```
/* Global Variables */

/* total number of calls that were attempted */
total_calls = 0;
/* number of calls that were blocked at initialization */
blocked_calls = 0;
/* number of calls that got dropped during handover */
dropped_calls = 0;

/* initialize an array of 20 stations */
stations = array(Stations, 20);
/* for each of the stations, assign an id and create 10 free channels */
for i in 1 to 20 {
   stations[i].available_channels = 10;
}

/* Call Process */
CallProcess() {

   /* increase total number of calls attempted */
   total_calls++;

   /* generate base station idx using distribution X */
   curr_station_idx = random_X();
   curr_station = stations[curr_station_idx];
   /* generate car position (m) randomly in 0 to 2,000 (with equal probability) */
   car_position = random_unif_continuous(0, 2000);
   /* generate car direction randomly (with equal probability) */
   /* car_direction = -1 if car moving left, and +1 if moving right */
   car_direction = random_unif_discrete((-1, +1));
   /* generate car speed (m/s) using distribution Y */
   car_speed = random_Y();
   /* generate call duration (s) using distribution Z */
   call_duration = random_Z();

   /* check if there is a channel available to make the call */
   if (curr_station.available_channels == 0) {
      /* block the call at initialization */
      blocked_calls++;
      return;
   } else {
      /* acquire one of the available channels */
      curr_station.available_channels--;
   }

   /* calculate the distance (m) to the next station */
   if (car_direction == -1) {
      distance_till_station_exit = car_position;
   } else {
      distance_till_station_exit = 2000-car_position;
   }
```

```
  /* calculate the time (s) to the next station */
  time_till_station_exit = distance_till_station_exit / car_speed;

  /* while the call will continue as the station is exited */
  while (call_duration > time_till_station_exit) {

    /* finish the journey in the current station */
    Hold(time_till_station_exit);
    /* update the call duration left */
    call_duration -= time_till_station_exit;
    /* exit station and free up the channel */
    curr_station.available_channels++;

    /* if the car is exiting the highway */
    /* at station 1 and exiting left, or at station 20 and exiting right */
    if ((curr_station_idx == 1 && car_direction == -1) || (curr_station_idx == 20 && car_direction == 1)) {
      return;
    }

    /* handover the call to the next station */
    curr_station_idx += car_direction;
    curr_station = stations[curr_station_idx];

    /* check if there is a channel available to make the call */
    if (curr_station.available_channels == 0) {
      /* drop the call during handover */
      dropped_calls++;
      return;
    }
    else {
      /* acquire one of the available channels */
      curr_station.available_channels--;
    }

    /* the distance (m) to exit station will be the length of the station = 2,000 */
    distance_till_station_exit = 2000;
    /* calculate the time (s) to the next station */
    time_till_station_exit = distance_till_station_exit / car_speed;

  }

  /* finish the journey in the current station */
  Hold(call_duration);
  /* terminate call and free up the channel */
  curr_station.available_channels++;

  /* terminate the call process */
  return;

}
```