Software Design Document

Airport take-off time slot simulation in Python.

Beteab Gebreyesus

10/05/2018

Introduction

Airport is a busy place with vast numbers of people passing through, which in turn requires efficient processes to make it seamless. Lots of resources are mobilized according to schedules to enable this.

We can build programs that can simulate the request and delivery of services at the airports based on the events happening at a predetermined rate. By simulating/generating requests for resources we can simulate the process of servicing those requests.

Purpose

Description of the Problem

As planes taxi and takeoff, they request resources (gates, ground operations) to enable them. The request for the resources must be managed according to priorities and safety requirements. When planes are getting ready to take-off they request access to taxi in preparation for take-off from the runway.

The program we build will tackle the problem of taxiing planes that request resources. There will be priority queues set up which will keep schedule of planes taking off. The program will receive a document listing all planes that made a request and request details. The queue will empty out per time requests and priority values.

Overview:

The application will work to schedule planes that submit request for take-off. They will inform the driver/controller the request details such as how long they need the runway for. It will assign take-off time in the day and will keep track of planes that take-off. We will have a print out of days take-off activity in order of takeoff

Implementation

- Read data from .txt file
- Create a priority Queue that will store data for takeoff request
- To read requests document and populate the queue
- Insert each request from file into the right place in the chain
- Print out the status of the queue as time moves along.
- print out a listing of the actual take off times of all planes

Ex. Input file could be from.txt file containing the various data fields: Output of our read may look like this

ID, Submission Time, Requested Start, Requested Duration
Delta 160, 0, 0, 4
UAL 120, 0, 5, 4
Delta 6, 2, 3, 6

Design

The program will be written in Python. The program will take in input from the user in the form of a text file (.txt) or comma separated value file (.csv). Program defines and uses classes and functions in python to simulate the run. We will have a regular list storing all the Request objects in order they are in the document. We will implement the queue by changing the 'next' attribute in Request object, when prioritize functions confirms priority.

Classes

Request.py

The request class has getters for all attributes as well as setters for the 'actual start', 'actual end' and 'next' attributes. The two setters will change the attributes if processor finds a higher priority item.

Request
flightID = name
reqDuration = reqduration
actualStart = 0
actualEnd = 0
submissionTime = subtime
reqStart = reqstart
next = None
showFlightInfo(self):
get_submissionTime(self):
get_reqStart(self):
get_actualStart(self):
get_flightID(self):
get_actualEnd(self):
get_reqDuration(self):
set_actualStart(self, start):
set_actualEnd(self, end):
set_nextFlight(self, Nxt):

Pr_Queue.py:

Pr_Queue

scheduleList[1000]
first = None
last = None
length = 0

sizeOfQueue(self):
getfirst(self):
getlast(self):
 	
Enqueue(self, newReq):
Dequeue(self, indx):
comparePriority(current, newReq):

Airport_Driver.py

- ➔ This class is meant to be where all calls to functions are done from. Will read in file passed as parameter.
- ➔ Will also send data items as attributes to build the object
- ➔ The item will be added to an list and we will change

Airport_Driver