40.317 Lecture 11

Dr. Jon Freeman

23 June 2020

Slido Event #G973

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Agenda

- Services as “Little Languages”
- Blockchains, part 1 of 2

Services as “Little Languages”

Let’s reconsider the asset manager application we have been using as our working example.

What, essentially, is its “design”?

“Little Languages”

We can think of the interface to our server as a little language – more precisely, a domain-specific language – which we have invented to solve this particular problem.

- *Not all languages are general-purpose.*

For a service that solves a high-level problem, to design such a language *is* to design the service!

- *System* design reduces to *language* design.

“Little Languages,” continued

One of the enduring fascinations of Computer Science is that you can use one language to implement another!

- c.f. “Turing Completeness,” out of scope

The new language can even introduce an entirely new paradigm, such as OO implemented via imperative!

In-Class Discussion

Is it OK to have logic in the GUI which depends on receiving and detecting a specific error message?

If this seems a little clumsy, what could we do instead?

In-Class Discussion, continued

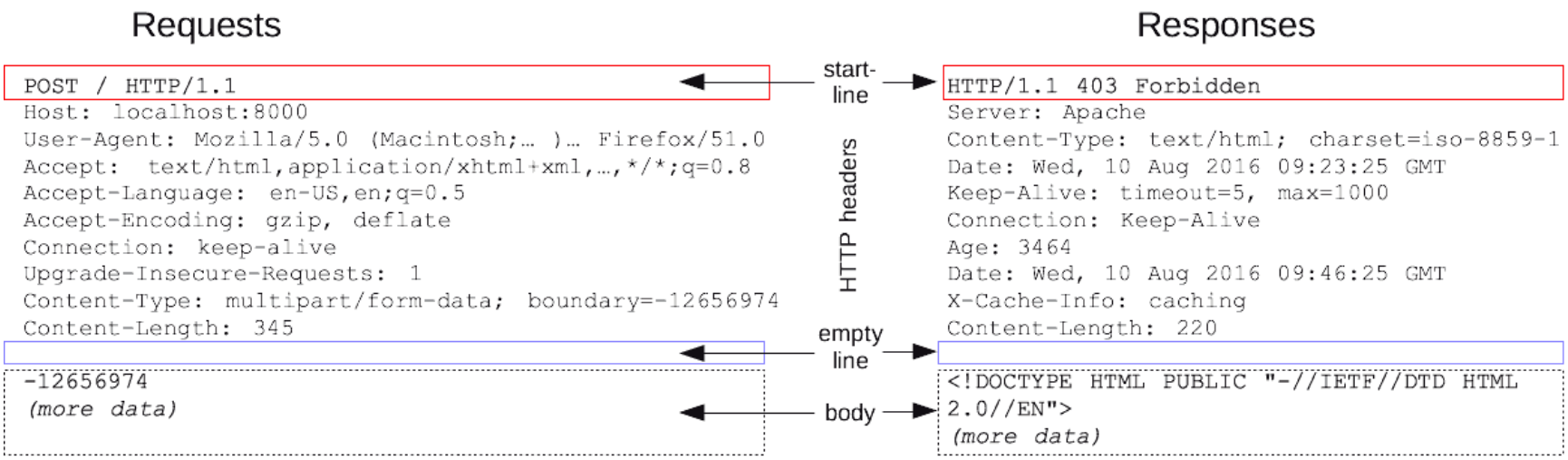
1. Break into groups.

2. Appoint a group spokesperson.

3. Prepare a group answer to each of the questions on the in-class handout.

In-Class Discussion, continued

Regarding the last question, recall the details of a browser's little language:



Source: https://developer.mozilla.org/en-US/docs/Web/HTTP/Messages

Design vs. Architecture

Per Martin Fowler, a software system's architecture is what *every* team member needs to know about it.

Agile methodologies and Gall's Law both seem to diminish architecture's importance.

At the end of Lecture 12, we will present an approach in which architecture and incremental development can co-exist.

Blockchain

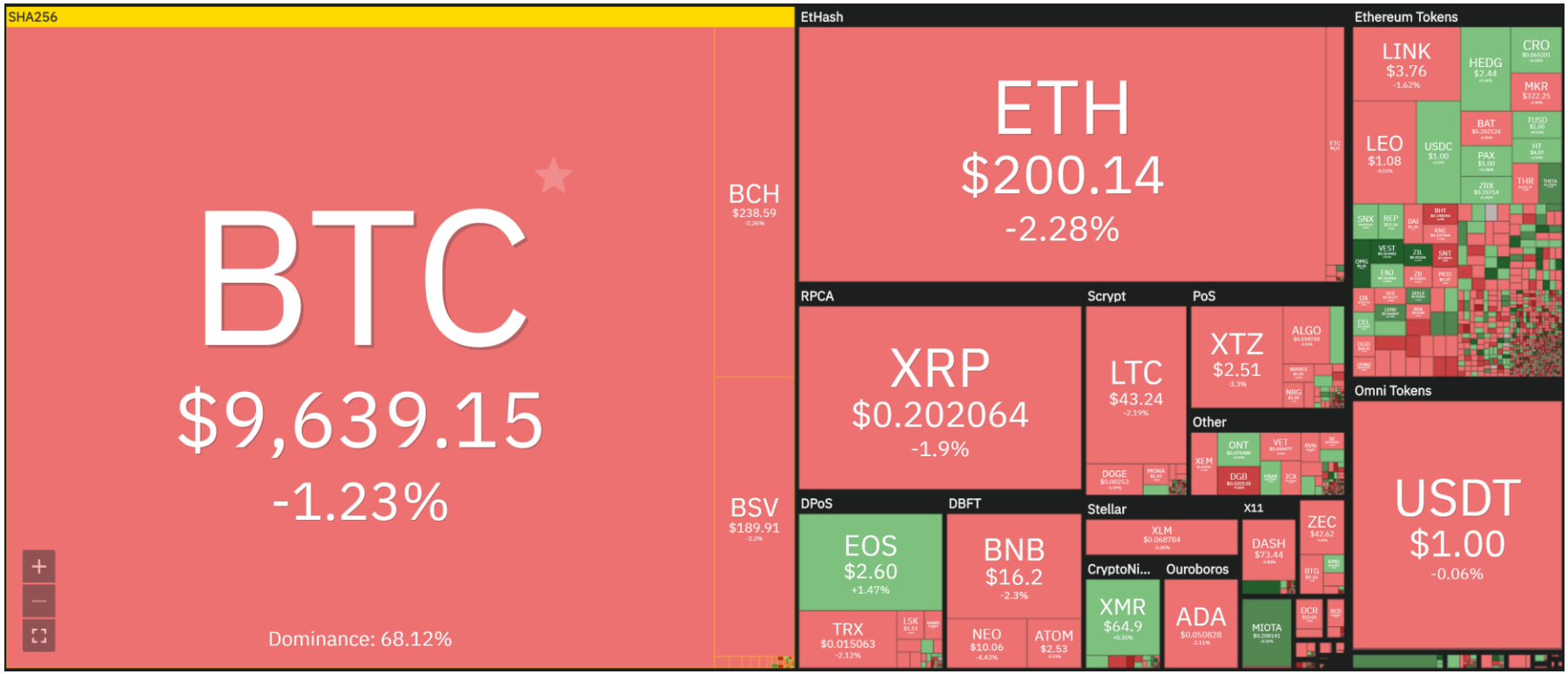
Blockchain is a distributed persistence mechanism.

Bitcoin uses blockchain, together with a competitive “mining” algorithm, to achieve *trustless, distributed consensus*.

It solves the double-spending problem using a peer-to-peer network alone, i.e. without a central authority.

Cryptocurrencies Are Not Our Focus

For one thing, they are evolving too quickly:



Source: https://www.coin360.com

A BETTER WORLD BY DESIGN.

Cryptocurrencies Are Not Our Focus

Having said that, Bitcoin is easy to explore because it is so transparent; for instance:

- https://bitinfocharts.com/bitcoin/
- https://blockchain.info/
- https://bitbonkers.com/ ☺

Definition of a Blockchain

1) A term that encompasses a broad range of distributed ledgers.

2) An append-only data structure containing data records which are cryptographically linked together.
   - Each copy contains every transaction ever executed.
   - Records are added to it when multiple parties reach a consensus based on pre-agreed rules.

What a Blockchain Is Not

A blockchain does *not* have to be:

- deeply distributed,
- public, or
- associated with a cryptocurrency.

How Blockchain Typically Works

The first block is called the genesis block.

Each block contains a hash of its parent (i.e. previous) block in the chain. Thus:

- A block can only have one parent.
- A block can only be created *after* its parent was created.
- Each block's hash is a function of the contents of the *entire* chain up to, and including, that block.

How a Blockchain Works, continued

A blockchain is valid if and only if

- it starts with the genesis block, and
- all blocks and transactions in it are valid.

A blockchain can fork. One-block forks occur when two blocks are created at roughly the same time.

Forked changes cannot be merged; when they occur, all but one must be discarded.

How a Blockchain Works, continued

Honest generators only build onto the *latest* block in the *longest* valid chain.

Blocks in shorter chains (or invalid chains) are not used for anything.

When clients switch to another, longer chain, all valid transactions of the blocks inside the shorter chain are re-added to the pool of queued transactions and will be included in another block.

How a Blockchain Works, continued

The decentralisation comes from the existence of a network of nodes which collectively store the blockchain and validate new transactions.

Each node updates its own chain when it becomes aware of a better (longer) version.

How a Blockchain Works, continued

The number of transactions in a block is a function of:

- the sizes of the serialised transactions;
- the block's capacity, a system-wide setting.

Statistically,
Average Transactions Per Block =
Block Size / Avg Transaction Size.

How Bitcoin-like Currencies Work

There are two types of nodes: relay nodes and mining nodes. Anyone can create a node of either type.

Together, the nodes establish an agreed history of transactions. Each transaction is simply the transfer of an asset from one address to another.

How Bitcoin-like Currencies Work

The owners of the mining nodes (the "miners") create new blocks and fill them with transactions (from a queue).

They earn a reward for this (for Bitcoin, 6.25 BTC since 11 May 2020) because they must earn the right to create a new block by solving a computationally difficult problem.

- Hard to solve, but easy for others to verify.
- These rewards add BTC into the system.

How Bitcoin-like Currencies Work

Each transaction must be confirmed. This involves:

- placing it in a (new or existing) block, *and*
- getting most other nodes to accept it.

BTC's mining process involves luck and chance; on *average* a new block appears every 10 minutes, but variance is high.

  - Mining difficulty is re-adjusted once every 2 weeks to *ensure* a 10 minute avg wait.

Interacting with a Blockchain

All you have is

- A public key a.k.a. address, and
- a private key

These two keys form your “wallet.”

You use your *private* key to digitally sign your transactions.

Anyone can use your *public* key to confirm your transactions really came from you.

Public vs. Permissioned Blockchains

Bitcoin's blockchain is fully decentralised, hence public. But a blockchain could also:

- require permission to read its data;
- limit the parties who can transact on it;
- appoint special participants who validate transactions.

Permissioned blockchains need not have mining / proof of work.

Ripple is an example of such a blockchain.

Primary Benefits of Blockchains

- Transparency
  - Of algorithms, code, and transactions
- Immutability
  - Hacks are possible, but very expensive
- Redundancy
  - Every node has the complete history
- Security
  - No single server or data centre to attack

Primary Benefits, in More Detail

- Establish digital identity
- Serve as a system of record
  - Thereby establish *provenance*
- Prove immutability
- Serve as a platform
  - Cryptocurrencies were first, of course
  - Next: smart contracts
  - Vision: a Web layer for *transmitting value*

A Digression on Your Online Security

- There are many very alarming stories
- Best practice for crypto: a cold hardware wallet, purchased directly from its maker
- Improve your *overall* digital safety
- *Authentication* problems persist due to deeper problems with online *identity*
  - Self Sovereign Identity (e-book, slides)
  - Metadium
  - Solid, via Sir Tim Berners-Lee
  - and national schemes such as SingPass

Types of Smart Contracts

- "Vending machine" contracts: programs triggered by, or triggering, blockchain activity
- Smart legal contracts: Traditional contracts plus timestamps, tokens, auditing, document coordination, etc.
- Ethereum smart contracts: programs which control blockchain assets, executed over interactions on Ethereum's blockchain

Recommended Reading

coindesk's Beginner's Guide:
https://www.coindesk.com/information/

State of Blockchain and Crypto in 2020:
https://medium.com/bosagora/the-state-of-blockchain-cryptocurrency-in-2020-ad28aa6fa48f

Thank you

A BETTER WORLD BY DESIGN.

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN