DangerLog

Handle asynchronous messages

Our backend needs to communicate with our front-end, UPS server, and the World Simulator; with each side, we send requests and receive responses asynchronously. So we use three threads two handle each side communication, and at each thread, we handle each requests in multi-thread processing.

Handle the ack mechanism of the world simulator

Keep track of the incrementing of sequence number coming from our amazon: a map to store the sequence number of each request we sent and the timer of this request. Since world send back ack to us when it receives our requests, so when we receive the ack, we will remove this sequence number and timer from the map and cancel this timer. If we do not receive this ack, we will resend it to the world simulator, and we set the timeout value, when the timer exceeds this value, we will resend our requests.

Handle world asynchronous

Since world will drop packages, so we need to handle the situation when it does not receive our requests or ack, it will resend the message to us. So we use appropriate identifiers in the response to figure our what request a message is in response to; we use a hashset to store the sequence number of the received responses, and every time when we receive a response, we will firstly check if it is in this set, if so, do not handle it again, if not, handle it and add it to this set.

When and How backend handles the requests from front-end

When our customers place an order in our website, our backend needs to know that and handle this request. So we use TC socket to connect backend and frontend, when customers place orders, front-end send a message to our backend.

Dealing with front-end repeated order

The user could refresh pages when using our website. For example, refreshing checkout page may create multiple orders.

Solution: adding code to check the database, validate every time a customer wants to refresh a page or submit a html form. If the order has been placed, push a prompt on the page and ignore the checkout operation.

Only our registered user can buy at our website

Our home page will show the recommendation of the products and catalog list, but if customers do not login, they cannot buy it!

Avoid the same packageid

When the web meets some error, it may send the same package id to backend, but the package id needs to be unique, so we avoid it via front-end, we add a check condition to handle it.