Final Examination

November 2007

COMP9116
Software System Development Using B

Time allowed: 3 hours
Total number of questions: 3

Questions are not necessarily of value stated in this paper.

Time allowed: 3 hours
Number of questions: 3
The questions are not of equal value.

The answers to this examination are to be submitted to the class account in the form of a Rodin export (zip) of your development.

All comments should be written as annotations in the machines, and commenting is encouraged.
1 Directions for setting up

1) Start Rodin and establish a convenient place for your workspace.


3) You should find that you have a project named Exam2007 and within that three context machines: Flight_ctx, Booking_ctx and Time_ctx, and one machine Flight.

4) You may want to establish your own perspective.

5) When you are finished you should export your project and then run: give cs9116 exam your archive

2 About this exam

Although this exam paper has allocated marks to questions, it is recognized that problems can occur during a laboratory exam, and it is difficult to determine exactly how much of this paper can be done in three hours under exam conditions. For those reasons, the examiner reserves the right to vary the marking scheme to take account of any difficulties.

Ultimately this exam will be assessed on the basis of all parts of questions completed and with fair compensation made for the general completion of the paper by all candidates. Completeness and correctness of answers will be important.
Preamble: Please read this first

In this paper you are going to be concerned with the modelling of a small airline reservation system with the following characteristics:

- flights are denoted by flight identifiers ($FLIGHT$); it should be understood that flights denote a particular route, like $Qantas001$;
- scheduled flights are assigned to particular days. Any flight is assigned at most once to any particular day;
- days are represented by natural numbers ($DAY$);
- customers booking seats identify the flight, the day and the number of seats required. The booking operation returns a unique booking identifier; no seat numbers are involved.

How the modelling will be structured

Apart from context machines, the modelling will be split into three layers:

Flight: (a machine) models flight information and events that are used to maintain that information;

Booking: (a refinement of Flight) models booking information and a single event to make a booking;

Browsing: (a refinement of Booking) models a simple user interface that allows a person to browse a number of flight options before making a booking, or perhaps rejecting all options.

Question 1 — 30 marks

1) Open $Flight\_ctx$ and $Flight.mch$.

2) The following variables are suggested:
   - $flights$ established flight identifiers ($FLIGHT$);
   - $capacity$ map from $flights$ to number of seats (maximum $maxseats$);
   - $scheduledflights$ map showing the days on which each flight is scheduled;
   - and a skeleton of such variables already exists in the Flight machine, but you may change these if you wish.

3) Give a sufficiently strong invariant.

4) Initialise the variables.

5) Specify an event $NewFlight$ with arguments $flight$ and $seats$ representing the flight identifier and the number of seats on the flight, respectively.

6) Specify an event $ScheduleFlight$ with arguments $flight$ and $day$ representing the scheduling of a flight on a particular day. The day must be in the future.

7) Check proof obligations and attempt to discharge any undischarged POs.
   If you cannot discharge POs formally you can alternatively discuss them in the machine.
Question 2 — 40 marks

1) Create a refinement of Flight called Booking.

2) Add appropriate variables and their invariant. Suggestions are:

   - bookings: the set of allocated booking identifiers (from BOOKING);
   - bookedflight: records flights booked;
   - bookedday: records days booked;
   - bokedseats: records seats booked;
   - flightbookings: records seats booked on flights on days.

3) Add an event BookFlight with arguments booking identifier, flight, day and number of seats.

   Note carefully: although this machine is a refinement of Flight, it simply adds an orthogonal state and event and does not modify the state or events of Flight.
Question 3 — 20 marks

The intention of this question is to provide a refinement of Booking called Browsing that gives a more flexible interface than BookFlight, so this machine does refine that operation. Only an outline will be given and the design will be left to you.

The notion is that there will be events that allow a purchaser to browse a number of potential flight (flight, days and seats). These potential bookings should be reserved—but not booked. A full design would initiate a countdown to remove the reservation after some time if it has not been booked. At the end of a session a customer would accept or reject the reserved flights.
Context Machines

CONTEXT Flight_ctx

SETS

\textit{FLIGHT} /* set of flight ids */

CONSTANTS

maxflights

AXIOMS

\textit{axm1} : \textit{finite} (FLIGHT)
\textit{axm2} : maxflights \in \mathbb{N}_1

END

CONTEXT Booking_ctx

SETS

\textit{BOOKING}

CONSTANTS

maxbooking

AXIOMS

\textit{axm1} : maxbooking \in \mathbb{N}_1

END

CONTEXT Time_ctx

CONSTANTS

\textit{DAY} /* set of days */
\textit{today} /* any day */

AXIOMS

\textit{axm1} : \textit{DAY} \subseteq \mathbb{N}
\textit{axm2} : \textit{finite} (\textit{DAY})
\textit{axm3} : \textit{today} \in \textit{DAY}