Theory of Blockchain

Session 9:

Ethereum – Part 1

Module 1 – Ethereum Core Idea

Bitcoin is not Alone

- There are plenty of other cryptocurrencies
  - Monero
  - Litecoin
  - Zcash
  - Ripple
  - IOTA (tangles)
  - …

- But Ethereum opened a new chapter in blockchains in 2013.

Ethereum

Ethereum was proposed by Vitalik Buterin in 2013~2014. But it went live in 2015.

The idea of smart contracts was first conceived in 1994 by Nick Szabo. But it was not really realized before Ethereum. Buterin wanted to generalize the idea of blockchains, and mixed it with programming to create smart contracts.

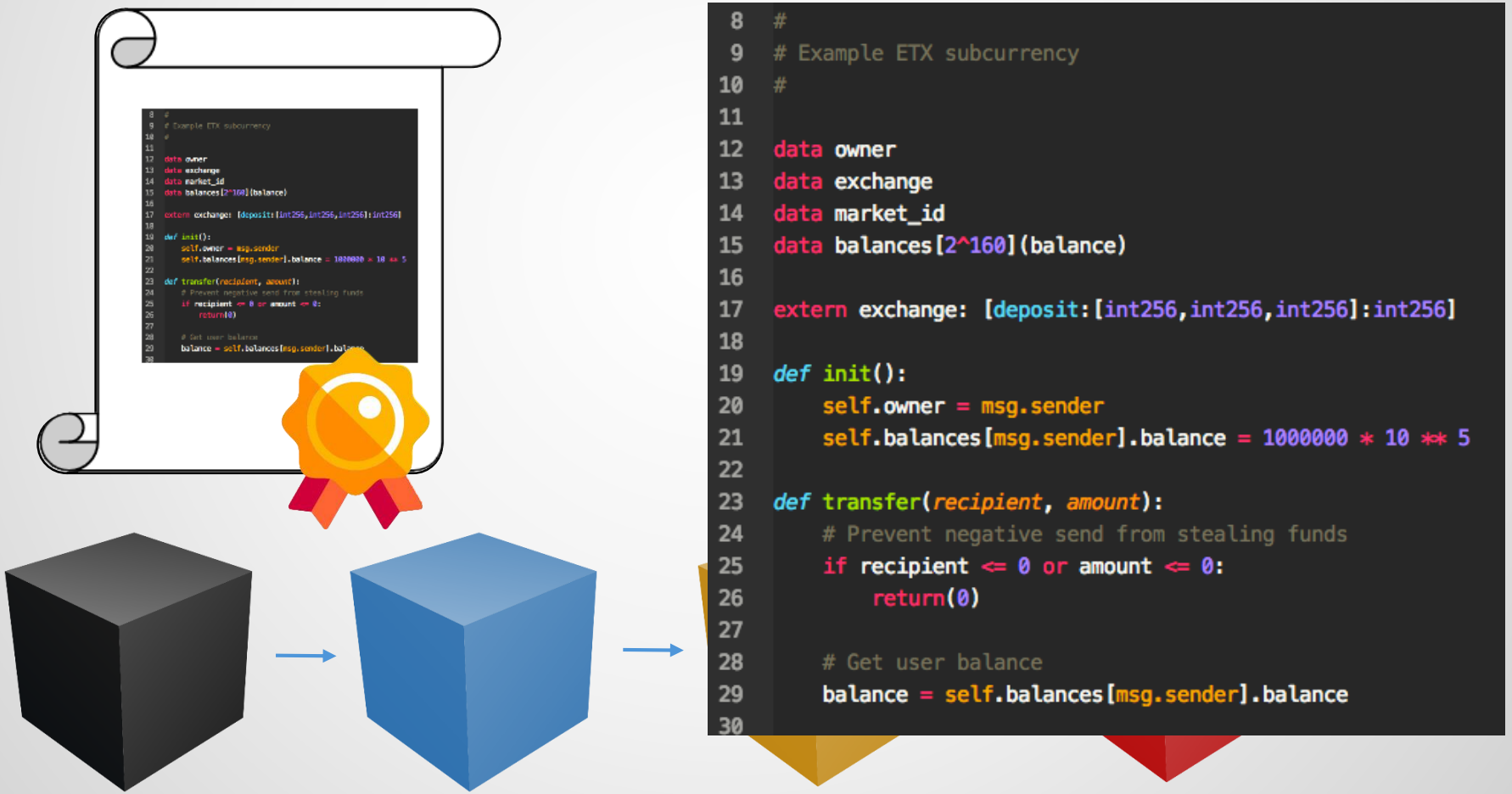
Ethereum's Whitepaper

Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform.
By Vitalik Buterin (2014).

When Satoshi Nakamoto first set the Bitcoin blockchain into motion in January 2009, he was simultaneously introducing two radical and untested concepts. The first is the "bitcoin", a decentralized peer-to-peer online currency that maintains a value without any backing, intrinsic value or central issuer. So far, the "bitcoin" as a currency unit has taken up the bulk of the public attention, both in terms of the political aspects of a currency without a central bank and its extreme upward and downward volatility in price. However, there is also another, equally important, part to Satoshi's grand experiment: the concept of a proof of work-based blockchain to allow for public agreement on the order of transactions. Bitcoin as an application can be described as a first-to-file system: if one entity has 50 BTC, and simultaneously sends the same 50 BTC to A and to B, only the transaction that gets confirmed first will process. There is no intrinsic way of determining from two transactions which came earlier, and for decades this stymied the development of decentralized

What is a Smart Contract? (animated)

- It’s a piece of code or computer program, stored in blockchain.

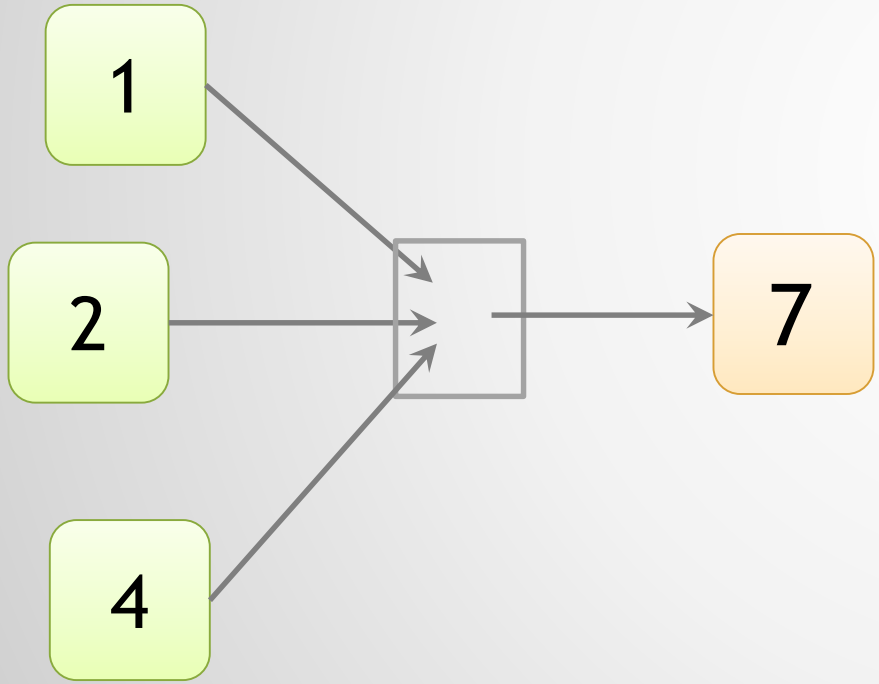


```
#
# Example ETX subcurrency
#

data owner
data exchange
data market_id
data balances[2^160](balance)

extern exchange: [deposit:[int256,int256,int256]:int256]

def init():
    self.owner = msg.sender
    self.balances[msg.sender].balance = 1000000 * 10 ** 5

def transfer(recipient, amount):
    # Prevent negative send from stealing funds
    if recipient <= 0 or amount <= 0:
        return(0)

    # Get user balance
    balance = self.balances[msg.sender].balance
```

A Different yet Similar Management

Bitcoin

(transaction-based)



Ethereum

(account-based)

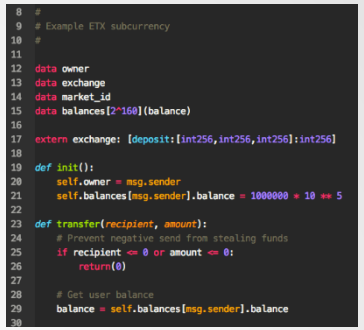
Address: 0x39ae1..
Balance: 5.2 ETH
Contract/Code:
“….
b=b+i;
….
“

It’s more efficient to update a balance rather than keeping the track of unspent transactions. It’s easier to look up during programming too. However, anybody who wants to verify the correctness of states, must revert to the transactions.

Ethereum Account Types

Externally (user)-owned Accounts

- Owned by an active external body (e.g. person or company)
- Is able to submit transactions (to trigger running of a contract or to transfer money)
- Mainly has: Address + Balance

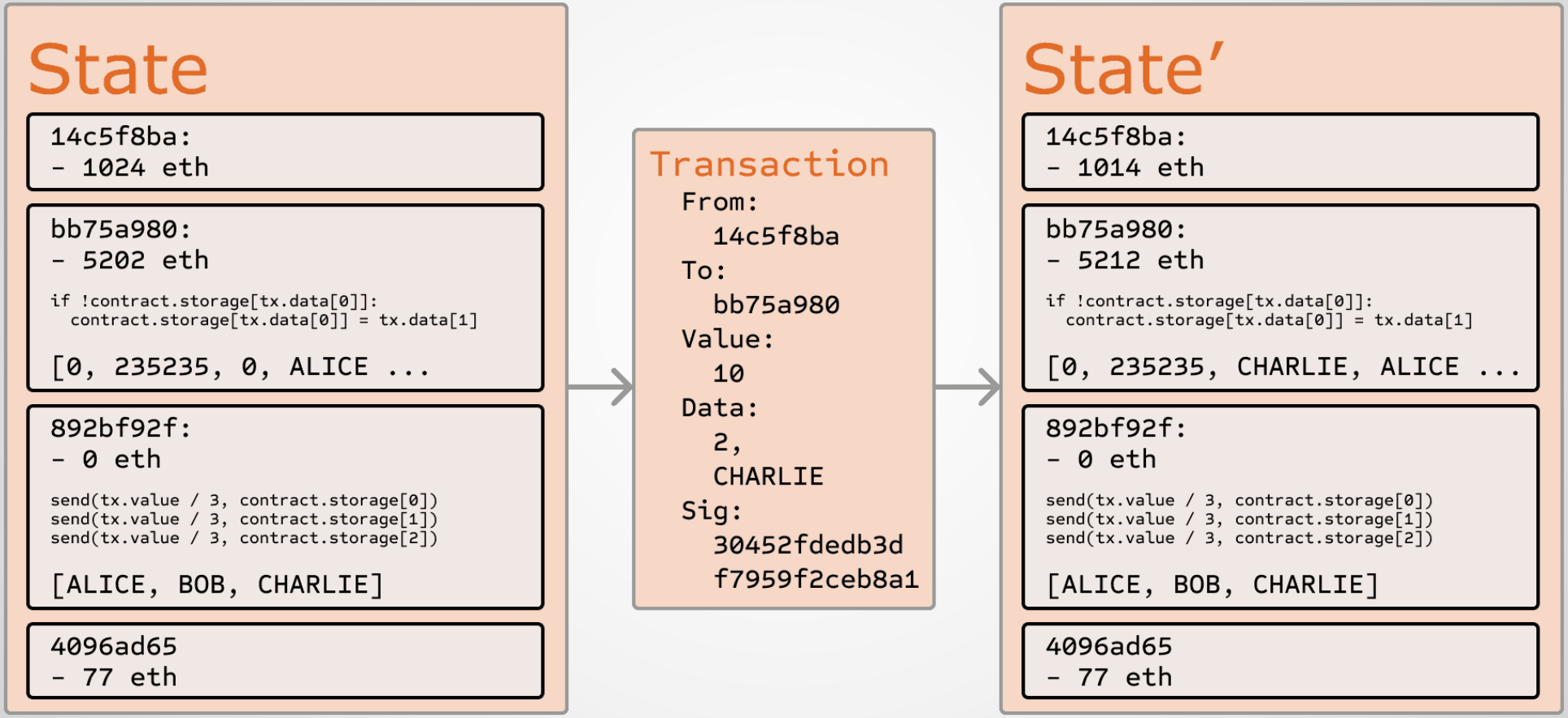


```
#
# Example ETX subcurrency
#

data owner
data exchange
data market_id
data balances[2^160](balance)

extern exchange: [deposit:[int256,int256,int256]:int256]

def init():
    self.owner = msg.sender
    self.balances[msg.sender].balance = 1000000 * 10 ** 5

def transfer(recipient, amount):
    # Prevent negative send from stealing funds
    if recipient <= 0 or amount <= 0:
        return(0)

    # Get user balance
    balance = self.balances[msg.sender].balance
```

Smart Contract Accounts

- Owned and managed by a contract.
- Maintains a state
- Execution is triggered by transactions
- Mainly has: Address + Contract + Storage (can contain Balance)

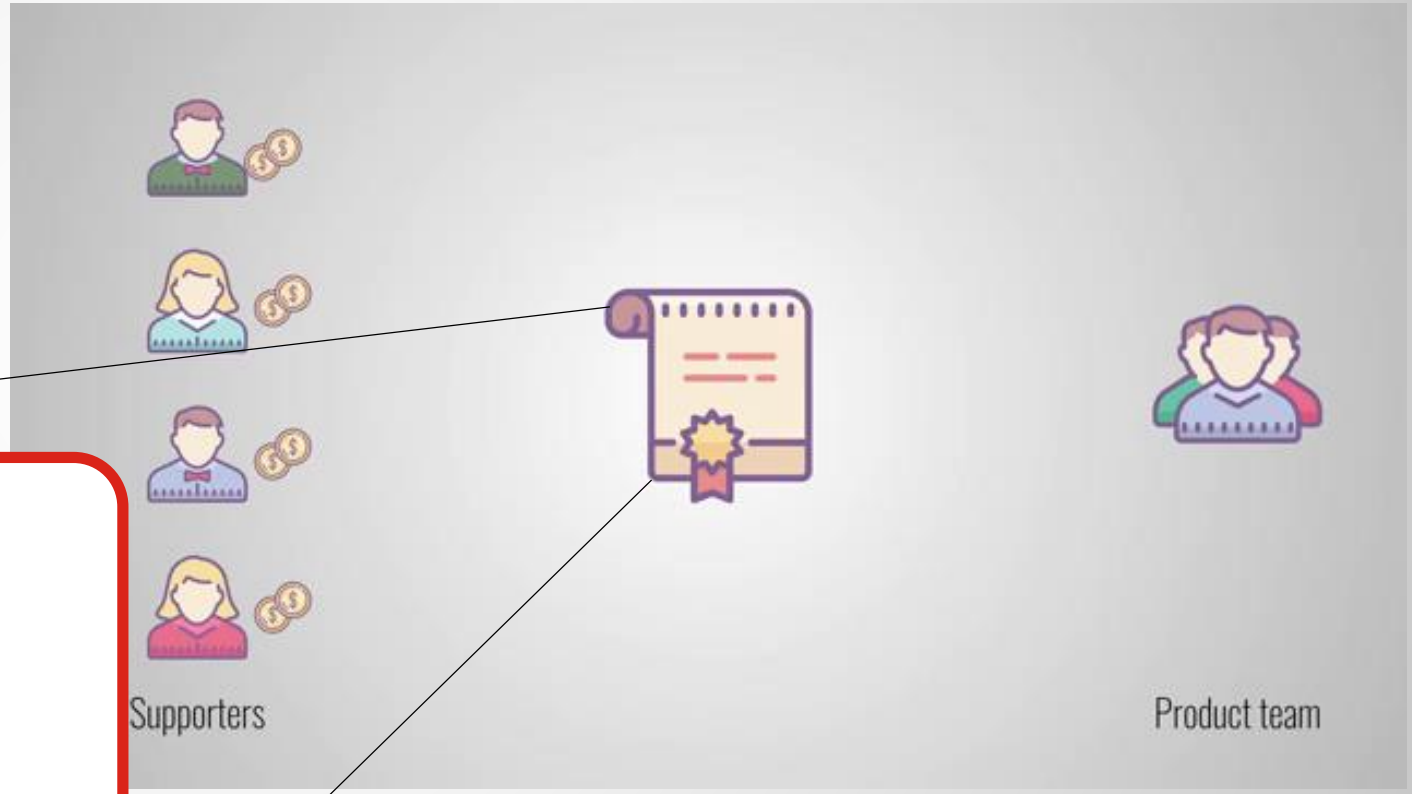
State of the Network is Updated by Transactions



CrowdSourcing with Smart Contracts

In Ethereum, you can interact with smart contracts as well as human beings (2 account types).

You can write a smart contract (program) that collects money for a project. Programs can have if/then, loops, etc.

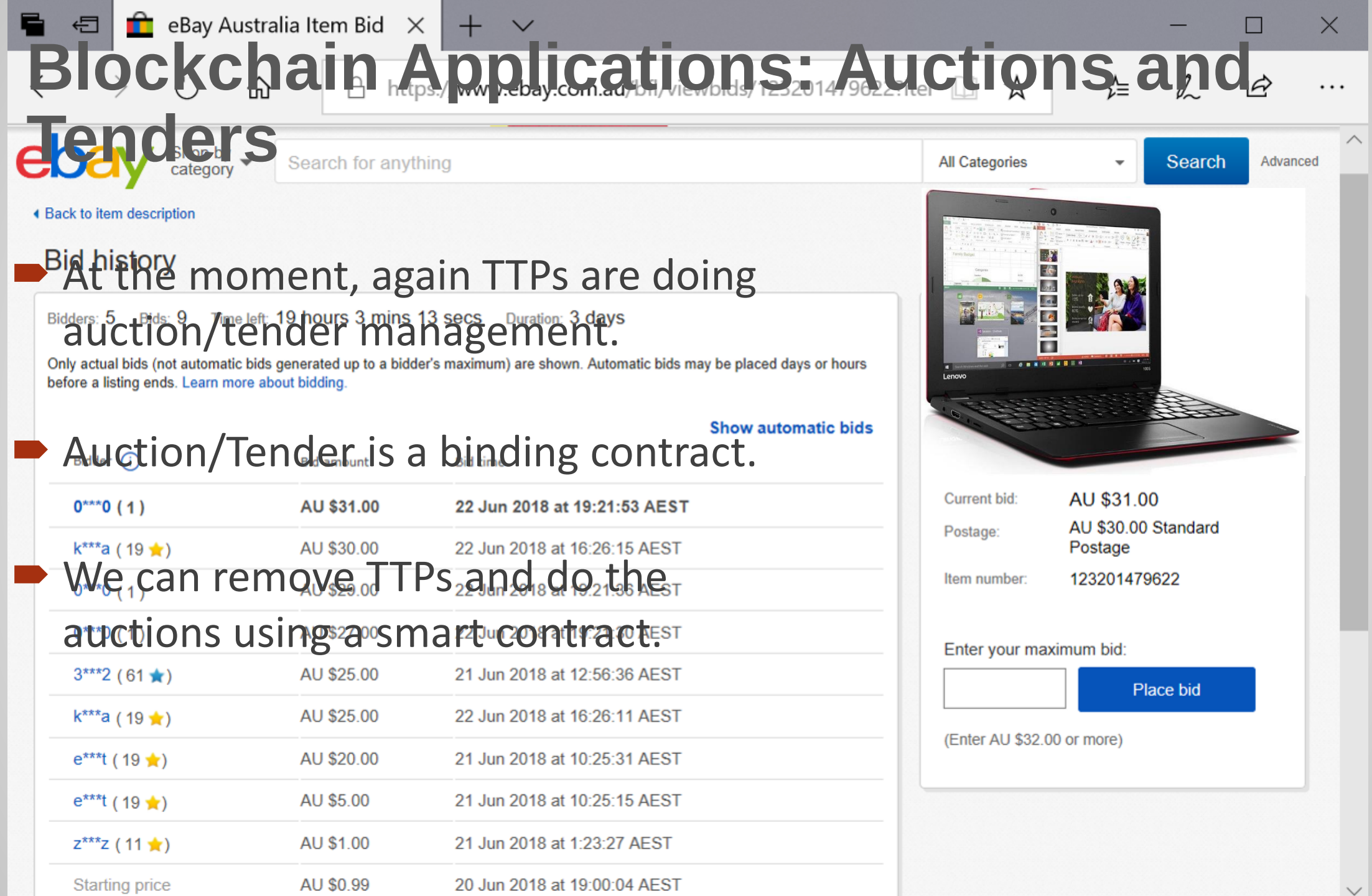


(Source: Smart Contracts by Savjee)

If the collected money ≥ T , then
{
Transfer money to the team
}
else
{
refund the money
}

Blockchain Applications: Auctions and Tenders

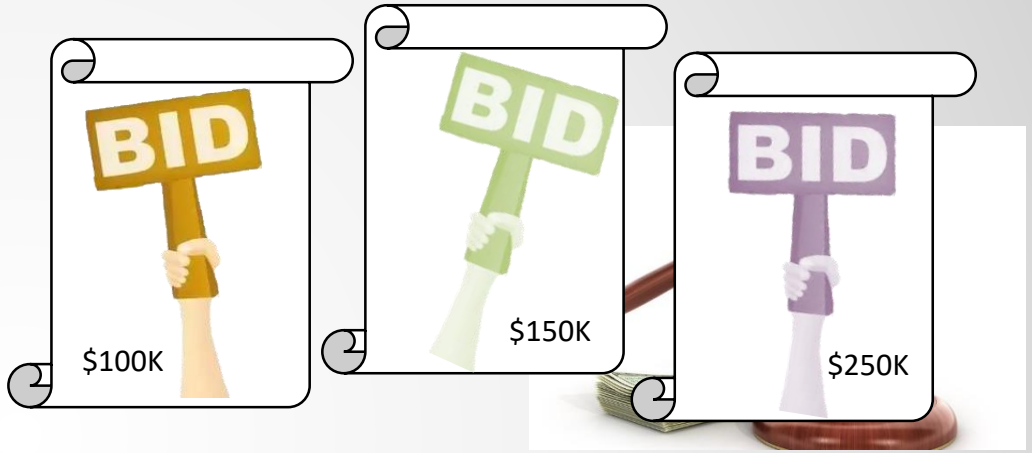
- At the moment, again TTPs are doing auction/tender management.
- Auction/Tender is a binding contract.
- We can remove TTPs and do the auctions using a smart contract.

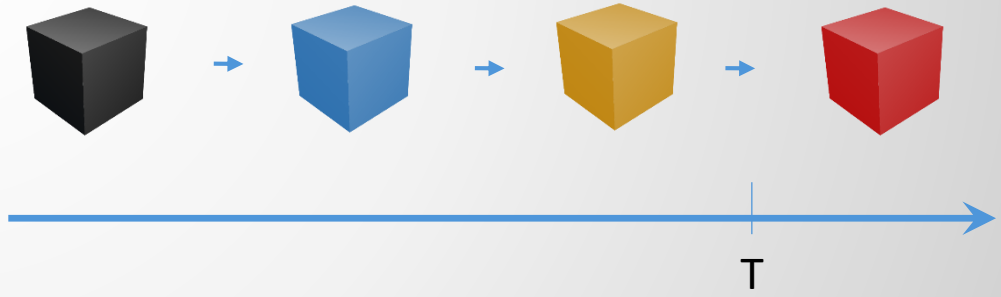


Blockchain Applications: Auctions & Tenders

- Transactions to the contract (offers/bids) can be verified by everybody and everything is transparent.

```
If  (offer > Greatest_offer & t <T)  then
   {
        Increase to the highest bid
        Switch to the new buyer
   }
    else
  {
        keep the current bid and buyer
  }
```

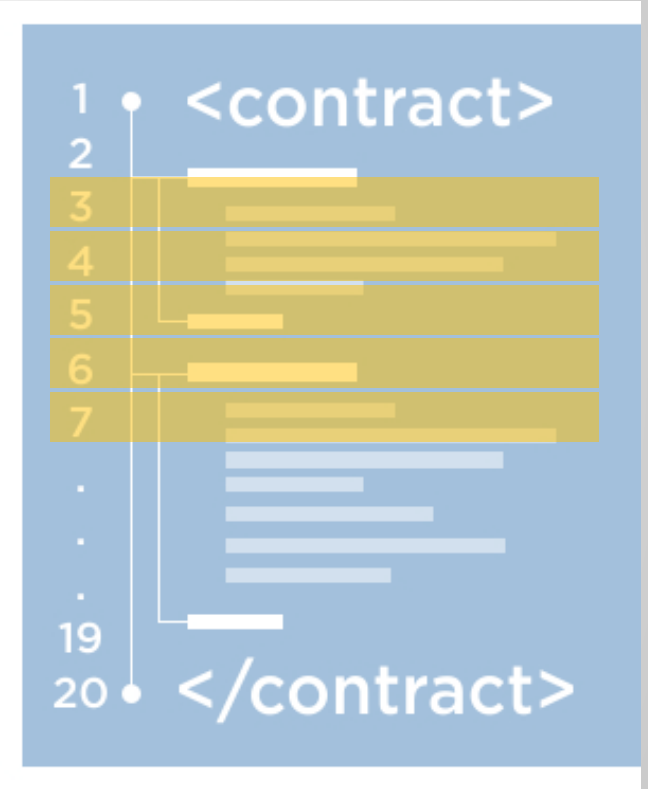




How is the Consensus Reached on a Contract?

- Similar to Bitcoin, everybody receives a copy of the smart contract (program) as well as all the other interactions done with it.

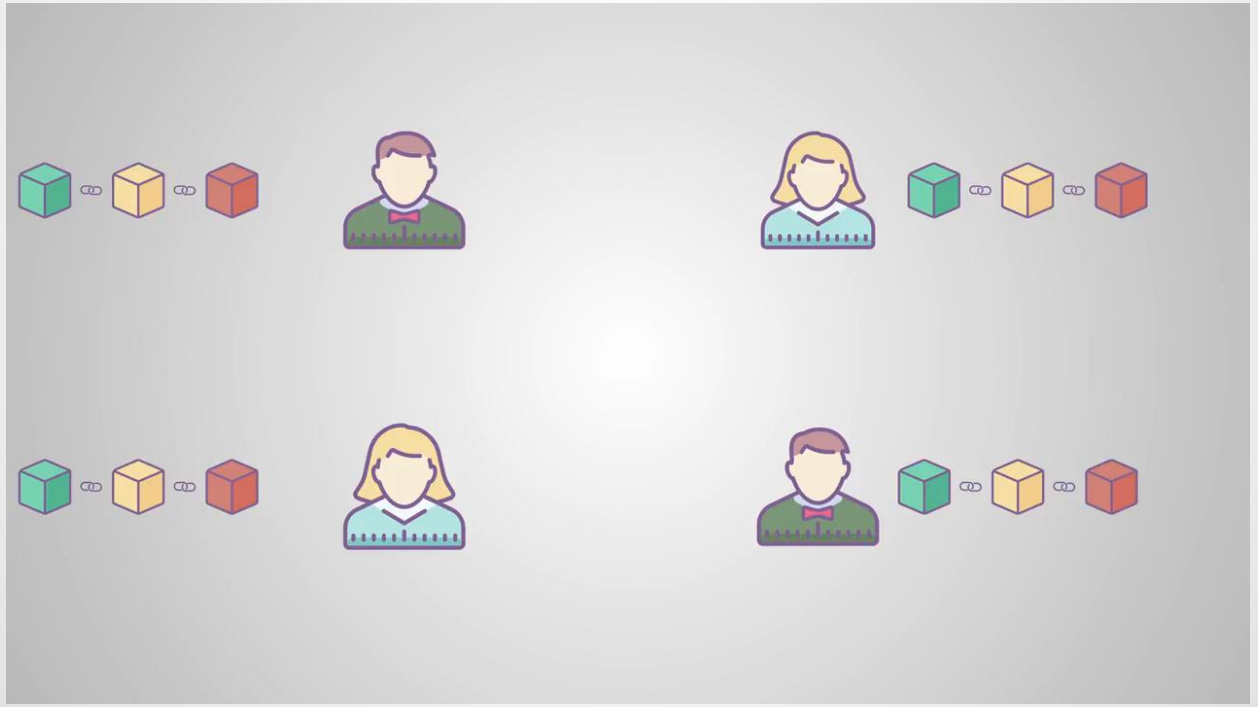
- Everybody can run the code on their computer and give it the same interactions to find the current state of the contract.



(bitsonblocks.net)

How is the Consensus Reached on a Contract?

- Again, we can use the blockchain to ease the processing and make sure the interactions are in the correct order.



(Source: Smart Contracts by Savjee)

What Comes Next …

- We learned the core idea behind Ethereum, i.e. smart contracts.
- We saw how complex programs can potentially be implemented on this decentralized platform under zero trust assumption.
- We next learn about Ethereum Virtual Machine and the concept of gas.

See you in the next module …