CSSE4603/7032 Models of Software Systems

Assignment 2: Go Card

Samuel Teed 43211915

April 22, 2016

Let CLASS denote the type a card can be and PEAK indicated whether the time is during peak or off peak. The peak variable can also be used by the system to aquire the discounted fare.

$$\begin{array}{l} CLASS ::= Adult \mid Child \mid Concession \mid Senior \\ PEAK ::= On \mid Off \end{array}$$

Also let the set of locaitons a card can be used at be denoted by:

$$[PLACE]$$

When cards are touched on or off the time of the event will be recorded. To know if the current journey segment is within an off peak time a set off all off peak times needs to exist within the system.
Therefore let:

$$[TIME]$$

denote the set of all posible times, and let OFFPEAK be a subset of TIME. Therefore:

$$OFFPEAK \subseteq TIME$$

In regards to this system time will natrual numbers which refer to minutes.
A segment will be any two points where the card has been touched on at one point and then touched off at the other.

$$\begin{array}{|l} \hline \textit{Segment} \\ startLocation : PLACE \\ endLocation : PLACE \\ startTime : \mathbb{N} \\ endTime : \mathbb{N} \\ date : \mathbb{N} \\ \hline \end{array}$$

As an entire segment cannot be known until the card has been touched off the card will need to store the intial segment data before it reached its destiniation. This data can be kept in a PartialSegment schema.

$$\begin{array}{|l} \hline \textit{PartialSegment} \\ startLocation : PLACE \\ startTime : \mathbb{N} \\ date : \mathbb{N} \\ \hline \end{array}$$

An incentive allows cards who are involved in nine (9) journeys in a single week be given free travel for the remainder of that week. A function which can take the date of a journeys last segment and find out what week the journey occured in while keep track of this.

$$GetWeek : Segment \to \mathbb{N}$$

A Journey is a sequence of segments as well as the week which the journey took place.

Journey

$segments : \mathrm{seq}(Segment)$
$week : \mathbb{N}$

$week = GetWeek(head(segments))$

The card stores what class the card is, all of its previous Journeys as well as its current balance and the current partial segment if the card is current in transit, other wise the currentSegemnt will be null.

Card

$class : CLASS$
$balance : \mathbb{Z}$
$history : \mathrm{seq}(Journey)$
$currentSegment : PartialSegment$

When the user enters a vehicle they touch on their Go Card. The function will take in where the card has been touched on, time and date. This initial segment data is stored in the cards currentSegment variable.

TouchOn

$\Delta Card$
$location? : PLACE$
$time? : \mathbb{N}$
$date? : \mathbb{N}$

$currentSegment = null$
$\exists\, partialsegment : PartialSegment \bullet$
$\quad partialsegment.startLocation = location?$
$\quad partialsegment.startTime = time?$
$\quad partialsegment.date = date?$
$currentSegment' = partialsegment$
$balance' = balance$
$history' = history$

A function within the system will need to exist which can take start and end locations and then be able to evaluate how many zones were traveled through, therefore determining the cost of travelling between the locations.

$$FareCalculation : PLACE \times PLACE \to \mathbb{N}$$

If the card has been used in nine (9) or more journeys in the current week (Monday-Sunday), the card will not be charged for the current journey. To allow for this the IsFree function will return the numbers 1 or 0; 1 if the card has not been on nine journeys and 0 if it has. The returned number will be multipled by the amount being deducted from the cards balance, so if the card is entitled to a free journey the charge will be zero, otherwise it is unaffected.

$$
\begin{array}{l}
\mathit{IsFree} : \mathit{Card} \to \mathbb{N} \\
\hline
\forall\, \mathit{card} : \mathit{Card} \bullet \\
\exists\, \mathit{thisWeek} : \mathbb{N} \mid \mathit{thisWeek} = \mathit{Head}(\mathit{card.history}).\mathit{week} \land \\
\exists\, \mathit{weeksJourneys} : \mathrm{seq}(\mathit{Journey}) \mid \mathit{weeksJourneys} = \mathit{card.history} \mid \mathit{card.history.week} = \mathit{thisWeek} \\
\quad \#\mathit{weekJourneys} >= 9 \Rightarrow \mathit{IsFree} = 0 \\
\quad \#\mathit{weekJourneys} < 9 \Rightarrow \mathit{IsFree} = 1
\end{array}
$$

A function will need to check if the segment occured during an off-peak time; if the segment is within an off-peak time the cost will be discounted.

$$
\begin{array}{l}
\mathit{IsPeak} : \mathit{Segment} \to \mathit{PEAK} \\
\hline
\forall\, \mathit{segment} : \mathit{Segment} \bullet \\
\mathit{segment.startTime} \in \mathit{OFFPEAK} \land \mathit{segment.endTime} \in \mathit{OFFPEAK} \Rightarrow \mathit{IsPeak} = \mathit{Off}
\end{array}
$$

When the user has completed a segment they will touch off their Go Card. At this time the Touch Off function will check if the user has just copmleted a new journey or added to a current one. This is done by checking if the card has been used 3 times in the last 3 and a half hours. The cards balance is then deducted the amount of the segment multipled by any incentive discounts that might apply.

$$
\begin{array}{l}
\textit{TouchOff} \\
\hline
\Delta \mathit{Card} \\
\mathit{location?} : \mathit{PLACE} \\
\mathit{time?} : \mathbb{N} \\
\mathit{date?} : \mathbb{N} \\
\hline
\exists\, \mathit{newSegment} : \mathit{Segment} \land \\
\exists\, \mathit{newJourney}, \mathit{lastJourney} : \mathit{Journey} \bullet \\
\quad \mathit{newSegment.startLocation} = \mathit{currentSegment.startLocation} \land \\
\quad \mathit{newSegment.endLocation} = \mathit{location?} \land \\
\quad \mathit{newSegment.startTime} = \mathit{currentSegment.startTime} \land \\
\quad \mathit{newSegment.endTime} = \mathit{time?} \land \\
\quad \mathit{newSegment.date} = \mathit{date?} \land \\
\quad \mathit{lastJourney} = \mathit{Head}(\mathit{history}) \\
\quad \#\mathit{lastJourney.segments} = 3 \lor \\
\quad \mathit{Head}(\mathit{lastJourney.segments}).\mathit{endTime} <= \mathit{time?} - 210 \Rightarrow \\
\qquad \mathit{newJourney} = \langle \mathit{newSegment} \rangle \\
\qquad \mathit{history}' = \mathit{newJourney} \frown \mathit{history} \\
\qquad \mathit{balance}' = \mathit{balance} - \mathit{FareCalculation}(\mathit{newSegment.startLocation}, \mathit{newSegment.endLocation}) \times \\
\qquad\quad \mathit{IsPeak}(\mathit{newSegment}) \times \mathit{class} \times \mathit{IsFree}(\mathit{history}') \\
\quad \#\mathit{lastJourney.segments} < 3 \land \\
\quad \mathit{Head}(\mathit{lastJourney.segments}).\mathit{endTime} > \mathit{time?} - 210 \Rightarrow \\
\qquad \#\mathit{lastJourney.segments} = 1 \Rightarrow \mathit{balance}' = \mathit{balance} - \\
\qquad\quad (\mathit{FareCalculation}(\mathit{Head}(\mathit{lastJourney}).\mathit{startLocation}, \mathit{newSegment.endLocation}) - \\
\qquad\quad \mathit{FareCalculation}(\mathit{Head}(\mathit{lastJourney}).\mathit{startLocation}, \mathit{Head}(\mathit{lastJourney}).\mathit{endLocation})) \times \\
\qquad\quad \mathit{IsPeak}(\mathit{newSegment}) \times \mathit{class} \times \mathit{IsFree}(\mathit{history}) \\
\qquad \#\mathit{lastJourney.segments} = 2 \Rightarrow \mathit{balance}' = \mathit{balance} - \\
\qquad\quad (\mathit{FareCalculation}(\mathit{Tail}(\mathit{lastJourney}).\mathit{startLocation}, \mathit{newSegment.endLocation}) - \\
\qquad\quad \mathit{FareCalculation}(\mathit{Tail}(\mathit{lastJourney}).\mathit{startLocation}, \mathit{Head}(\mathit{lastJourney}).\mathit{endLocation})) \times \\
\qquad\quad \mathit{IsPeak}(\mathit{newSegment}) \times \mathit{class} \times \mathit{IsFree}(\mathit{history}) \\
\qquad \mathit{Head}(\mathit{history}') = \mathit{newSegment} \frown \mathit{Head}(\mathit{history}) \\
\mathit{currentSegment}' = \mathit{null}
\end{array}
$$

With the above specification, operations TouchOn and TouchOff can be combined to form the operation:

$Touch == TouchOn \vee TouchOff$