CS412 Solutions to exercise sheet 7

Refinements

1. There are a number of ways you could use the function representation to remove nondeterminism. Here, spaces are allocated from the minimum available domain position. If someone cancels, their position is left as a "space" that can be filled.

```
REFINEMENT
   Passengers_r
REFINES
   Passengers

VARIABLES
    passfun

INVARIANT
    passfun: 1..50 >+> PID & ran(passfun) = pass

INITIALISATION
    passfun := {}

OPERATIONS
   book ( pp ) =  VAR xx IN
                      xx := min({nn|nn:1..50 & nn/:dom(passfun)});
                      passfun := passfun <+ {xx |-> pp}
                  END;

    /* Note that we don't have to worry about precondition again -
       that's all been stated at the previous level.               */

   cancel ( pp ) = passfun := passfun |>> {pp};

   oo <-- query ( pp ) = oo := bool ( pp : ran(passfun) );

   oo <-- spaces = oo := 50 - card ( ran(passfun) )


END
```

2. Without seeing the full details of what operations these machines intend

to provide it's not possible to be completely sure the linking invariants capture a suitable relationship. But here are some likely ways.

(a) $stock = \mathrm{dom}(bookarr) \wedge notforloan = \mathrm{dom}(bookarr \rhd \{neverloan\}) \wedge$
$onloan = \mathrm{dom}(bookarr \rhd \{outonloan\}) \wedge$
$stock - onloan - notforloan = \mathrm{dom}(bookarr \rhd \{readytoloan\})$

(b) $marked = ASSIGNMENT - \mathrm{ran}(tomark)$

(c) $\mathrm{dom}(mname) = \mathrm{dom}(minfo) \wedge \mathrm{dom}(maddr) = \mathrm{dom}(minfo) \wedge$
$\forall\, mm \bullet (mm \in \mathrm{dom}(minfo) \Rightarrow$
$minfo(mm) = mname(mm) \mapsto maddr(mm))$

3. Obviously there are lots of possibilites, but here's one way. You might be able to spot some ambiguity in the specification.

(a)
```
MACHINE        OpenDay

SETS           SID

PROPERTIES     card(SID) > 30

VARIABLES      volunteers, chosen

INVARIANT      volunteers <: SID & chosen <: SID  &
               card(volunteers) <= 30 & chosen <: volunteers &
               card(chosen) : {0,6}

INITIALISATION volunteers, chosen := {}, {}

OPERATIONS

   newvolunteer(vv) =
    PRE   vv : SID & vv /: volunteers & card(volunteers) < 30
    THEN  volunteers := volunteers \/ {vv}
    END;

  swap(v1,v2) =
    PRE   v1 : SID & v2 : SID & v1 : volunteers & card(vols) = 30 &
          v1:volunteers & v2 /: volunteers
    THEN  volunteers := (volunteers - {v1}) \/ {v2}
    END;

newchoice =
    PRE (chosen = {} & card(volunteerss) >= 6) or
        (chosen /= {} & card(volunteers) >= 12)
    THEN ANY cset
        WHERE cset <: volunteers & card(cset) = 6 & cset /\ chosen = {}
        THEN chosen := cset
```

```
        END
    END;
  /* Again, there could be a variety of ways of approaching this. To make
     the initial selection for "chosen" there have to be at least 6
     volunteers.  To swap them for a new choice, must have at least 6 MORE
     new ones.
  */

  oo <-- query = oo := chosen

END
```

(b)
```
INVARIANT  chosenr : iseq(SID) & volunteersr : iseq(SID) &
           ran(chosenr) = chosen & ran(volunteersr) = volunteers
```

(c) Again, scope for various different approaches here.

```
REFINEMENT
   OpenDay_r
REFINES
   OpenDay

VARIABLES
   volunteersr, chosenr

INVARIANT volunteersr : iseq(SID) & chosenr : iseq(SID) &
          ran(chosenr) = chosen & ran(volunteersr) = volunteers

INITIALISATION
   volunteersr, chosenr := [], []

OPERATIONS

   newvolunteer(vv) = volunteersr := volunteersr <- vv;

   swap(v1,v2) =  LET ii BE ii = volunteersr~(v1)
                  IN volunteersr := volunteersr <+ {ii |-> v2}
                  END;

   /* The requirements say to make the initial selection deterministic
      but not the subsequent selection. This is one way.

   newchoice =
   IF chosenr = {} THEN chosenr := (1..6) <| volunteersr
   ELSE ANY cseq
        WHERE cseq : iseq(SID) & ran(cseq) <: ran(volunteersr) &
              card(cseq) = 6 & ran(cseq) /\ ran(chosenr) = {}
        THEN chosenr := cseq
        END
```

```
    END;

    oo <-- query = oo := ran(chosenr)

END
```

(d) Lots of possibilities here, just give some a try.