**RMS Implementation Phase 2**

PendingOrder and OpenOrder:-

In these modules we basically create a class used to contain order data

Pending orders are those which have not yet been acknowledged by exchange

Open orders are those which have been acknowledged by exchange as open

Exposure:-

In this module we store buy and sell prices, quantities, used to calculate exposure.

This module contains a function to return MTM utilization (for that token) based on above stored values.

InpOut:-

This module just contains parameters for input and output format for functions “SET\_INSTANCE\_LIMITS” and “GET\_UTILIZATION”

InstanceData:-

In this module we store a particular instance’s data which includes its SOV,POVetc,. Limits and utilizations.

Function reset sets all utilizations to 0.

Functions beginning with handle basically implement the rules to verify where a given order value violates that limit.

Function handleAll calls all other handle functions. It return true(“SUCCESS”) if no limit is breached within specified limits to check in String notFrom. Otherwise “it returns “FAILURE”

Function SOV makes sure only SOV limit is breached.

Function POV\_TOV\_POS makes sure only their respective limits are breached.

Function EXP makes sure only Exposure limit is breached.

Function PR makes sure only LTP limit is breached.

ApiImplementation:-

This module provides carious functions to get input orders requests , acknowledgements, trade messages both from RMS and Exchange.

SET\_INSTANCE\_LIMITS :- sets the breach limit for various rules in RMS.

RESET\_UTITLIZATION:- sets all the utilizations of various rules to 0.

GET\_UTILIZATION:- gives the current utilization of all the necessary rules.

ORDER\_REQUEST:- takes a order request, changes utilizations iff they increase and keeps the order in pending state. Also updates LTP.

ORDER\_TRADE :- takes a order Trade, changes utilizations on open order change open orders quantity for partial trade and removes open orders for complete trade.

ORDER\_ACK :- accepts acknowledgement of orders from exchange and changes utilizations (if not changed already)

GET\_PRICE\_QTY:- accepts base price and quantity, LTP and rejection type and tries to breach through that rejection type (if not already breached) increasing quantity first, and if that breaches 10% of Limit then price and quantity are changed accordingly. If breach of that single rule is not possible without breaching others then FAILURE is returned, otherwise the new quantity, price are returned.

Helper function to update netposition, exposure, etc,. are also present to reduce the code and make it more readable.