Vehicle Computer Simulator Psuedocode and Outline

**Objects Used:**

Car

- Pulled from common services server according to service type and position based on received order from demand server (SQL Database entry)

- Position and state reflected/updated in maps services

Map

- Reflects route and dynamic movement of car based on dispatch record

- Viewable from client side

**Algorithmic Outline:**

*Refer to refined system diagram for specific information of steps to each process*

*API Key and Client ID for Python Client Library required. API key obtained and progress awaiting on Bitbucket/Pycharm/DigitalOcean access as of 2/14/19 Sprint 2.*

function receiveAndNotify(orderFromUser){

Approximate position based on destinationAddress from orderFromUser

Use destinationAddress approximation and serviceDetails to fetch appropriate vehicle from SQL database in supply side server (SELECT \* FROM Vehicles WHERE “Position” …. AND “Service” LIKE serviceDetails)

Create dispatch record based off vehicle (vehicle info, destination address)

Send createRoute(dispatchRecord) back to supply server }

function createRoute(dispatchRecord){

Pull vehicle info and destination address from dispatchRecord

Set start point at vehicle position and end point at dispatchRecord

Send HTTP request to Directions API using start point and end point

Return response}

function displayMap(route)//Intention to be integrated to WebApps{

Instantiate map based on information from route (HTTP request to Maps API and Directions API)

//Called to client side via HTML}

function deploy(route) {

displayMap(route)

Send HTTP request to Directions API to dynamically travel route

//Made for purposes of sprint and project. Ideally sent by supply server}

*Documentation bookmarked and accessible for both Maps and Directions API*

*Current language recommendation = Python via team concesus*