Term 2: Bi-Weekly Report 1

Atos Blockchain Team 3

George Pîrlea, Alexis Enston, Danish Alvi

January 16, 2016 – January 27, 2016

1 Overview

Since the last report, we have partially implemented a web-based user interface for the app, recorded a team video, developed the request/provide/get record parts of the API, implemented smart contracts for identity and some record types (reference and membership), developed a main BitKariero smart contract and improved the build process.

(This report also includes some of the tasks we've completed over the winter break.)

2 Meetings

Jan 17: Lab meeting

Attendees: George Pîrlea, Danish Alvi, Alexis Enston

- Allocated tasks:
  - George: API for deploying contracts, request record, provide record
  - Danish: identity contract, reference and membership (revokable) contract
  - Alexis: list all records for an address

Jan 27: Lab meeting

Attendees: George Pîrlea, Danish Alvi, Alexis Enston

- Reviewed completed work and planned future work
- Allocated marks

3 Completed tasks

- Partially implemented user interface (identity, records, requests)
- Developed initial version of the API: request/provide/get records

- Implemented main BitKariero smart contract: events
- Integrated UI and backend build systems
- Created smart contract for identity
- Implemented reference and membership record types (smart contracts)

4 Plan for the next two weeks

- Extend API: identity management, record verification, identity verification
- Integrate API into the user interface
- Write unit tests for smart contracts
- Start experimenting with IPFS

5 Individual Reflection

George: I've done a large chunk of the work for the UI, wrote and delivered the elevator pitch, and implemented an initial version of the API. EmbarkJS, much like the wider Ethereum ecosystem, is quite immature and very poorly documented. A lot of the effort I've put in has revolved around discovering how things work and how to get around bugs in EmbarkJS. I've also helped Alexis with improving our (initially brittle) build system.

Alexis: Since the last bi-weekly report, I have worked on the team video, and done further development work. I focused on solving the requirement of obtaining a list of reference requests sent by the user. Ethereum provides no log of transactions sent by the user, so in order to find them you would need to parse every transaction in every block of the blockchain, which is very inefficient and will take a few hours to complete. To solve this issue, I implemented a main BitKariero contract which keeps a log of all requests sent, and users can then scan this log quickly and efficiently. We may also use this main contract for other features, such as storing the list of authentication authorities trusted by the system. I have also started working on integrating the BitKariero API into the React UI, and have got our two build systems, embark and gulp, working together successfully with George.

Danish: Over the course of the previous week, I had been constructing a feasible and contract-oriented UML model for the implementation of the three different types of requests.

This week, I have primarily worked on implementing the identity and reference in Solidity for the requests based on the UML model I had constructed. Other than writing the main contract code in solidity, I also tested these methods by making a CLI interface for deploying the contracts using Web3 JavaScript API, and successfully created a NodeJS implementation for deploying these contracts.