Decentralized Open Source Sponsorship

aka OpenSourcePizza
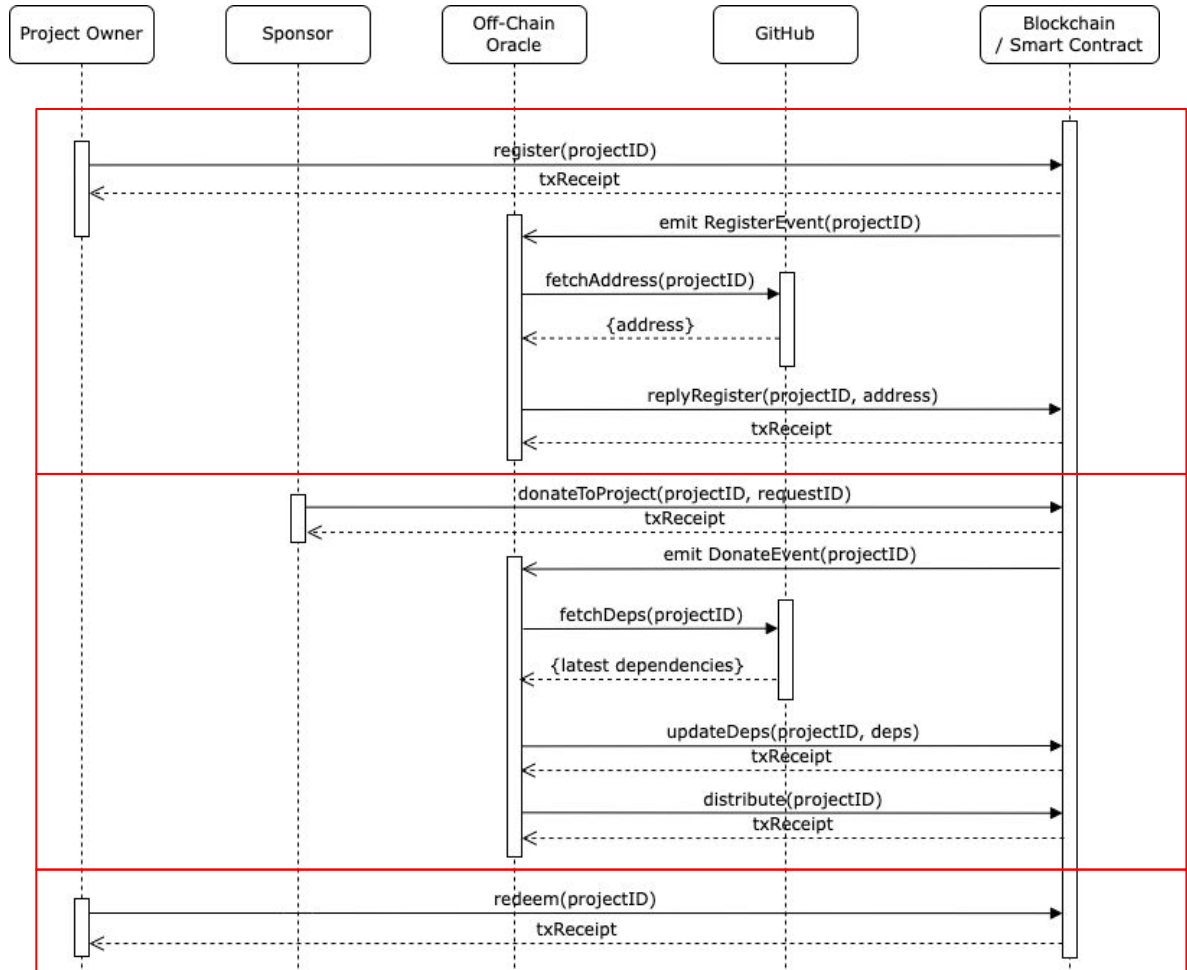
share the love for open source projects

Background

Problem:

Open source projects are not getting properly funded. Projects that are fundamentally important and referenced a lot by other projects do not necessarily receive the support that they deserve
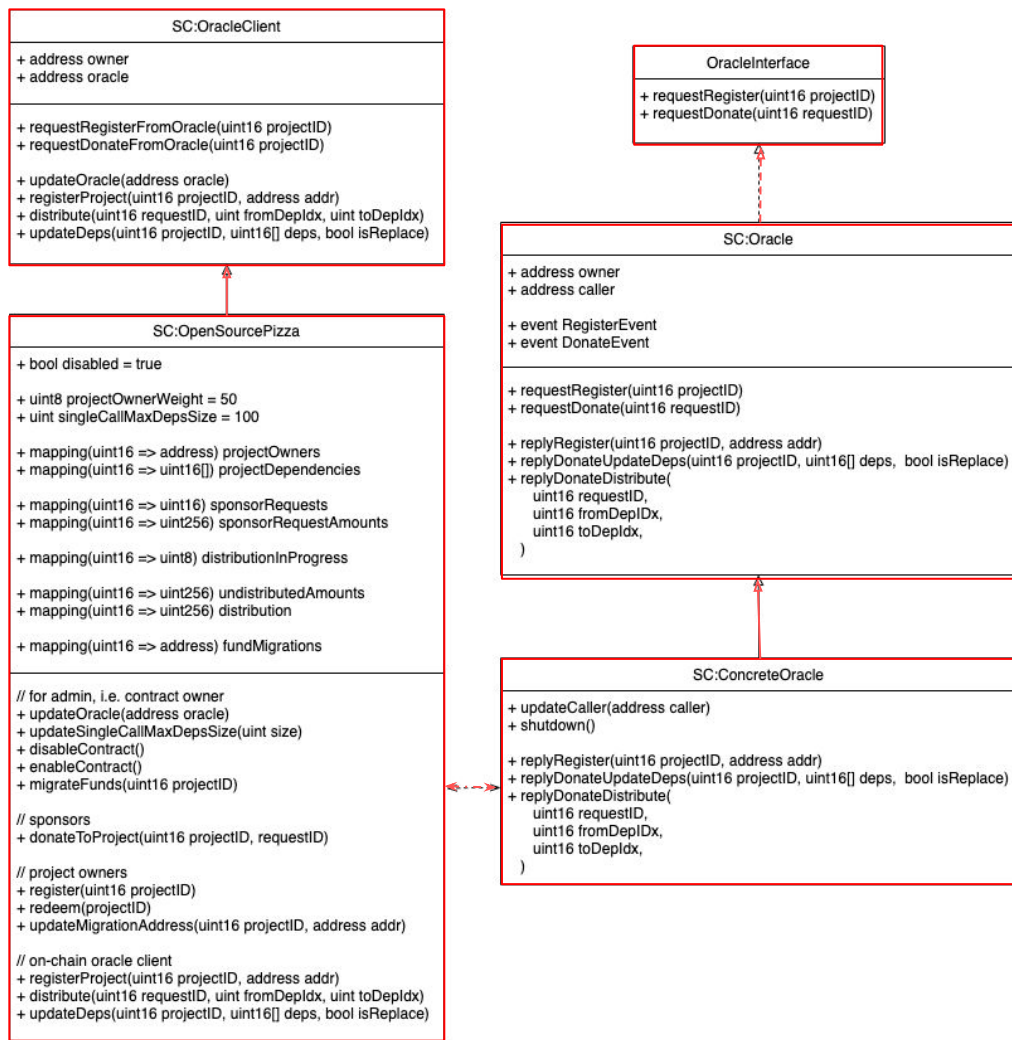
Proposal:

Distribute donations to the sponsored project and its dependent projects, so that projects that get referenced can also be benefit from indirect sponsorships.
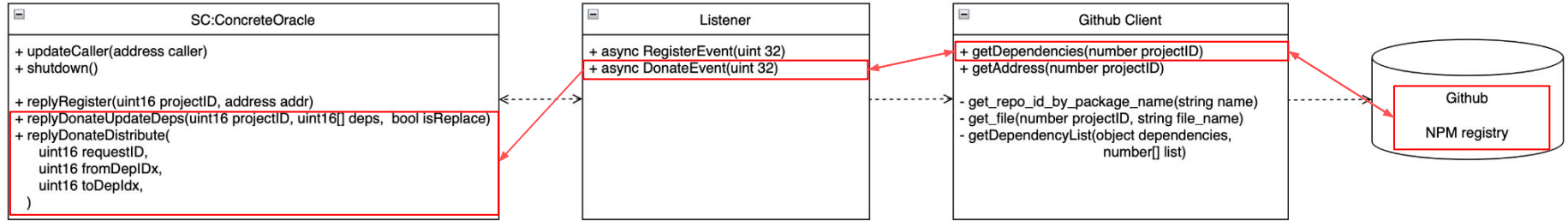
System Architecture

System Architecture
On-chain Components

SC:OracleClient

+ address owner
+ address oracle

+ requestRegisterFromOracle(uint16 projectID)
+ requestDonateFromOracle(uint16 projectID)

+ updateOracle(address oracle)
+ registerProject(uint16 projectID, address addr)
+ distribute(uint16 requestID, uint fromDepIdx, uint toDepIdx)
+ updateDeps(uint16 projectID, uint16[] deps, bool isReplace)

SC:OpenSourcePizza

+ bool disabled = true

+ uint8 projectOwnerWeight = 50
+ uint singleCallMaxDepsSize = 100

+ mapping(uint16 => address) projectOwners
+ mapping(uint16 => uint16[]) projectDependencies

+ mapping(uint16 => uint16) sponsorRequests
+ mapping(uint16 => uint256) sponsorRequestAmounts

+ mapping(uint16 => uint8) distributionInProgress

+ mapping(uint16 => uint256) undistributedAmounts
+ mapping(uint16 => uint256) distribution

+ mapping(uint16 => address) fundMigrations

// for admin, i.e. contract owner
+ updateOracle(address oracle)
+ updateSingleCallMaxDepsSize(uint size)
+ disableContract()
+ enableContract()
+ migrateFunds(uint16 projectID)

// sponsors
+ donateToProject(uint16 projectID, requestID)

// project owners
+ register(uint16 projectID)
+ redeem(projectID)
+ updateMigrationAddress(uint16 projectID, address addr)

// on-chain oracle client
+ registerProject(uint16 projectID, address addr)
+ distribute(uint16 requestID, uint fromDepIdx, uint toDepIdx)
+ updateDeps(uint16 projectID, uint16[] deps, bool isReplace)

OracleInterface

+ requestRegister(uint16 projectID)
+ requestDonate(uint16 requestID)

SC:Oracle

+ address owner
+ address caller

+ event RegisterEvent
+ event DonateEvent

+ requestRegister(uint16 projectID)
+ requestDonate(uint16 requestID)

+ replyRegister(uint16 projectID, address addr)
+ replyDonateUpdateDeps(uint16 projectID, uint16[] deps, bool isReplace)
+ replyDonateDistribute(
    uint16 requestID,
    uint16 fromDepIDx,
    uint16 toDepIdx,
)

SC:ConcreteOracle

+ updateCaller(address caller)
+ shutdown()

+ replyRegister(uint16 projectID, address addr)
+ replyDonateUpdateDeps(uint16 projectID, uint16[] deps, bool isReplace)
+ replyDonateDistribute(
    uint16 requestID,
    uint16 fromDepIDx,
    uint16 toDepIdx,
)
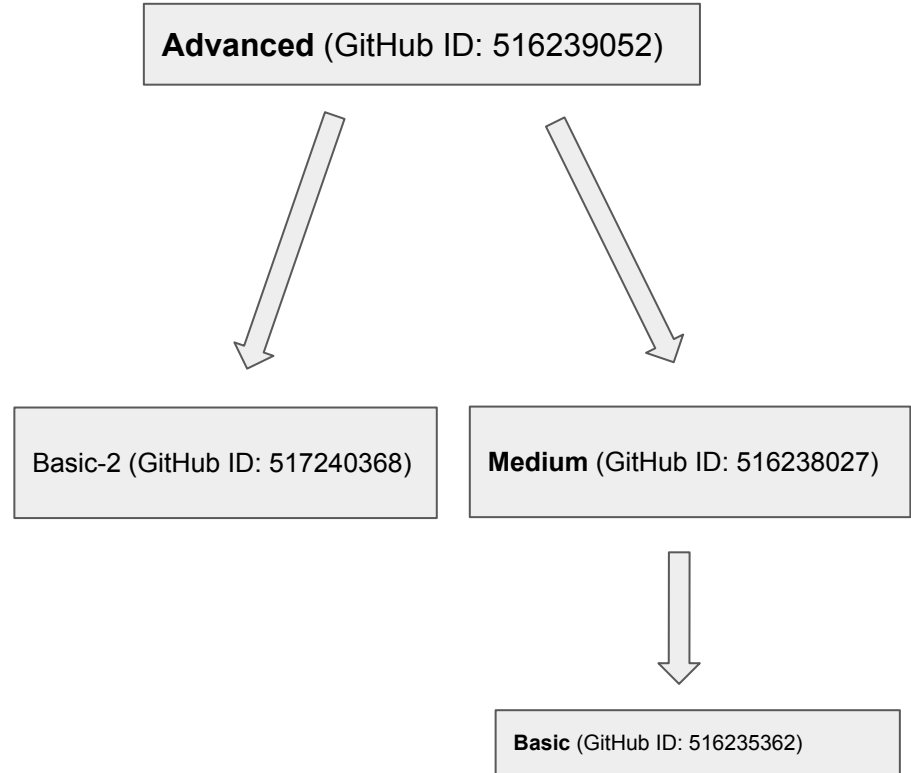
System Architecture - Off-chain Components

Demo

To demo, here is a sample use case. A project exists which has 2 dependencies, one of which has its own single dependency.

We will:

1. _Register_ Project **Advanced**, **Medium** & **Basic** (in bold)
2. _Verify_ these transactions
3. _Donate_ 30 Wei to Project Advanced
4. _Verify_ that the dependencies have been updated successfully
5. _Check_ that the distribution for project Advanced = 15 Wei, and the remaining 5 Wei
6. _Redeem_ funds successfully for Project Advanced
7. _Show_ that redeeming funds for the unregistered project does NOT work
8. _Register_ them, and redeem their 5 Wei successfully

Advanced (GitHub ID: 516239052)

Basic-2 (GitHub ID: 517240368)

Medium (GitHub ID: 516238027)

Basic (GitHub ID: 516235362)

Code Review - Patterns and Design Considerations

- Pull payment
- Check-Effects-Interaction
- Mutex
- State machine and Emergency Stop
- Ownership and Access Restriction

- Payload limit
- Remediation and Migration

Code Review - Unit Tests

- Correctness
  - donate, distribute and redeem
- Access Restriction
  - updateOracle
  - registerProject
  - updateDeps
  - distribute
  - migrateFunds
  - redeem
- Race Conditions
  - distribute to a project with single distribute function call
  - distribute to a project with multiple distribute function calls
  - distribute to a project with multiple distribute function calls, while distributing for another request referring to the same project
  - update dependencies of a project when there's distribution in progress