ECE568 ERSS Project - Mini-Amazon / Mini-UPS Protocol Document

Group No.: 8

Members:

Project	Team Member 1	Team Member 2
Amazon	yc538	sl785
UPS	jc977	zg105
Amazon	yw491	xj52
UPS	yw520	hy201

Tech Stack

Language: **Python, Java**

Framework: **Django, Spring Boot, Vue.js**

Database: **PostgreSQL, MongoDB**

Data Format: **Google Protocol Buffer Message Format**

Interaction Description

From Amazon to UPS

Amazon **MUST** send the `hid` (warehouse id) to UPS to order a truck to specific warehouse.

Amazon **MUST** send the `hid`, `package_id`, `location_x`, `location_y`, `email`(to identify users) and a list of `AItem`.

Amazon **MUST** send the ~~`location_x`, `location_y`,~~ `truck_id`, `package_id` to the UPS if one package is loaded.

Amazon **MUST** send the `truck_id` to the UPS to inform the loaded is completed.

All the commands above **MUST** have an unique `seqnum`.

Amazon **MAY** response the `error_code` and `error_message` if encounter any argument or runtime errors. The `seqnum` of `AError` depends on which `UCommand` to response.

Amazon **MUST** transfer commands in ACommand format ~~and acks **SHOULD** be added if this command is a response to a previous UPS command~~.

From UPS to Amazon

~~UPS **SHOULD** response to the Amazon with the truck_id for the pickup request.~~

UPS **MUST** send the truck_id and hid to the Amazon after the truck arrived to the specific warehouse.

UPS **MUST** send the package_id of package to Amazon if the package is delivered.

All the commands above **MUST** have an unique seqnum to locate the error.

UPS **MAY** response the error_code and error_message if encounter any argument or runtime errors. The seqnum of UError depends on which ACommand to response.

UPS **MUST** transfer commands in UCommand format ~~and acks **SHOULD** be added if this command is a response to a previous Amazon command~~.

Protocol Specification

```
syntax = "proto2";

//pick up request from amazon
message APickupReq{
      required int32 hid = 1;
      required int64 seqnum = 2;
}

//UPS response for pickup request
message UPickupRes{
      required int32 truckid = 1;
      required int64 seqnum = 2;
}

//UPS arrived notification
message UArrived{
      required int32 truckid = 1;

      required int32 whid = 2;
      required int64 seqnum = 3;
}

//
message AItem{
      required int32 itemid = 1;
      required int32 num = 2;
```

```
        required string name = 3;
        required string desc = 4;
}

//Amazon load request for a specific package
message ALoad{
        required int32 truckid = 1;
        required int32 hid = 2;
        required int64 packageid = 2;
        required int32 location_x = 4;
        required int32 location_y = 5;
        required int64 seqnum = 6;
        required string email = 7;
        repeated AItem itemInfo = 3;
}

//Amazon create package request
message ACreatePackage{
        required int32 hid = 1;
        required int64 packageid = 2;
        required int32 location_x = 3;
        required int32 location_y = 4;
        required int64 seqnum = 5;
        required string email = 6;
        repeated AItem itemInfo = 7;
}


//Amazon Complete Load
message ALoadComplete{
        required int32 truckid = 1;
        required int64 seqnum = 2;
}

//UPS package delivered
message UDelivered{
        required int64 packageid = 1;
        required int64 seqnum = 2;
}

//UPS Error code and message
message UError{
        required int32 code = 1;
        optional string msg = 2;
        required int64 seqnum = 3;
}

//Amazon Error code and message
message AError{
        required int32 code = 1;
```

```
      optional string msg = 2;
      required int64 seqnum = 3;
}

//Amazon command packet
message ACommand {
  repeated APickupReq pickups = 1;
  repeated ALoad toload = 2;
  repeated ALoadComplete comp = 3;
  repeated ACreatePackage create = 4;
  repeated AError error =5;

  repeated int64 acks =5;
}

//UPS command packet
message UCommand {
 repeated UPickupRes upickupRes = 1;
 repeated UArrived uarrived = 1;
 repeated UDelivered udelivered = 2;
 repeated UError uerror = 3;
 repeated int64 acks = 5;
}
```

Bare Minimum Functionality