ECE 568 Final Project Protocol Specification

Interoperability Group 2

Group 5	Mac Gray (mjg85)	Yuyu Hsieh (yh384)	UPS
Group 6	Jingxuan Li (jl1226)	Kaixin Lu (kl461)	Amazon
Group 7	Chelsea Lyu (jl1230)	Yijia Zhao (yz853)	UPS
Group 8	Qianqian Yu (qy44)	Sitang Gong (sg669)	Amazon

Message Format

All the messages between UPS and Amazon **MUST** be sent using the Google Protocol Buffer Message Format.

Communication Scenarios

There are 5 main scenarios that require communication between Amazon and UPS. See the Illustration Graph at the end of the document for more information.

1. Setup initial connection (Amazon → UPS)
2. Truck Request (Amazon → UPS)
3. Truck Arrived notice (UPS → Amazon)
4. Deliver Package request (Amazon → UPS)
5. Package Delivered notice (UPS → Amazon)

The exact message format for each communication is described in detail below.

1. **Setup initial connection to send World ID from Amazon to UPS.**
   When amazon initializes a world, amazon will send world Id to UPS so that ups connect to the same world.
   The message **MUST** contain:
   - a world Id number (world_id)
   - a sequence number as a unique identifier for the request (seq_num)

```
message ConnectWorldId{
  required int32 world_id = 1;
  required int64 seq_num = 1;
  required int64 seq_num = 2;
}
```

2. **Truck Request from Amazon to UPS.**
   When a package is packed and prepared for shipping, Amazon sends a request to UPS to assign a truck for transporting this package.
   The message **MUST** contain:
   - a sequence number (*seq_num*) for ack purpose
   - the id of warehouse (*warehouse_id*) to let UPS know which warehouse to go to
   - the location of warehouse (*warehouse_x*, *warehouse_y*) to let UPS know where the warehouse is

- the destination location of the package (*dest_x*, *dest_y*)
- the shipment id (*ship_id*) to let UPS knows which package to pick

The message **MAY** contain:

- an UPSOrder, which contains UPS user id and the product user purchases, if the Amazon user has an UPS account (see details below)

```
message RequestTruck{
  required int64 seq_num = 1;
  required int32 warehouse_id = 2;
  required int32 warehouse_x=3;
  required int32 warehouse_y=4;
  required int32 dest_x = 5; //destination
  required int32 dest_y = 6; //destination
  required int64 ship_id = 7;
  optional UPSOrder ups_order = 8;
}
```

UPS Order Information

When Amazon sends a request for a truck, Amazon **MAY** send an order information to UPS. The message **MUST** include:

- UPSuserId: the user account id of UPS related to this order
- product: an Amazon product info included product id, description, and product number. This product info is related to this order and **MUST** be available inside the corresponding UPS user account.

```
message UPSOrder{
 required string UPSuserId = 1;
 repeated AProduct product = 2;
 repeated AProduct product = 2;

}

message AProduct{
message Product{
 required int64 productId = 1;
 required int64 productCount = 2;
 required string productDescription = 3;
}
```

3. **Truck Arrived Message from UPS to Amazon.**

When a truck has arrived at the correct warehouse, UPS sends a message to Amazon that the truck can be loaded.

The message **MUST** contain:

- a sequence number (seq_num) to support ACK functionality

- the shipment id (ship_id) to let Amazon know which packages to load onto the truck
- the truck id (truck_id) to let Amazon know which truck to load the packages onto

```
message TruckArrived{
  required int64 seq_num = 1;
  required int64 ship_id = 2;
  required int32 truck_id = 3;
}
```

4. **Deliver package request from Amazon to UPS**.
   When Amazon gets the message from the warehouse that the packages have been loaded, it will send a "DeliverPackage" request to UPS, meaning the package is ready to be delivered. Each DeliverPackage message **MUST** include a seq_num(to ensure the uniqueness of each delivery request) and a ship_id(to identify the specific package to be delivered), both marked as required.

```
message DeliverPackage{
  required int64 seq_num = 1;
  required int64 ship_id = 2;
}
```

5. **Package Delivered Message from UPS to Amazon:**
   When each package is delivered by truck, UPS **MUST** send a delivery message to Amazon. The message **MUST** include:
   - seqnum: a request sequence number
   - ship_id: a shipment id to identify each package

```
message U2ADelivered{
  required int64 seq_num = 1;
  required int64 ship_id = 2;
}
```

6. **The Final Destination Message from UPS to Amazon:**
   Once UPS sends a message "UGoDeliver" to the World, UPS users are not allowed to modify the destination of the package, so UPS sends the final destination of the package to Amazon, allowing it to track the destination of each package.
   The message **MUST** include:
   - seq_num: a request sequence number
   - ship_id: a shipment id to identify each package
   - the destination location of the package (*dest_x*, *dest_y*)

```
message FinalDest{
  required int64 seq_num = 1;
  required int64 ship_id = 2;
  required int32 dest_x = 3;
  required int32 dest_y = 4;
}
```

Communication Messages

The 2 - 4 communication scenarios described above will be send inside one of the 2 command messages:

1. AtoUCommands for communication (Amazon → UPS)
2. UtoACommands for communication (UPS → Amazon)

The exact format is described below.

1. **Commands from Amazon to UPS.**
   All messages sent from Amazon to UPS **MUST** be in this format, including requests and responses. Each AtoUCommands can present one or both types of commands and it can ask for several repeated commands for both RequestTruck and DeliverPackage. To handle potential message loss, acknowledgements (acks) are used to ensure successful receipt of messages. If errors occur during communication from UPS to Amazon, Errs will be sent back. (see detailed error description below)

   ```
   message AtoUCommands{
     repeated RequestTruck truckReqs = 1;
     repeated DeliverPackage delivReqs = 2;
     repeated int64 acks = 3;
     repeated Err error = 4;
   }
   ```

2. **Commands from UPS to Amazon.**
   All the messages UPS sends to Amazon **MUST** follow the **UtoACommands** format for Amazon server to recognize and process.
   - If the message is used to notify Amazon that a truck has arrived at the requested warehouse for pickup, it **MUST** contain one or more TruckArrived messages.
   - If the message is used to notify Amazon that a certain package has been successfully delivered, it **MUST** contain zero or more U2ADelivered messages.
   - If any of the previously sent AtoUCommands failed to meet the specified message format, an Err message **MUST** be sent via a UtoACommands to notify Amazon of the mistake. (see detailed error description below)
   - If the message serves as a response for a previously sent AtoUCommands, it **MUST** contain one or more acknowledge number (acks) which matches the seqnum of the previously sent AtoUCommands to let Amazon know that the AtoUCommands have been successfully received.
   - Overall the UtoACommands **MAY** include one or more elements of the same type or different types, but **MUST** include at least one element inside.

   ```
   message UtoACommands{
     repeated TruckArrived arrived = 1;
     repeated U2ADelivered delivered = 2;
   ```

```
    repeated Err error = 3;
    repeated int64 acks = 4;
    repeated FinalDest dest = 5;
}
```

Error Message is included in AtoUCommands and UtoACommands if an error happens.
Error indicates that you failed to meet any of the **MUST** requirements specified at “UPS Commands details” or “Amazon Commands details” above.
The message **MUST** contain:

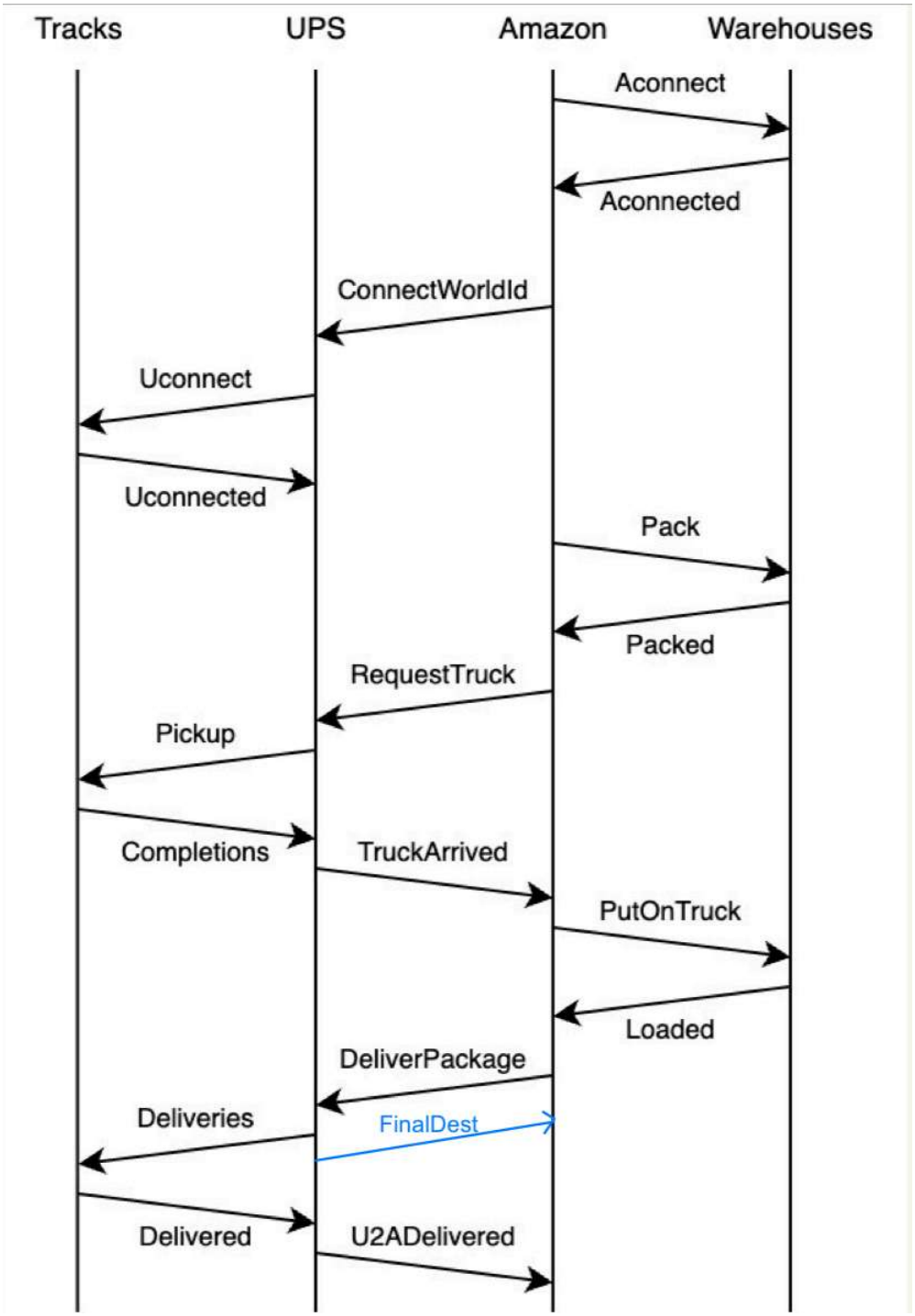
- a string(err) to describe your error
- an original sequence number(originseqnum) to clarify which request encounter error
- an sequence number(seqnum) to clarify its own request id

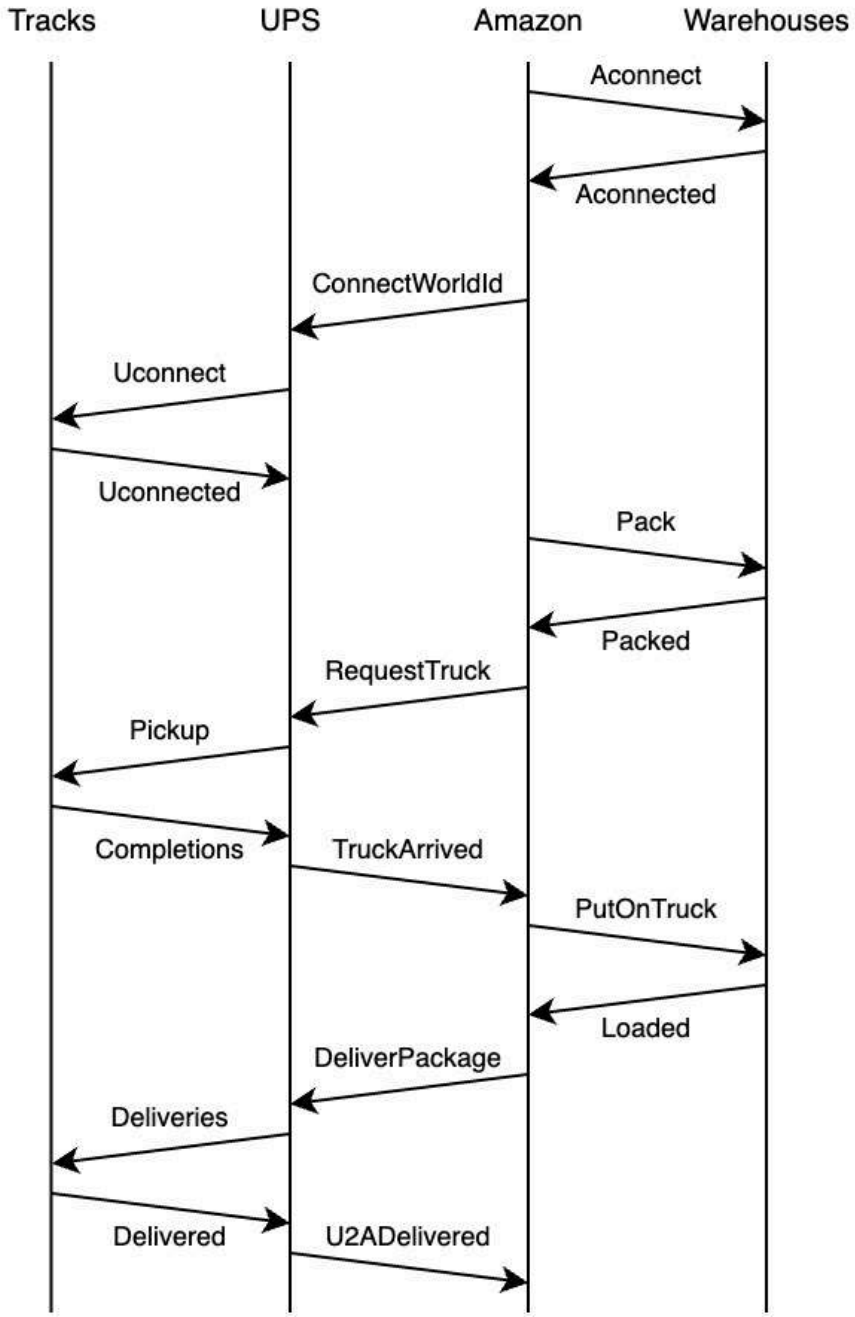
```
message Err{
    required string err = 1;
    required int64 originseqnum = 2;
    required int64 seqnum = 3;
}
```

Acknowledgement Mechanism

The protocol has an acknowledgement mechanism to ensure that every request is received at least once and processed at most once. Every request from Amazon to UPS (RequestTruck and DeliverPackage) and UPS to Amazon (TruckArrived and U2ADelivered) **MUST** contain a seqnum which serves as an unique identifier for each request. Once the other party successfully receives the request message, it **MUST** respond with an acks matching the original seqnum to signal that the message has been delivered. The sending party **MUST** continue resending the request if they did not receive a matching acks response. The receiving party **MUST** keep track of the unique seqnum to make sure it doesn’t act on the same request more than once.

Communication Scenario Illustration Graph




Tracks
UPS
Amazon
Warehouses
Aconnect
Aconnected
ConnectWorldId
Uconnect
Uconnected
Pack
Packed
RequestTruck
Pickup
Completions
TruckArrived
PutOnTruck
Loaded
DeliverPackage
Deliveries
Delivered
U2ADelivered