Babbage – a Mechanical Smart Contract Language

Christian Reitwießner
`chris@ethereum.org`

Abstract

Smart contract programming languages should be easy to understand and unambiguous. Usually, such languages are written in formal computer languages comprised of functions and variables. While this already makes them quite abstract and hard to understand, the fact that components of a smart contract can be referenced by name partly from anywhere in the program sometimes makes it almost impossible to see how different parts interact and fit together.

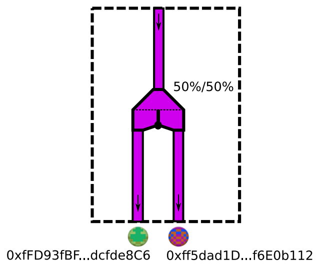
We propose a language that is situated at a level even untrained people can understand. Babbage is a visual programming language that consists of simple mechanical parts that interact with each other: Rods, levers, cords, springs and simple pneumatic hoses. Since components that want to interact with each other have to be physically close, the modularity of such systems is already guaranteed by design. Furthermore, people with no knowledge of programming languages have the chance to understand complex smart contracts.

A smart contract on Ethereum can be described as a vending machine made of glass, put at a public place: Once it is created, it is not possible to modify it, except by pushing the buttons that are mounted at its exterior. Furthermore, everyone has the possibility to watch how it works internally. Note that open source software installed on cloud computers is similar, but there is still an important difference: You can take a look at the published source code, but you have no way to tell whether the server actually runs the same code. This is similar to a vending machine not made of glass: You can look at the blueprints, but you never know what the machine actually does.

If the machine is made of glass, it is (depending on its complexity) quite easy to see what will happen if you put a coin inside and then push a button. You can even see if the drink you want to buy is still available.

We want to take this analogy further and build actual machines inside Ethereum. The benefits are that humans should be able to figure out how a mechanical machine works without any prior knowledge. Furthermore, it should be quite easy to see any possible way in which a switch (a variable in text-based programming languages) can be modified, because there has to be a physical connection to that switch. Finally, as Ether is modeled as a liquid flowing in pipes, you can directly see where it will go.

Also note that (software) engineers often resort to diagrams when they want to explain something. Admittedly, diagrams often simplify things and thus, a mechanical smart contract language might not be as expressive as a text-based language, but on the other hand, mechanisms that are so complex that they cannot be fully explained using a diagram should perhaps not be used for smart contracts anyway.



Let's start with an example. A smart contract that distributes all Ether sent to it equally among two addresses: