Blockchain for social good - Supporting the United Nations Social Development Goals

Point of contact
Shruti Appiah
shruti.appiah@iohk.io

Details of Challenge:

Blockchain finance for social good - Supporting the United Nations Social Development Goals

We encourage you to come up with ideas that support the delivery of one or more of the United Nations Sustainable Development Goals.

Your solution must be open source and developed using the Marlowe Playground. Although a comprehensive and full codebase is not required, you should include sufficient example code to demonstrate the effectiveness of your solution.

As important as example code is the production of a slide deck, outlining the problem you have identified or the way in which you can advance the achievement of the UN's Social Development Goals and how your solution addresses it. We are determined to inspire those who currently have little or no knowledge of blockchain and the slide decks produced by winning participants will be used to illustrate to the public, governments, and other organisations how blockchain can overcome many of the barriers to sustainable development - particularly in the rapidly developing nations where the UN works.

We want to keep this challenge open ended to bring the brightest ideas to the table - many of which we are certain that we have not formulated. To spur some thoughts, below are a few examples.

Microfinance

Microcredit offered by local community groups and microfinance organizations such as the Grameen Bank have been able to reduce the presence of predatory lending and high-interest loans that are typically the only options for low-income individuals to obtain credit. A study authored by Shahidur R. Khandker in The World Bank Economic Review on the effects of microfinance in Bangladesh suggested that "access to microfinance contributes to poverty reduction, especially for female participants, and to overall poverty reduction at the village level."

Today, microloans are practiced in geographically localized communities and neighborhoods. Decentralized finance allows for a higher degree of financial inclusion which is particularly

important to individuals who don't have access to reliable or accessible banking, credit, much less even have a proof of identity. Can microfinancing be extended beyond geographic boundaries or made decentralized to serve communities in developing nations and give them access to credit?

Continuous Funding & Accounting for Risk

Several social impact organizations face a conundrum while trying to obtain financial support for their projects. Social enterprises and impact organizations require a high upfront cost to start and implement their projects. However, most financial institutions are risk-averse and thus are not able to financially support these organizations, leaving them unable to bring their cause to life. This problem has led to the emergence of outcome-based financing methods. Outcome-based financing or results-based financing (RBFs) methods utilizes reliable and real-time performance data feeds to determine if the goals of the project are being met. For the financier, this approach de-risks the investment, and for the social impact organization, they are able to obtain regular and predictable instalments of funding.

Social impact projects operate in uncertain and risk-heavy environments. It is often difficult to predict risk prior to the project's start. This warrants the need for a dynamic system that can adjust its parameters according to varying risk. Some of these solutions include adaptive impact financing mechanisms such as the alpha-bond. The algorithmic nature of smart contracts allows for several desirable properties and economic mechanisms to be encoded into these systems.

Prizes

Awards for the winning entry or entries are twofold. IOHK has sponsored the challenge to the tune of $10,000 - to be awarded to winning proposals in the Cardano blockchain's native cryptocurrency token, ada. In addition, the selected proposals will receive post-award support that will be provided by the UN and IOHK, to help projects come to fruition.

Projects must address the following criteria:

- *Social impact for the SDGs - Does the project advance solutions for the UN's Sustainable Development Goals using blockchain technology? (20 points)*
- *Technical Prowess - Does this submission address the technical challenges of its implementation? (15 points)*
- *Scalability - Would this solution scale in its scope and beyond its presented context? (20 points)*
- *Does this solution complement or expand upon IOHK technologies? (15 points)*

- *Financing for the SDGs - Does the solution address either microfinancing, continuous funding, de-risking investments, incentivize crowdfunded donations, multi-stakeholder allocation of resources, or otherwise fall under the recommendations of the UN's Digital Finance Task Force Report? (15 points)*
- Volunteering for the SDGs - Does this solution have an innovative component that supports volunteering for the SDGs? (15 points)

About Marlowe and Marlowe Playground

Marlowe is a special-purpose language to express financial smart contracts built for Cardano. The https://alpha.marlowe.iohkdev.io/#/simulationMarlowe Playground enables developers to visually plug-and-play functions and modules to construct financial instruments. It provides a means for developers to not only write smart contract code, but to also perform preliminary iterative design using simulations, and the ability to formally verify and test smart contracts. This sandbox environment allows developers to build financial contracts and instruments end-to-end.

Marlowe in a nutshell

Marlowe is a small language, with a handful of constructs that, for each contract, describe behavior involving a fixed, finite set of roles:

- *A running contract can make a payment to a role or to a public key.*
- *In a complementary way, a contract can wait for an action by one of the roles, such as a deposit of currency, or a choice from a set of options. Crucially, a contract cannot wait indefinitely: if no action has been initiated by a given time (the timeout), then the contract will continue with another behavior, such as refunding any funds in the contract.*
- *Depending on the current state of a contract, it may make a choice between two future courses of action, which are themselves contracts.*
- *When no more actions are required, the contract will close, and any remaining currency in the contract will be refunded.*

When a contract is run, the roles it involves are fulfilled by participants, which are identities on the blockchain . This model allows a role to be transferred during contract execution, so that roles in a running contract can be traded. Each role is represented by a token on the chain, and transferring this transfers the ability to perform the role's actions. Taking this further, we can represent a single role with multiple tokens, thus allowing the role to be shared: this could be termed being 'securitised' .

The Marlowe playground

We deliberately chose to make the language as simple as possible, so that it is straightforward to implement on Cardano and in the Marlowe Playground . Marlowe describes the flow of crypto assets between participants, and for this to be implemented in practice on the Cardano blockchain, code has to be executed both on-chain and off-chain: remember, though, that just one Marlowe contract describes both parts. The on-chain part accepts and validates transactions that conform to the requirements of the contract: this part is implemented as a single Plutus script for all Marlowe contracts, with the particular Marlowe contract comprising a datum passed through the transactions. Off-chain, the Marlowe contract will be presented via the user interface and wallet, offering or, indeed, automating deposits, choices and receiving payments.

In the Playground, we're able to simulate contract behavior, so that potential users can walk through different ways that contracts can evolve, depending on the different actions taken by the participants. In the main simulation, users have an omniscient point of view and are able to perform actions by any participant, with the option at each point to undo the actions taken, and then to try a different path. The wallet simulation allows users to see behavior from one particular participant's perspective, thus simulating how that user will interact with the running contract once it is deployed on the blockchain.

This simplicity also makes it possible for us to model Marlowe contracts in an SMT solver , a logic analysis tool for automatically checking the properties of systems. Using this model, which we call static analysis, for each contract we are able to check whether or not it might fail to fulfil a payment, and if the contract can fail we get evidence of how it fails, helping the author to rewrite the contract if they wish.

We can build a formal model of our implementation in a proof assistant, in which we are able to produce machine-checked proofs of how the language behaves. While the SMT solver works for individual contracts, the proof assistant can prove properties of contract templates, as well as the system itself: for instance, we can show that in any running contract, the accounts it references can never be in debit. Simulation, static analysis, and proof provide complementary levels of assurance for a contract to which users will be committing assets to ensure that the contract behaves as it should.

We have seen how Marlowe contracts can be analysed in various ways, but how do authors actually write contracts in Marlowe? The Playground provides several ways of producing Marlowe contracts. Users can write Marlowe directly, but beginners often choose to build contracts visually, using an interactive Blockly editor. Working in this visual editor has the advantage of not having to remember Marlowe constructs, since they are provided in a palette in the shape of jigsaw puzzle pieces.

Alternatively, you can develop contracts in Haskell, because the Marlowe DSL is in fact embedded in Haskell. This approach allows users to build up contracts step by step from components. The Marlowe team is also working on a JavaScript embedding of Marlowe, so that users will be able to write smart contracts using JavaScript, but still enjoy all the advantages of analysis, simulation, and proof, as provided by the Marlowe implementation.

Required tool or tech stack:
Marlowe: https://wyohack.marlowe.iohkdev.io/#/simulation

Judging criteria:
Projects must address the following criteria:

- *Social impact for the SDGs - Does the project advance solutions for the UN's Sustainable Development Goals using blockchain technology? (20 points)*

- *Scalability - Would this solution scale in its scope and beyond its presented context? (20 points)*
- *Does this solution complement or expand upon IOHK technologies? (20 points)*
- *Technical Prowess - Does this submission address the technical challenges of its implementation? (10 points)*Volunteering for the SDGs - Does this solution have an innovative component that supports volunteering for the SDGs? (10 points)
- *Does this solution complement or expand upon IOHK technologies? (20 points)*
- *Financing for the SDGs - Does the solution address either microfinancing, continuous funding, de-risking investments, incentivize crowdfunded donations, multi-stakeholder allocation of resources, or otherwise fall under the recommendations of the UN's Digital Finance Task Force Report? (10 points)*
- *Does this solution address one of the priorities in the UN Secretary General's Data Strategy? (10 points)*

Names of appointed judges:
Charles Hoskinson, Romain Pellerin, (IOHK) Marc Liberati, Tristan Roberts (UN personnel)

Prize description

1. Up to $10,000 - remitted in the Cardano blockchain's native cryptocurrency, ada.
2. Ongoing support, from UN and IOHK personnel, to support development of selected projects

Prize terms and conditions
(As per challenge description)

Rights required?
No - projects must be open source.