# Exchange Rate Tool Design

The Exchange Rate Tool (ERT) is a single threaded application that queries specific crypto exchanges for the USD/HBAR exchange rate, calculates a median, performs a smoothing operation if out of bound from the previous midnight rate value and distributes that median to the hedera network and other tools. The ERT has a scheduled thread that awakes and executes [this schedule is implemented using AWS lambda and CloudWatch trigger] at a configured interval the following procedure

1. Query a number of crypto exchanges for current USD<->HBAR exchange rates.
2. Calculate the median of the exchange rates.
3. Detect anomalous values, if out of bound smooth it.
4. Create an exchange rate JSON file using the median calculated above.
5. Network
   1. Create the transaction to update the exchange rate file
   2. Sign the transaction to update the exchange rate file
   3. Submit the transaction to the network
6. Pricing DB
   1. POST the exchange rate file to the DB using a REST API
7. Log ALL activity

## Exchange Rate File

Format

[{

"CurrentRate":{

"hbarEquiv" : 30000,

"centEquiv" : 360000,

"expirationTime" : 1575747397

},

"NextRate":{

"hbarEquiv" : 30000,

"centEquiv" : 450000,

"expirationTime" : 1591558597

}

}]

# Algorithm

At time t0, there is an existing exchange rate ernow that is active (on the network & tools) and will expire at te. Both the network & the ERT know ernow and te..By default, nodes will use this value for the exchange rate indefinitely (I.e nodes won’t expire the exchange rate unless there is a new value).

The ERT follows the algorithm below in creating & distributing a new exchange rate to the network nodes.

1. At time t0, ERT queries exchanges & calculates the median exchange rate ernew
2. Create an exchange rate JSON file (formatted as above) with
   1. CurrentRate/centEquiv = ernow
   2. CurrentRate/expirationTime= next hour (epoch time)
   3. NextRate/centEquiv = ernew
   4. NextRate/expirationTime = next next hour (epoch time)
3. Convert JSON to protobug
4. Create & sign the FileUpdate tx using the protobuf exchange rate file
5. Submit FileUpdate to network
   1. Nodes update exchange rate in memory. Nodes now have both the current and next rates, and when to transition from former to latter.
6. At time t0 + f, ERT repeats Step 1, using the newly calculated exchange rate as ernew, and the previous ernew as ernow

# Network Details

Exchange Rate File ID: 0.0.112

Account that sends the Exchange Rate FileUpdate transaction: 0.0.57

FileUpdate transaction should have fee set to 0

On the initial run, the transaction is sent to a network node/nodes mentioned in the config file. In each run we fetch the address book of that network and use it in the next run as the network node/nodes to which we send the transactions so that we always have an updated addresses of the nodes that we want to communicate with.

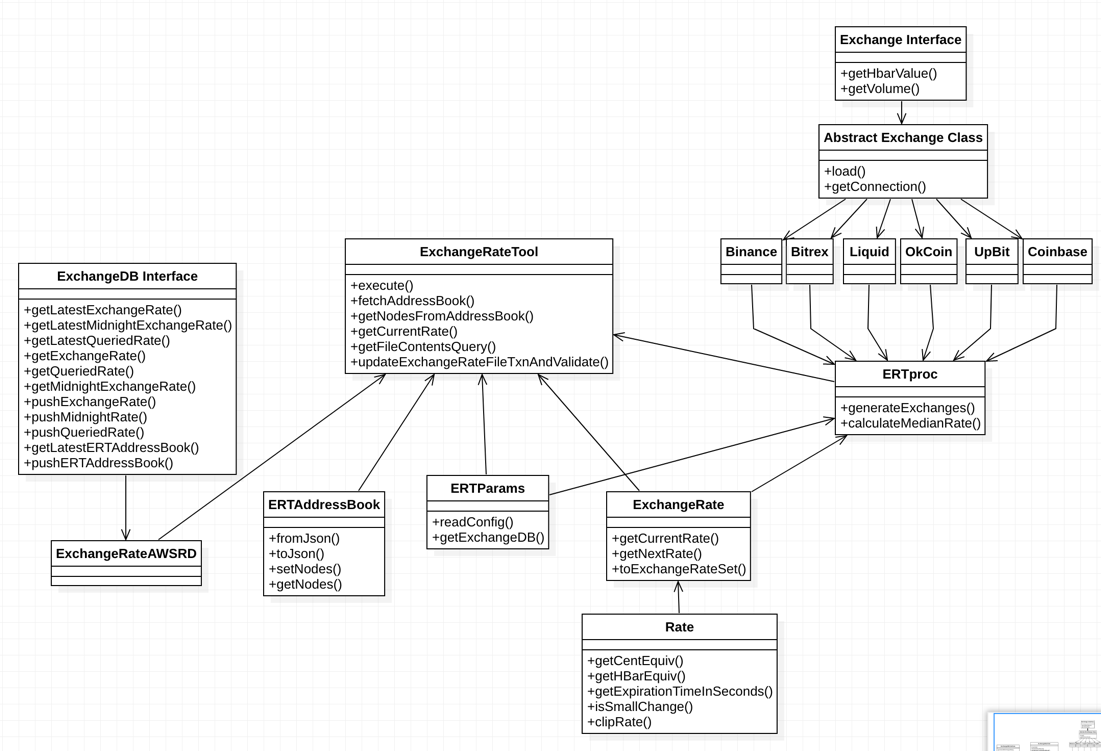
# Anomaly Detection

The ERT will compare each calculated median exchange rate to the median exchange rate we used at the last midnight UTC and determine if the difference is greater than the specified max delta percentage. If it is out bound of this delta, we smooth it to that delta limit.

# Retry on Error

If ERT fails to successfully upload the Exchange Rate File to the network, we will rerun the whole procedure for a maximum number of 4 times and in each run choosing a different node from the addressbook.

# ERT Design:



# DB Design:

