Ethereum Smart Contracts (Requirement Analysis)

Introduction about Ethereum and Smart Contracts

Smart contract is just a phrase used to describe computer code that can facilitate the exchange of money, content, property, shares, or anything of value. When running on the blockchain a smart contract becomes like a self-operating computer program that automatically executes when specific conditions are met. Because smart contracts run on the blockchain, they run exactly as programmed without any possibility of censorship, downtime, fraud or third party interference.

Ethereum is a decentralized platform that runs smart contracts: applications that run exactly as programmed without any possibility of downtime, censorship, fraud or third-party interference.

The problem the system will solve

The current problem with payments is that the claims process can take weeks or even months to be paid. The process is still very manual and requires a large degree of human action. This adds up to a lot of administrative costs, which result in higher premiums for customers. Any company can automate payments by coding them into a smart contract. Not only does the smart contract reduce the administrative costs, but transparency and trust in the process is visible to all stakeholders and all regulatory bodies – thanks to the distributed nature of the smart contracts on the blockchain.

By using smart contracts, there will be a lot of blockers removed and problems solved. Mentionable benefits would be eliminating the risk of manipulation or fraud, allowing faster working processes, reducing the possibility of human error, discarding the reliance on third-parties, decreasing the cost of service, enabling transparency of the transaction to all concerned parties.

Clients of the system (and their needs, vision, requirements for the system)

Every person owning a property that is taxable through any means by the government, state or any state-related public institution should take advantage of the system we envision. One should stop relying on the physical payment of the taxes he's enlisted on or on the timely bank transactions for transfering the money from an account to another or to a public institution.

Client's vision over the system is an automated, trustful, fast and cheap system which will automatically withdraw an amount of currency from their e-wallet and accomplish covering all the tax payments he needs. The client should not have to deal with papers, signatures, bills or even other persons. The client should spend the least amount of energy to have all of his taxes paid in the least amount of time with the most trusted process available. The client should not be afraid of being a victim of a fraudulent corrupt system and leverage the system's processes to escape these situations.

System

Our envisioned system relies on the blockchain technology which uses Ethereum cryptocurrency to create and successfully run smart contracts. These smart contracts should be genuine enough to transfer the needed amount of cryptocurrency after meeting all the conditions enforced by it. After succeeding the smart contract, the final state of the world should have a payer which had its amount withdrawn and the receiver which obtained the amount transferred with the monthly or yearly term met.

Input & output details

The receiver (e.g. public institution) needs to define the conditions of the smart contract based on the officials regulations of the tax while the client should specify the amount of cryptocurrency and the address of the smart contract payment received.

The receiver will output a payment receipt that serves as a confirmation for the client that the transaction has succeeded and the tax is paid for another time range (month, year).

Expectations (functional, non-functional)

The system is expected to (functional):

- Allow users to introduce a desire amount of Ethereum
- Process the amount of Ethereum and the transaction
- Validate/invalidate the conditions of the smart contract
- Accomplish the state-related tax payment
- Allow institutions to deploy self-created smart contracts

The system's non-functional expectations are:

- To be a secure environment for the transactions
- To ensure fast processed transactions between entities
- To comply the state's ethics, laws or constitution

Behavior & characteristics (a.k.a possible actions, use-case scenarios, constraints)

Our smart contracts Ethereum-based system will allow individuals enroll into an automated workflow which will help them spend less time on tax payments and help the focus on other daily activities.

The system will be able to:

- Render a simple, intuitive UI that allows the input of the Ethereum amount
- Create a blockchain transaction for the specified amount and received
- Successfully propagate the transaction into the Ethereum decentralized network
- Validate/invalidate the conditions of the contract
- Allow the receiver to acquire the amount of Ethereum specified by the payer

Another interesting characteristic would be to implement an automated trigger based on specified time-dates that will run the transaction itself so that the user should never worry about making it himself.

Example: Bob has a car that has been recently registered and he has to pay the yearly tax for it. Instead of walking to a public institution and hand the money himself or set up a bank account to transfer the money yearly, the system can be configured to be triggered every 1st January every year and run a transaction for him. That way, he will comply with all the official regulations enforced by the cryptographic mechanisms of blockchain.

The system should constrain the transaction if a series of conditions is not met. Example:

- The payer's e-wallet does not contain the right amount of cryptocurrency
- The smart contract's address is missing or invalid
- The receiver's e-wallet is missing or invalid

Major components

Smart Contract

- *TaxPayment* Contract<payer, receiver, amount> (along with other functionalities)

Web Application

- User Interface
- Mechanism for UI - Transaction interaction
- Mechanism for propagating Transaction to Ethereum network

Technologies used

- **Solidity**

Solidity is a statically-typed programming language designed for developing smart contracts that run on the EVM. Solidity is compiled to bytecode that is executable on the EVM(Ethereum Virtual Machine). With Solidity, developers are able to write applications that implement self-enforcing business logic embodied in smart contracts, leaving a non-repudiable and authoritative record of transactions. Writing smart contracts in smart contract specific languages such as Solidity is referred to as easy (ostensibly for those who already have programming skills).

- **Truffle Ganache**

Truffle is a world class development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine (EVM), aiming to make life as a developer easier. With Truffle, you get, through others, built-in smart contract compilation, linking, deployment and binary management, automated contract testing for rapid development, criptable, extensible deployment and migrations framework.
Ganache is a personal blockchain for Ethereum development you can use to deploy contracts, develop your applications, and run tests.

- **Pyethereum -** the Python core library of the Ethereum project.

Analogies & examples

Imagine you're a farmer who wants to buy insurance, so you'll be reimbursed if there's a drought. Per your contract, if the temperature is more than 38 degrees for more than 100 days, you should get an insurance payout of $10,000.

Today, you'd probably have to collect documentation about the weather and submit it to your insurance company. And then, you'd wait. Maybe you get reimbursed, maybe you don't. You might find out that you have to submit additional documentation. Or that you waited too long to file your claim.

With a smart contract, at the end of summer, the insurance company would execute your policy. They'd check public weather records for your location, and if they found that the weather had been over 100 degrees for more than 100 days, they'd automatically transfer the money to you. This eliminates counterparty risk and provides huge efficiencies for the insurance company, while also providing a better experience for you, the farmer.