Software Engineering FIT3037
Sample Solution Assignment 2 Second Semester, 2011

[FIRSTNAME, FAMILYNAME, ADDRESS, REGISTRATION, REAL,
DATE, DOLLARS]
FIRSTNAME is the type for the set of all possible first names
FAMILYNAME is the type for the set of all possible family names
ADDRESS is the type for the set of all possible members addresses
REGISTRATION is the type for the set of all possible plane registration numbers
REAL is the type for the set of floating point numbers
DATE is the type for the set of all possible dates
DOLLARS is the type for the set of all money values

Member_Type ::= SOCIAL | SOLO | STUDENT
Committee_Member::= COMMITTEE | NON_COMMITTEE
Availability ::= AVAILABLE | FLYING | SERVICE
Num_Engines ::= SINGLE | DUAL
RESPONSE ::= Initialised | Cannot re-initialise| OK| Already Member | Unknown Plane |
Plane Unavailable | Plane Available | Pilot Licenced | Unknown Pilot |
Pilot Not Licenced | Not a member | Incorrect family name |
Not enough seats | Flight entered | Durations must be > 0.0 |
Plane not currently flying | Flight not found

NextMember_No :
Next_Flight_No :

State space schema

AeroClubSys

familynames : FAMILYNAME
firstnames : FIRSTNAME
addresses : ADDRESS
member_types : Member_Type
member_hours : REAL
committee : ℙ
flying_hours : REGISTRATION REAL
engine : REGISTRATION Num_Engines
seats : REGISTRATION
plane_availability :REGISTRATION Availability
hiring_cost : REGISTRATION DOLLARS
flight_plane : REGISTRATION
flight_pilot :
flight_passengers :
flight_duration : REAL

```
    flight_date  :                  DATE
 ____________________________________
|
|   dom familynames = dom firstnames = dom addresses = dom member_types
|   dom familynames = dom member_hours
|   committee ⊆ dom familynames
|   dom flying_hours = dom engine = dom seats = dom plane_availability = dom hiring_cost
|   dom flight_plane = dom flight_pilot = dom flight_passengers = dom flight_duration
|   dom flight_plane = dom flight_date
|   ran flight_planes  ⊆ dom flying_hours
|   ran flight_pilot  ⊆ dom familynames
|____________________________________________
```

****

Initialisation operation

```
 ____ Init ______________________
|   Δ AeroClubSys
|   rep! : RESPONSE
|____________________________________
|
|   ( (#(dom familynames)< 1    #(dom flying_hours) < 1    #(dom flight_plane) < 1 )
|   familynames' =  ∅
|   flying_hours' =  ∅
|   flight_planes' =  ∅
|   committee' = ∅
|   rep! = Initialised
|   NextMember_No  = 1
|   Next_Flight_No  = 1 ) ∨
|   Rep! = Cannot re-initialise
|____________________________________________
```

****

Schemas for Add_Member operation

```
┌─── Add_Member_Basic ──────────────────
│ Δ AeroClubSys
│ fam? : FAMILYNAME
│ fir? : FIRSTNAME
│ add? : ADDRESS
│ type? : Member_Type
│ com?  :Committee_Member
│ r! : RESPONSE
├──────────────────────────
│ (fam? ∉ ran familynames ∨ #{x: familynames~(fam?) | firstnames(x) == fir?} = 0 )
│ r! = OK
│ familynames’ =  familynames    { (NextMember_No, fam?)}
│ firstnames’ = firstnames    { (NextMember_No, fir?)}
│ addresses’ = addresses    { (NextMember_No, add?)}
│ member_types’ = member_types    { (NextMember_No, type?)}
│ member_hours’ = member_hours    { (NextMember_No, 0.0)}
│ com?== COMMITTEE
│ committee’= committee    { NextMember_No}
│ NextMember_No’ = NextMember_No + 1
└──────────────────────────────
```

```
┌─── Already_A_Member ──────────
│ Ξ AeroClubSys
│ r! : RESPONSE
├──────────────────────────
│ r! = Already Member
└──────────────────────────────
```

Add_Member ≙ Add_Member_Basic ∨ Already_A_Member

****

Schemas for New_Flight operation

Check that plane is available

```
┌─── Plane_ Available_Now      ─────────
│ Ξ AeroClubSys
│ plane? : REGISTRATION
│ r! : RESPONSE
├──────────────────────────
│ plane? ∈ dom plane_availability
│ plane_availability(plane?) == AVAILABLE
│ r! = Plane Available
│
```

```
┌─── Plane_Not_Available_Now ───────
│ Ξ AeroClubSys
│ plane? : REGISTRATION
│ r! : RESPONSE
├──────────────────────────
│ (plane? ∉ dom plane_availability
│ r! = Unknown Plane  ) ∨
│ plane_availability(plane?)     AVAILABLE
│ r! = Plane Unavailable
└──────────────────────────────
```

Next, is pilot a member and licenced?

```
┌─── Pilot_Licenced       ──────────
│ Ξ AeroClubSys
│ p? :
│ fam? : FAMILYNAME
│ r! : RESPONSE
├──────────────────────────
│ p? ∈ dom familynames
│ fam? == familynames(p?)
│ member_types(p?) == SOLO
│ r! = Pilot Licenced
└──────────────────────────────
```

```
┌─── Pilot_Not_Licenced  ──────────
│ Ξ AeroClubSys
│ p? :
│ fam? : FAMILYNAME
│ r! : RESPONSE
├──────────────────────────
│ (p? ∉ dom familynames
│ r! =  NotaMember ) ∨
│ (fam?    familynames(p?)
│ r! =  Incorrect family name ) ∨
│ member_types(p?)    SOLO
│ r! =  Pilot Not Licenced
└──────────────────────────────
```

Not_Enough_Seats

Ξ AeroClubSys
plane? : REGISTRATION
cust? :
r! : RESPONSE

seats(plane?) < cust?
r! = Not enough seats

Actually enter new flight

```
┌─── Enter_Flight_Details ─────────
│ Δ AeroClubSys
│ p? :
│ plane? : REGISTRATION
│ cust? :
│ r! : RESPONSE
├────────────────────────
│ seats(plane?)   = cust?
│ flight_plane '  = flight_plane    {(Next_Flight_No, plane?)}
│ flight_pilot '  = flight_pilot    {(Next_Flight_No, p?)}
│ flight_passengers '  = flight_passengers    {(Next_Flight_No, cust?)}
│ flight_duration '  = flight_duration    {(Next_Flight_No, 0)}
│ plane_availability '  = plane_availability    {(plane?, FLYING)}
│ plane_date ' = plane_date    {(Next_Flight_No, GET_DATE())}
│ Next_Flight_No  = Next_Flight_No + 1
│ r! = Flight entered
└──────────────────────────────
```

New_Flight ≙ (Pilot_Licenced ∧Plane_Available_Now ∧ Enter_Flight_Details) ∨
Pilot_Not_Licenced ∨ Plane_Unavailable_Now ∨ Not_Enough_Seats

****

Schemas for Flight_Completed Operation

```
┌──── Enter_Flying_Hours ──────────────────
│ Δ AeroClubSys
│ rego : REGISTRATION
│ date? :  DATE
│ duration? : REAL
│ r! : RESPONSE
│ f :
│ flying :  ℙ
│ new_total : REAL
├──────────────────────────────
│ flying = { x : dom flight_plane | flight_plane(x)==rego? ∧ plane_availability(flight_plane(x))
│   == FLYING ∧ flight_date(x)==date? }
│ #flying == 1
│ x:flying  • f = x
│ duration? > 0.0
│ plane_availability(flight_plane(f)) == FLYING
│ flight_duration’ = flight_duration    {(f, duration?)}
│ plane_availability’ = plane_availability    {(flight_plane(f), AVAILABLE)}
│ new_total = flying_hours(flight_plane(f)) + duration?
│ flying_hours’ = flying_hours    {(flight_plane(f), new_total)}
│ new_total=member_hours(flight_pilot(f))+ duration?
│ member_hours’= member_hours    {(flight_pilot(f), new_total)}
│ r! = OK
└──────────────────────────────────

┌──── Wrong_Flight ────────────────
│ Ξ AeroClubSys
│ rego? : REGISTRATION
│ date? :  DATE
│ flying :  ℙ
│ r! : RESPONSE
├──────────────────────────────
│ (rego? ∉ flight_plane
│    r! = Unknown Plane )
│ ( flying = { x : dom flight_plane | flight_plane(x)==rego? ∧ plane_availability(flight_plane(x))
│   == FLYING ∧ flight_date(x)==date? }
│   ( # f lying == 0
│     r! = Flight not found ) )
└──────────────────────────────────

┌──── Duration_Not_Valid ──────────
│ duration? : REAL
│ r! : RESPONSE
├──────────────────────────────
│ duration ≤ 0.0
│
```

r! = Duration must be > 0.0

Flight_Completed ≙ Enter_Flying_Hours ∨ Wrong_Flight ∨ Duration_Not_Valid

****

Flying_Hours Operation

Get_Flying_Hours

Ξ AeroClubSys
family? : FAMILYNAME
given? : FIRSTNAME
flown! : REAL
p :

family ∈dom familynames
given? ∈dom firstnames
x: dom familynames | familynames(x) == family? ∧ firstnames(x) == given? • p=x
flown! = member_hours(p)

Wrong_Name

Ξ AeroClubSys
family? : FAMILYNAME
given? : FIRSTNAME
person : ℙ
r! : RESPONSE

person = {   x: dom familynames | familynames(x) == family? ∧ firstnames(x) == given?}
#person < 1
r! = Not a Member

Flying_Hours ≙ Get_Flying_Hours ∨ Wrong_Name

****

Currently_Flying Operation

```
┌──── Currently_Flying ─────────────
│ Ξ AeroClubSys
│ flying  : ℙ
│ member_flying! : REGISTRATION
│ member_name! : REGISTRATION     FAMILYNAME
├───────────────────────
│ flying = { x : dom flight_plane | plane_availability(flight_plane(x)) == FLYING }
│ member_flying! =  { y: flying   flight_plane(y)     flight_pilot(y) }
│ member_name! =  { y: flying   flight_plane(y)     familynames(flight_pilot(y)) }
└────────────────────────────
```

****

Committee_Flying_on_Date Operation

```
┌──── Committee_Flying_on_Date ─────
│ Ξ AeroClubSys
│ date? :  DATE
│ flying  : ℙ
│ member_flying! : REGISTRATION
│ member_name! : REGISTRATION     FAMILYNAME
│ member_firstname! : REGISTRATION     FIRSTNAME
├───────────────────────
│ flying = { x : dom flight_plane | flight_date(x) == date?}
│ flying' = flying    committee
│ member_flying! =  { y: flying   flight_plane(y)     flight_pilot(y) }
│ member_name! =  { y: flying   flight_plane(y)     familynames(flight_pilot(y)) }
│ member_firstname! =  { y: flying   flight_plane(y)     firstnames(flight_pilot(y)) }
└────────────────────────────
```