Given the structure of Ethereum, it is not possible to schedule calls to contracts at a point in the future. This is because contracts can not subscribe to events and therefore all the actions need to be triggered by an entity external to the contract. In many applications, one may be interested in scheduling an operation (transaction, execution of a particular function) in the future. One specific example is the blockchain-based lotteries and other gambling decentralized applications (Dapps) that base the generation of random numbers on network variables. For security reasons, lotteries need to use parameters of the network taken at some time point in the future. Also, some applications require delaying payments (ether transactions) automatically, a seller may decide to wait for a specific number of block-confirmations before doing the final step towards completing a business transaction, etc.

Chronos is a smart contract that does just that. With Chronos, any contract can register a request to be called at a particular block. The smart contract has been designed to be efficient, reducing gas consumption related to the schedule of the execution of calls. Furthermore, the code is fully available on Github and can be used by anyone. The main features of Chronos are:

- Simple integration with your smart contract.
- Allows for withdrawal of the gas-cost not consumed during the call.
- Single or recurrent call to the contract.
- Available on the Test Network (Rinkeby).

It is straightforward to integrate any smart contract with Chronos. Adding two functions into a contract's code makes it capable of scheduling and receive calls. The code of the smart contract and examples of how to integrate any contract with Chronos can be obtained here. Chronos is currently available on the test-net (Rinkeby) at this address: 0x4896FE22970B06b778592F9d56F7003799E7400f

Chronos is able to solve the problem described but becomes centralized as currently, the execution depends on one administrator. The following step is to allow anyone on the network to be part of the applications such that the system became less centralized and therefore more reliable. This implies to develop a distributed mechanisms for the front end application.