CZ4015 Simulation & Modelling

Assignment 2

Submitted by: Lim Zhi Qing

Matriculation Number: U2021897L

NANYANG TECHNOLOGICAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

System State Variables

```
Integer: Directions     /* number of directions */
Integer: Cells          /* number of cells */
Integer: CellSize       /* diameter of each cell */
Integer: Channels       /* number of channels per cell */
Integer List: CellCalls /* number of active calls for each cell */

Integer: TotalCalls     /* total number of calls */
Integer: DroppedCalls   /* number of dropped calls */
Integer: BlockedCalls   /* number of blocked calls */

LinkedList: FEL
```

Main Process

```
/* Initialize state variables*/
Directions = 2;
Cells = 20;
CellSize = 2000;
Channels = 10;
CellCalls = List[Cells];

TotalCalls = 0;
DroppedCalls = 0;
BlockedCalls = 0;

While (terminating condition is not true) {
    Hold(A); /* inter-arrival time */
    Create and activate a call process;
}

/* Compute simulation statistics */
```

Call Process

```
Integer: Cell = Distribution(X);
Integer: Direction = UniformDistribution(Directions);
Integer: Position = UniformDistribution(CellSize);
Integer: Speed = Distribution(Y);
Integer: Duration = Distribution(Z);
Integer: TravelTime;

TotalCalls = TotalCalls + 1;

If (CellCalls(Cell) >= Channels) {
    BlockedCalls = BlockedCalls + 1; /* block call */
    Terminate Call Process;
}
Else {
    CellCalls(Cell) = CellCalls(Cell) + 1; /* accept call */
    TravelTime = Position / Speed;
    If (Duration < TravelTime) {
        TravelTime = Duration;
    }
}

Hold(TravelTime);
Duration = Duration - TravelTime;

While (Duration > 0) {
    CellCalls(Cell) = CellCalls(Cell) - 1; /* deallocate call */
    Cell = Cell + Direction;               /* get next cell */
    If (CellCalls(Cell) >= Channels) {
        DroppedCalls = DroppedCalls + 1;   /* drop call */
        Terminate Call Process;
    }
    Else {
        CellCalls(Cell) = CellCalls(Cell) + 1; /* handover call */
        TravelTime = CellSize / Speed;
        If (Duration < TravelTime) {
            TravelTime = Duration;
        }
        Hold(TravelTime);
    }
    Duration = Duration - TravelTime;
}

Terminate Call Process;
```

Hold

```
Hold(T)
{
    Put Resume-event with time Now+T into FEL;
    Suspend execution of thread;
    Transfer to scheduler;
}
```

Scheduler

```
Scheduler
{
    If (sim not done) {
        Remove event from FEL;
        Call event handler;
    }
}
```

Resume-Event Handler

```
Resume-Event Handler
{
    Activate process from Hold/Wait point;
}
```