Client Module

Note : X_y indicates that y has digitally signed the X using the private keys.

Note : Decryption of keys implies that the digital signatures are being verified using the public keys.

Note: Hash(X) means that it is the cryptographic has function of the variable X.

Note: "Keys" include both public keys and private keys. Public keys are broadcasted to everyone whereas private keys are given to respective elements.

Application sends a request to the client to perform the operations

1. Client :On Receiving(request,o)

   //Client has to perform an operation 'o'

   //sends configuration request to the Olympus periodically for a new configuration

   Send (request ,"Configuration") to Olympus

   //Waits until the Olympus sends the configuration of the replicas and keys

   Await(Receive(response,"Configuration") and Receive(response,"keys") from Olympus)

2. Client : On receiving(response,"configuration") and (response,"keys")

   //i_c is the unique identifier of the client to the replicas

   //Client sends the operation along with its unique identifier to the head and also //receives the new request from the application

   Send(request, o ,i_c,"initial_transmission")$_{client}$ to the head

   o = get next_request from the application

   //client starts the timer and waits for the result proof

   Timer.start()

   Await(Receive(response,result_proof)$_{tail}$ or timer.expires())

3. Client : On receiving(response,result_proof,r)

   Decrypt the request from the tail using the keys
   //Client verifies the result and sends a reconfiguration request if it detects the proof of //misbehaviour

```
If ( Validate_result(result_proof,r)):
        Timer.stop()
Else:
        Send(request,"Re-configuration" ,Configuration) to Olympus
```

4. Client : On receiving(response,"error")

```
//If it receives the error message from the replica then it sends the re-configuration request to //the Olympus to get the latest configuration
Send(request,"Re-configuration" ,Configuration) to Olympus
```

5. Client: When Timer expires

```
//Send the operation to all the replicas
Send(request,o,i_c,"retransmission")client to all the replicas
```

```
Def Validate_result(result_proof,r):
        //Count is for counting the no. of correct replicas
        Count = 0
        For r_replica in result_proof:
                //If r_replica in result proof differs with the result r , return false.
                If r_replica != r:
                        Return False
                // If r_replica in result proof matches with the result r , increment the count
                //by 1
                Count+=1
        // If no. of correct replicas are less than t+1 , return false
        If count < t+1:
                Return False
        Else:
                Return True
```