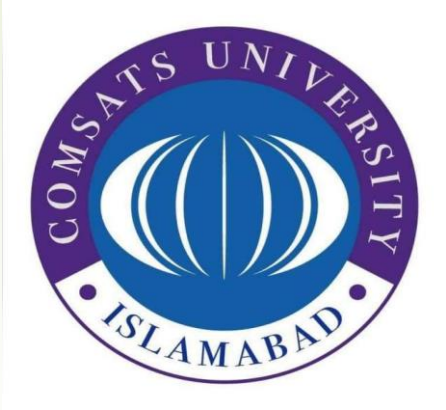

Department of Computer Science, CUI Lahore Campus

Formal Methods

By

Farooq Ahmad

The Airport Class

- A system that keeps track of aircraft that are allowed to land at a particular airport. Aircraft must apply for permission to land at the airport prior to landing. When an aircraft arrives to land at the airport it should only have done so if it had previously been given permission. When an aircraft leaves the airport its permission to land is also removed.

Operations on System

- **givePermission**: records the fact that an aircraft has been granted permission to land at the airport.
- **recordLanding**: records an aircraft as having landed at the airport.
- **recordTakeOff**: records an aircraft as having taken off from the airport.
- **getPermission**: returns the aircrafts currently recorded as having permission to land.
- **getLanded**: returns the aircrafts currently recorded as having landed.
- **numberWaiting**: returns the number of aircrafts granted permission to land but not yet landed.

UML Model

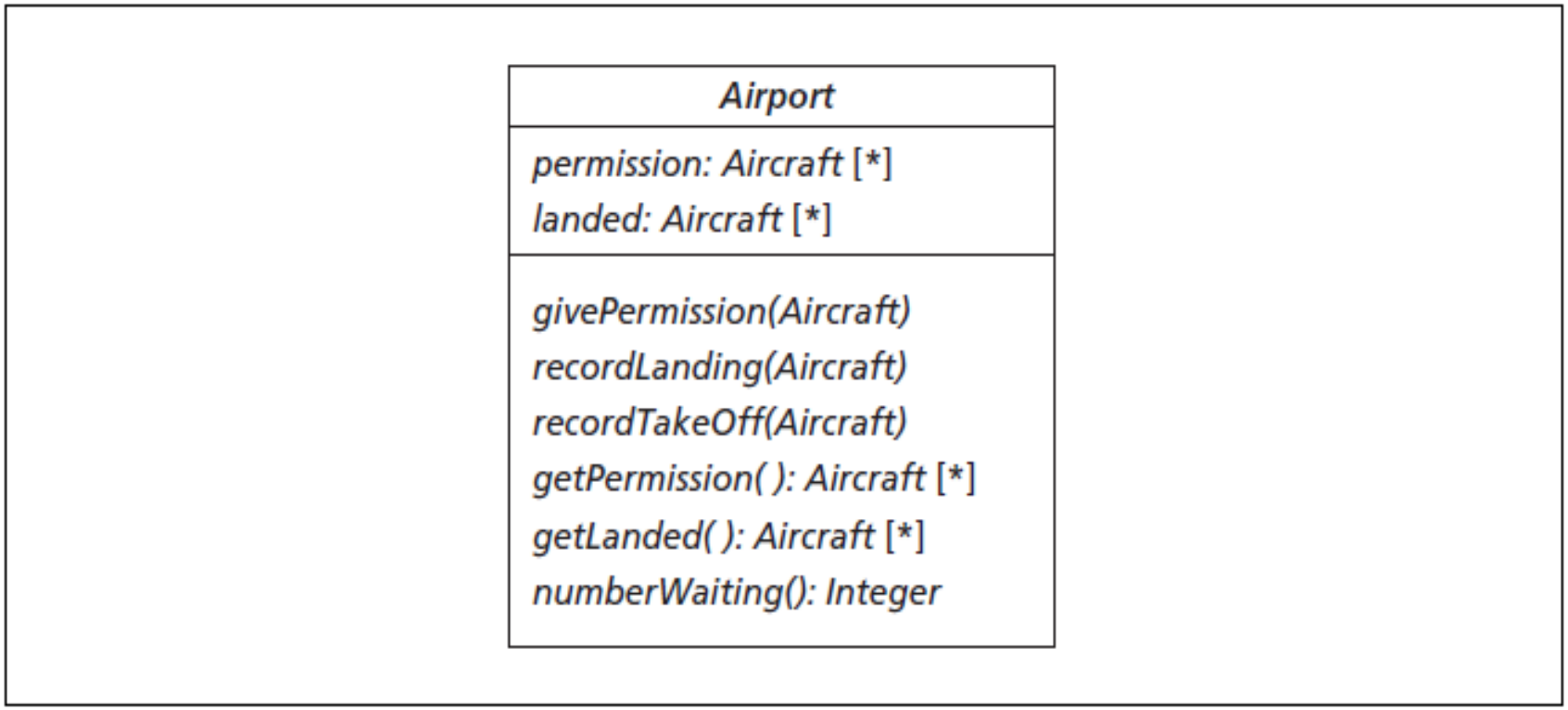

Figure 5.2 A UML specification of the *Airport* class

types

types

Aircraft = TOKEN

state

state *Airport* of

permission: *Aircraft* -set

landed: *Aircraft* -set

Invariant and Initialization

$$\text{inv mk-}Airport(p,l) \triangleq l \subseteq p$$

$$\text{init mk-}Airport\ (p, l) \triangleq p = \{\ \} \wedge l = \{\ \}$$

Data Specification

types

Aircraft = TOKEN

state *Airport* of

permission: *Aircraft*-set

landed: *Aircraft* -set

inv mk-*Airport*(*p*,*l*) $\underline{\Delta}\ l \subseteq p$

init mk-*Airport* (*p*, *l*) $\underline{\Delta}\ p = \{\} \wedge l = \{\}$

end

Operations

givePermission (*craftIn*: *Aircraft*)

ext wr *permission*: *Aircraft* -set

pre $craftIn \notin permission$

post $permission = \overline{permission} \cup \{craftIn\}$

Operations

recordLanding (*craftIn*: *Aircraft*)

ext rd *permission*: *Aircraft* -set

wr *landed*: *Aircraft* -set

pre $craftIn \in permission \wedge craftIn \notin landed$

post $landed = \overline{landed} \cup \{craftIn\}$

Operations

recordTakeOff (*craftIn*: *Aircraft*)

ext wr *permission*: *Aircraft* **-set**

wr *landed*: *Aircraft* **-set**

pre $craftIn \in landed$

post $landed = \overline{landed} \setminus \{craftIn\} \wedge permission = \overline{permission} \setminus \{craftIn\}$

Operations

getpermission( ) *out*: *Aircraft* -set

ext rd *permission*: *Aircraft* -set

pre TRUE

post *out* = *permission*

Operations

getLanded( ) out: Aircraft -set

ext rd *landed: Aircraft* -set

pre TRUE

post *out = landed*

Operations

numberWaiting( ) *total*: ℕ

ext rd *permission*: *Aircraft* -set

rd *landed*: *Aircraft* -set

post *total* = card (*permission* \ *landed*)

Reference and Reading Material

- Formal Software Development From VDM to Java, Chapter # 5: **Sets**