Tendering process

Core Idea

The core idea is to distribute the contract tender mechanism in phases with each phase being implemented using a smart contract. We define a few states in the contract tendering to ensure fairness in the process: OPEN, CLOSED, FINISHED. It prevents the owner and users from executing functions that cannot be executed in that particular state.

Bidding Phase

In this phase, the organizations/individuals can submit their tender in a sealed envelope-like mechanism where his bid is not known to anyone or any other bidder. An option to revise the tender is given where an individual/organization can revise their tender as many times as they please.

The bidder will have to provide a quotation on submitting the tender and cannot submit a tender without a quotation. The blockchain stores the smart contract takes the tender and registers it. In the situation of a revised tender, he must submit a quotation lower than the previous quotation otherwise there is no meaning of revising it.

There will be no option of revoking a submitted tender. In the case of an organization/individual who has submitted a tender and got the contract do not want to abide by it by withdrawing, such an option will not be entertained. A submitted tender cannot be revoked.

This tender submission process is kept as a secret and can only be accessed by that bidder as the hash of the smart contract is visible to others and not the actual link of the vote with the voter. The current bid is also not displayed to a generic bidder. The bidder can only put his bid or revise it and wait for the results to come. This will always result in a fair and impartial contract tender.

To submit the tender, no fees are required, only a quotation is required

In this phase the contract tender process is OPEN.

Contract Announcement Phase

In this phase the contract tender process is CLOSED.

The blockchain has stored all the quotations entered by hundreds of organizations/individuals who have submitted their tenders.

The contract tender is awarded to the organization/individual who submits the lowest quotation in their tender.

The lowest quotation is computed among all the quotations submitted in the tenders. In the case of revised tenders that are lower than the original tender, it is considered.

As the quotation values have been kept secret to the bidders, the situation of 2 or more organizations/individuals submitting the same value of lowest quotation. However, the contract can be awarded to only an individual entity.

Thus, if there are multiple such cases of the same lowest quotation, we choose a single random bidder among all the bidders and award the contract to him. The random function will choose among the bidders with the lowest quotation. Since it is a random function, all the bidders have an equal probability to be chosen. This ensures that fairness is not lost during the computation of the contract awardee.

After the lowest quotation is obtained, the contract is announced to the specific organization/individual. The payment for the contract is given to the contract awardee which completes the contract tender process.

It terminates with the FINISHED state as a tender contract has been awarded and money has been transferred.

We also implement a change state mechanism to prevent changes that are prohibited which ensures that no action is taken to spoil the contract tender system. The only valid state change is from OPEN to CLOSED to FINISHED. All other state changes are invalid and need to be disregarded.

Smart Contract Implementation

```
// Create a smart contract
contract Tender{

    // Specified number of bidders
    uint256 NO_OF_BIDDERS = MAX_BIDDERS;

    // Infeasible large quotation to replace with smaller values
    uint256 MAX_VAL = 10000000000000000000000000000000000;

    // OPEN = 1, CLOSED = 2, FINISHED = 3
    // Only valid order is 1 -> 2 -> 3
    // Initially Tender is OPEN
    uint Tenderstate = 1;

    // Stores the lowest quotation entered by an individual/organization
    uint256[] quote;

    // Mark that all the bidders have not put a quotation
    for (uint256 i = 0; i < NO_OF_BIDDERS; i++){

        // Infeasible large quotation which will be replaced with smaller values
        quote.push(MAX_VAL);
    }

    // Function for individual/organization/bidder to put his lowest quotation
```

```
function bidding(uint256 quotation, address bidder){

    // Requirements

    // Tender be OPEN to submit lowest quotation
    require Tenderstate == 1;

    // Revised quotation must be lower than the previous
    // Initial quotation will always be accepted as quote[bidder] is set to INFINITE in the beginning
    // Any quotation will be less than INFINITE, so it will be accepted
    require quotation < quote[bidder];

    // Set/Revise the quotation of the bidder
    quote[bidder] = quotation;
}

// Announce the contract awardee
function contract(){

    // As the contract awardee is being computed, we cannot consider more quotations
    // Tender is CLOSED
    changestate(2):

    // Tender must be CLOSED to evaluate the contract awardee
    require Tenderstate == 2;

    // Set the value of the lowest quotation to an infeasible large value
    // It will be replaced with smaller values
    uint256 lowest_quotation = MAX_VAL;

    // A simple loop to calculate the minimum value in a loop

    // Traverse through quotations of all the individuals/organizations
    for (uint256 i = 0; i < NO_OF_BIDDERS; i++){

        // Check if their quotations than the current lowest quotation
        if (quote[i] < lowest_quotation){

            // Replace the current lowest quotation with a lower quotation
            lowest_quotation = quote[i];
        }
    }

    // Since we have replaced the current lowest quotation with lower values, at the end
    // We have no value lower than this (otherwise it would have been replaced)
```

```
// lowest_quotation contains the least quotation among all the bidders

// To store the addresses of all the bidders who bid the lowest quotation
address[] awardees;

// Traverse through quotations of all the individuals/organizations
for (uint256 i = 0; i < NO_OF_BIDDERS; i++){

    // The condition when the bidder bid the lowest quotation
    if (quote[i] == lowest_quotation){

        // Add to the list of lowest quotations
        awardees.push(i);
    }
}

// We have a list of all bidders who gave the lowest quotation value

// Base case
// If there is only a single bidder with lowest quotation
if (awardees.length == 1){

    // Contract is awarded to that bidder
    uint256 tender_given = 0;
}

// Case where 2 or more bidders with the lowest quotation
// We need to randomly generate the contract awardee among all possibilities
// This gives an equal probability of awarding the contract for all the bidders
// Ensures a contract is awarded is declared in this situation
else{

    // Generate an integral random number using any function within the range
    uint256 tender_given = generaterandomnumber(0, awardees.length);

    // Verification within the indices
    require 0 <= tender_given <= awardees.length;
}

// Obtain the address of the contract awardee from index
address contractor = awardees[tender_given];

// Give the contract to the contract awardee
// Payment of the contract money to the awardee
payable(contractor.transfer(lowest_quotation));
```

```
        // Contract awardee is displayed
        return contractor;

        // As the contract awardee has been computed and the payment has been made
        // Contract Tender process is FINISHED
        changestate(3);
    }

    // Valid change of the Tender state
    function changestate(uint state){

        // Either we can move 1 -> 2 or 2 -> 3
        require (Tenderstate == 1 && state == 2) ||
                (Tenderstate == 2 && state == 3);

        // Change the current Tender state
        Tenderstate = state;
    }
}
```