nolan, M. S.: Fundamentals of air traffic control, Brooks/Cole, Wadsworth, 3rd edition, 1998.

[35] Pickin, S., Jard, C., Thierry, J. R., Jeze, J. M., Traon, Y. L.: Test synthesis from UML models of distributed software, IEEE Transaction on Software Engineering, 33 (4), pp. 252-268, 2007.

[36] Rademaker, R., Koeners, G. J. M.: Analyze possible benefits of real-time taxi flow Optimization using actual data, 30th IEEE/AIAA Digital Avionics Systems Conference, 2011.

[37] Shorrock, S. T., Kirwan, B.: Development and application of a human error identification tool for air traffic control, Applied Ergonomics, 33 (4), pp. 319-36, 2002.

[38] Spivey, J. M.: The Z notation: a reference manual, Prentice Hall International, (UK) Ltd., 1992.

[39] 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.

[40] Villiers, J.: ERASMUS - A friendly way for breaking the capacity barrier. ITA, 58, 2004.

[41] Weigang, L., Dib, M. V. P., Alves, D. P., Crespo, A. M. F.: Intelligent computing methods in air traffic flow management, Transportation Research Part C: Emerging Technologies, 18 (5), pp. 781-93, 2010.

[42] Wolfe, S. R., Jarvis, P. A., Enomoto, F. Y., Sierhuis, M.: Comparing route selection strategies in collaborative traffic flow management, EEE/WIC/ACM International Conference on Intelligent Agent Technology, IEEE Press, Fremont, pp. 59-62, 2007.

[43] Yang, L. C., Kuchar, J. K.: Prototype conflict alerting system for free flight, Journal of Guidance, Control, and Dynamics, 20 (4), 1997.

[44] Yousaf, S., Khan, S. A., Zafar, N. A., Ahmad, F., Khan, M. A.: Formal Analysis of Arrival Procedure of Air Traffic Control System, Life Science Journal 9 (4), 2012.

[45] Yousaf, S. Zafar, N. A., Khan, S. A.: Formal analysis of departure procedure of air traffic control system, 2nd International Conference on Software Technology and Engineering, 2010.

[46] Zafar, N. A., Araki, K.: Formalizing moving block railway interlocking system for directed network, Research Reports on Information Science and Electrical Engineering, Kyushu University, Japan, 8(2), pp. 109-14, 2003.

[47] Zafar, N. A.: Modeling and formal specification of automated train control system using Z notation, IEEE Multitopic Conference, INMIC06, pp. 438-43, 2006.

[48] Zafar, N. A., Khan, S. A., Araki, K.: Towards the safety properties of moving block railway interlocking system, International Journal of Innovative Computing, Information and Control, 8(8), pp. 5677-90, 2012.

[49] Zafar, N. A.: Safety Control Management at Airport Taxiing to Take-off Procedure, The Arab Journal of Science and Engineering, 2014 (Accepted).