DD2485 Programmable Society

Beyond Blockchain: Unpacking the Innovations of Internet Computer Protocol (ICP)

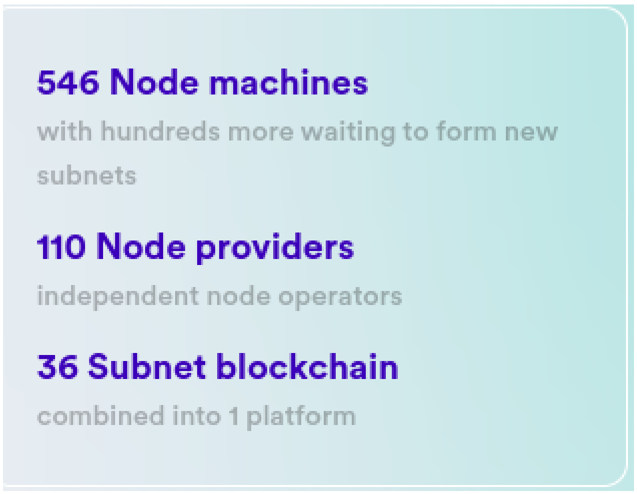
Name: Iosif Koen

First step in Internet Computer

The Internet Computer's mission is to improve and expand the internet's capabilities. It introduces the idea of an endless blockchain, which is differentiated by its speed, capacity, and reusability. It is regarded as the third important invention in the blockchain area. The Internet Computer Protocol (ICP) is a blockchain-based platform created to combine cloud computing with the benefits of blockchain technology, including statefulness and decentralization. On the other hand, it also combines the advantages of cloud computing—such as quick, scalable, and versatile computation—with the blockchain architecture.

Internet Computer protocol the basics

Web3 apps may function fully on-chain thanks to the Internet Computer Protocol (ICP), which provides a completely decentralized architecture that successfully addresses typical problems with corporate clouds, such security flaws and excessive expenses. Supporting fully decentralized apps and handling their frontend, backend, and data storage is one of the ICP's unique features. With the help of this feature, dApps[2] may take use of blockchain technology's security and decentralization while yet remaining quick and inexpensive. ICP uses a "true scaling" approach, creating new subnets on a regular basis to support an "unbounded" number of dApps with infinite data storage. In addition, the ICP gives the community the ability to take part as node providers; their official web page[3] explains this procedure. This promotes decentralization, variety, and resilience in addition to encouraging community work and organization. Apart of the the ideological part of it they achieve more decentralized, diverse and resilient IC(internet computing).


546 Node machines
with hundreds more waiting to form new subnets

110 Node providers
independent node operators

36 Subnet blockchain
combined into 1 platform

The Internet Computer imports an autonomous serverless cloud functionality to the public internet and as its developers claim it makes possible to develop almost any service or application in this decentralized network. IC is a highly capable platform/project

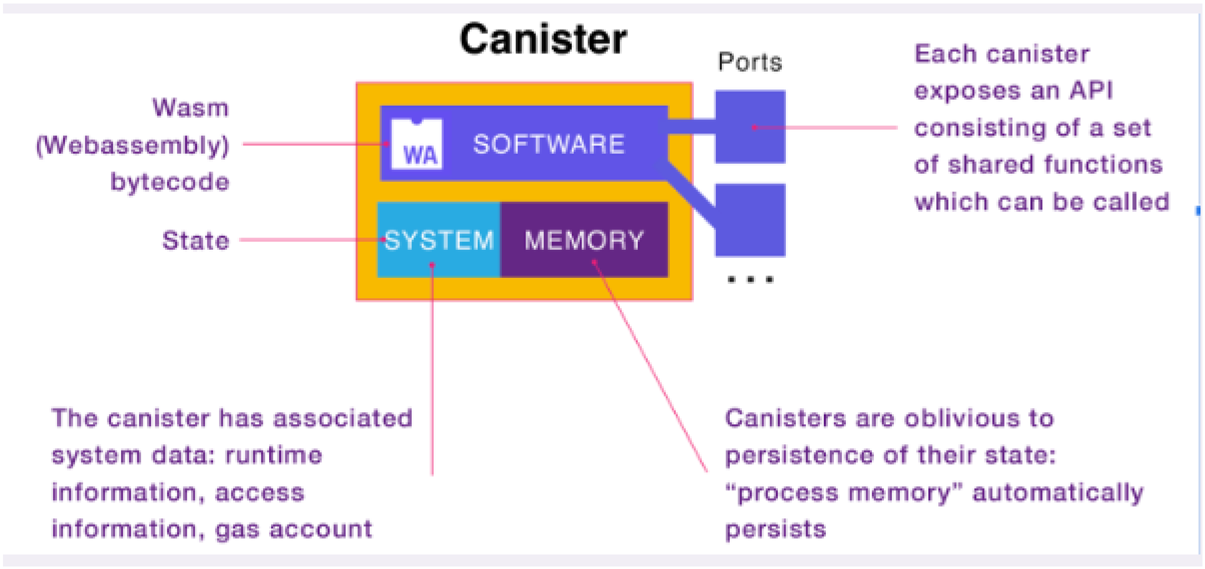
Internet Computer protocol in more depth

To help readers grasp the Internet Computer Protocol (ICP) and its underlying technologies better, the author will provide a quick explanation of a few of its features in this part. A brief overview of the ICP's architecture will also be included, along with an explanation of some of its main components.

The architecture

In the Internet Computer, smart contracts are known as "canister smart contracts" or simply "canisters," each comprising a bundle of WebAssembly[7] bytecode and smart contract storage. These canisters are hosted on subnets, which are the primary architectural components of the ICP. Subnets function as independent blockchains, operated by node machines or node providers (this will be further discussed in the following subsection). A single subnet can securely support tens of thousands of canister smart contracts, collectively using hundreds of gigabytes of memory. Presently, there are dozens of subnets, with a projection of thousands more in the future, as indicated by developers. Each canister hosted on a subnet has its code and data stored on every node in that subnet, and the code is executed by every node of the subnet. This method of replicating storage and computation allows the Internet Computer to achieve high fault tolerance, similar to how the Cassandra Database ensures high fault tolerance and availability. This replication feature stems from the core of the Internet Computer Protocol (ICP), which employs a high-throughput, low-latency consensus mechanism and an efficient virtual machine for WebAssembly execution, supported by a blockchain. The IC's multi-subnet architecture facilitates seamless interaction between smart contracts, akin to traditional microservices[8] architecture, but fully integrated within the blockchain.

A particularly interesting and innovative feature of the Internet Computer Protocol (ICP) is the way canisters communicate with each other. They use asynchronous messages, which means they neither block nor drop messages while sending. Instead, they process the response when it eventually arrives. This implies that the canisters (which can be thought of as capsules or packages containing elements previously described) can communicate and exchange messages without waiting, thereby enhancing the speed and efficiency of the network. With this approach, the ICP can easily scale by adding more subnets to the node machine. This suggests that there might eventually be a need for vertical scaling.


Canister
Ports
Wasm (Webassembly) bytecode
WA
SOFTWARE
State
SYSTEM
MEMORY
Each canister exposes an API consisting of a set of shared functions which can be called
The canister has associated system data: runtime information, access information, gas account
Canisters are oblivious to persistence of their state: "process memory" automatically persists

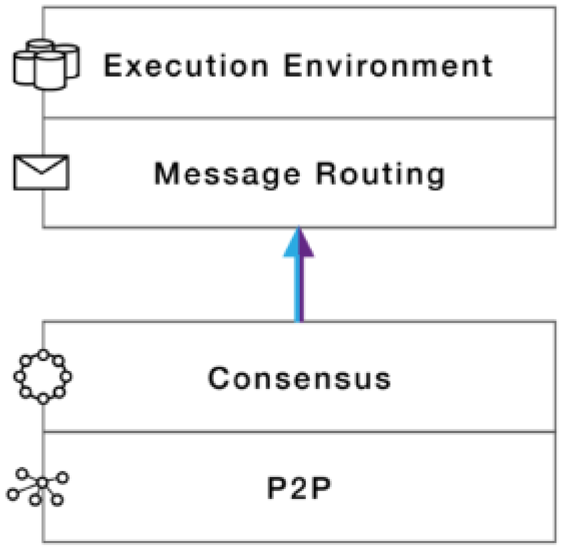
The Internet Computer Protocol (ICP) extensively utilizes chain-key cryptography[9], which is essentially a collection of advanced cryptographic protocols that enable its decentralized operation and unprecedented scalability. This toolbox comprises various technologies designed to robustly and securely address operational challenges. For example, it deals with how the network manages issues related to faulty or malicious nodes and protocol upgrades. An interesting example highlighted by the developers is the ability of nodes to easily add new subnets without having to validate the entire blockchain from the first block, unlike other blockchains such as Bitcoin or Ethereum. This functionality is referred to as **chain-evolution technology**. Furthermore, the **chain-key signature**, a component of chain-key cryptography, allows canisters to interact with other blockchains using threshold cryptography.

Last but not least, the Internet Computer Protocol (ICP) achieves full decentralization through its final component: governance. This decentralized governance is enabled by a tokenized Decentralized Autonomous Organization (DAO), known as the Network Nervous System (NNS). Through the NNS, each individual distributed application (dapp) can establish its own governance system. This is done by customizing and deploying a tokenized DAO based on the Service Nervous System (SNS) specifically for the dapp.

More information about the architecture of ICP can be found here: [10]

An overview of Internet Computer Protocol

The Internet Computer Protocol (ICP) forms the core of the Internet Computer. As the foundational protocol of the Internet Computer, the ICP operates as a four-layer protocol running on the nodes of each subnet. These nodes execute the core protocol, creating a blockchain-based replicated state machine that progresses independently from other subnets. This architecture of numerous, concurrently operating subnets is key to enabling the Internet Computer's virtually limitless scalability. Additionally, it's important to note that canisters can execute updates, altering their state. To maintain synchronization across all canisters on the hosted subnet nodes, the ICP ensures that every node executes the exact same messages in the same order, a process known as "full determinism." Each node on the Internet Computer runs a replica process, which is organized in a layered architecture consisting of four layers.



1. Execution
2. Message routing
3. Consensus
4. Peer-to-peer

Peer-to-peer

Starting with the lowest layer of the protocol, the Peer-to-Peer (P2P) layer is tasked with ensuring secure and reliable communication among the nodes within a subnet. This layer establishes a virtual peer-to-peer broadcast network, linking all the nodes of a subnet and creating a P2P communication fabric. Thanks to this layer's functionality, a node can efficiently broadcast network messages to its peers.

Consensus

Every blockchain requires a consensus mechanism for nodes to agree on the processing of messages. The ICP's consensus mechanism's role is to order inputs so that all replicas in a subnet process them in the same sequence. The ICP employs the ICC1 protocol, a straightforward and robust consensus protocol. It ensures the basic guarantees of a consensus algorithm, such as safety, which helps all replicas agree on the input order, and liveness, ensuring steady progress for the replicas. The developers explain in their paper that with `n replicas`, if at most `f < n/3` of the replicas are faulty, their consensus algorithm guarantees safety in a completely asynchronous manner, meaning under minimal communication assumptions. Additionally, they assure liveness with a partially synchronous assumption, logically setting time boundaries (denoted as δ) for message transmission.

Message routing

In every Internet Computer (IC) round, the message routing component receives a batch of messages processed by the Consensus. This batch is identical across each node in the subnet. Upon receipt, these messages are added to the input queue of their respective canister, a process known as induction. Subsequently, this triggers the execution round, potentially generating new messages in the output queues of the executed canisters. Once execution is complete, the messages in the output queues are dispatched by the message routing component to their intended recipients.

The recipients of these messages may be canisters on different subnets. The message routing layer is responsible for facilitating communication between canisters across various subnets. This process involves incorporating messages into a block and sending them to the receiving subnet, commonly referred to as cross-subnet messaging or XNet messaging. Secure XNet messaging is a crucial feature for the architecture of loosely-coupled subnets and is essential for the scalability of the Internet Computer (IC).

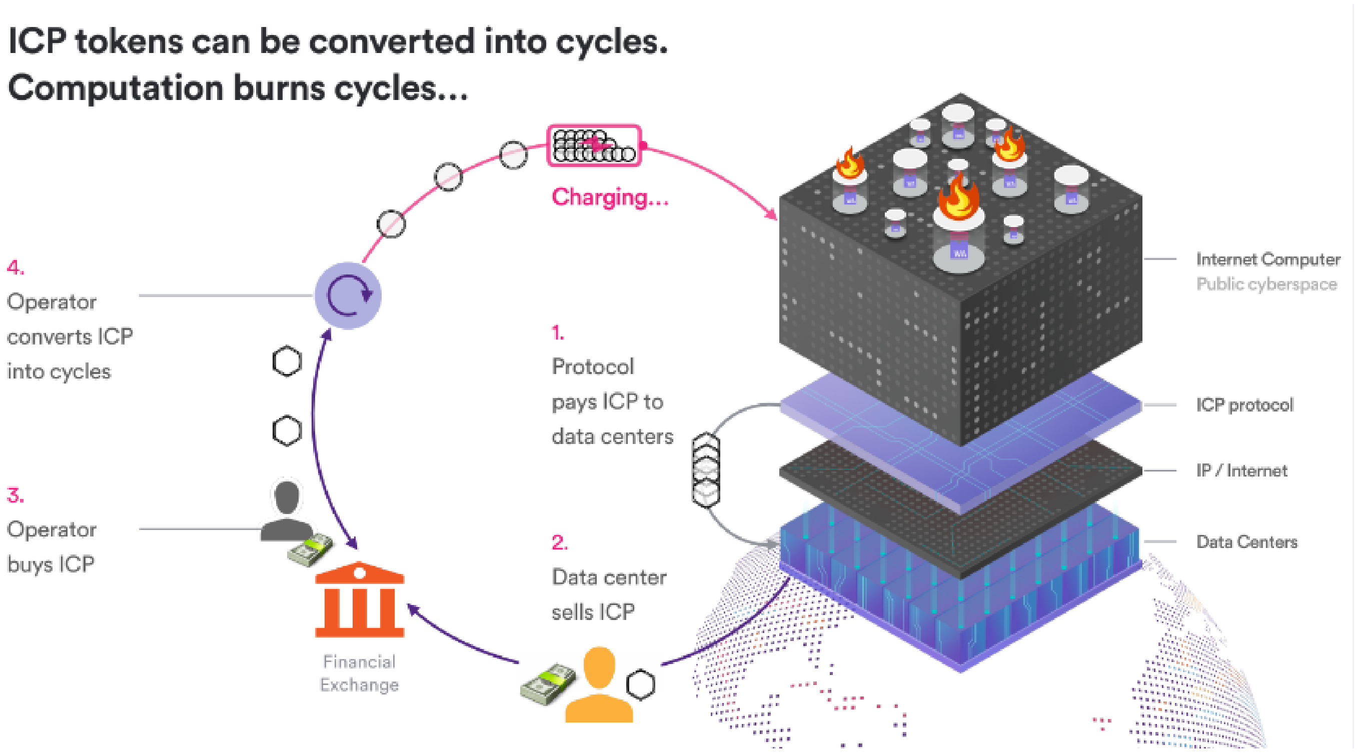
Another significant feature implemented by message routing is what is referred to as state certification. This involves the subnet certifying parts of its replicated state in every round, achieved in a decentralized manner. This certification process is crucial as it allows other subnets to verify the authenticity of streams between subnets. Additionally, it enables users to authentically access messages that have been previously submitted.[11]

Execution

The Execution layer is the highest level of the Internet Computer (IC) protocol stack, responsible for executing canister smart contract code. This code runs on a WebAssembly[12] virtual machine, which is deployed on every node. The WebAssembly bytecode operates deterministically and at near-native speed. The message routing layer transfers control of the messages to the Execution layer, which then processes the messages deterministically. This processing continues until either all messages in the canister queue are completed or the cycle limit for the round is reached, ensuring that round times are bounded.

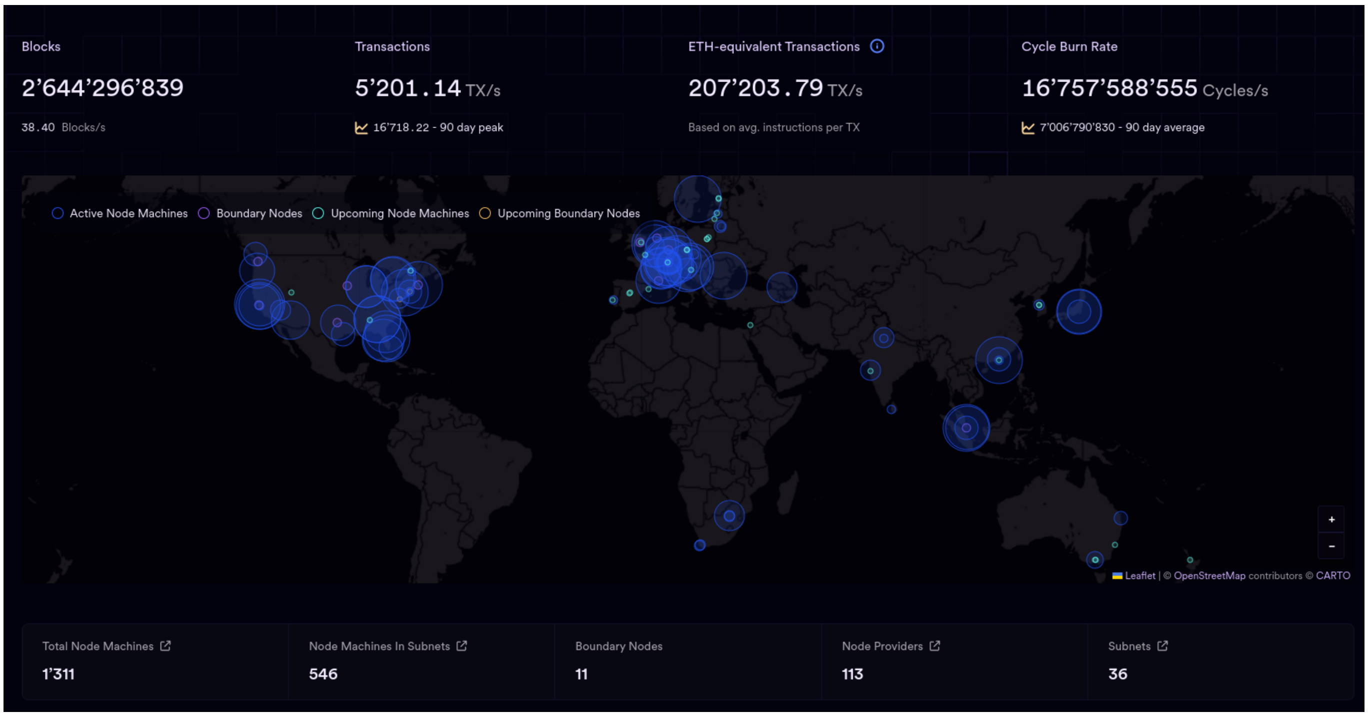
Important to mention here is that the execution layers implements very unique functionalities, such as:

- Deterministic time slicing(DTS)
- Concurrency
- Pseudorandom number generator



The node providers

The Internet Computer Protocol (ICP), as previously explained, is supported by a **decentralized** network of node machines. These node machines are physical hardware devices independently provided by data centers around the world, enabling the ICP to achieve a fully decentralized architecture. It is worth noting that the ICP diverges from traditional cloud service models. Unlike conventional cloud providers like AWS or Azure, which depend on a single private entity, the ICP relies on public utility and autonomous governance, essentially driven by the community.[4]



Blocks
2’644’296’839
38.40 Blocks/s

Transactions
5’201.14 TX/s
16’718.22 - 90 day peak

ETH-equivalent Transactions
207’203.79 TX/s
Based on avg. instructions per TX

Cycle Burn Rate
16’757’588’555 Cycles/s
7’006’790’830 - 90 day average

Total Node Machines	Node Machines In Subnets	Boundary Nodes	Node Providers	Subnets
1’311	546	11	113	36

Let's Google some smart contracts

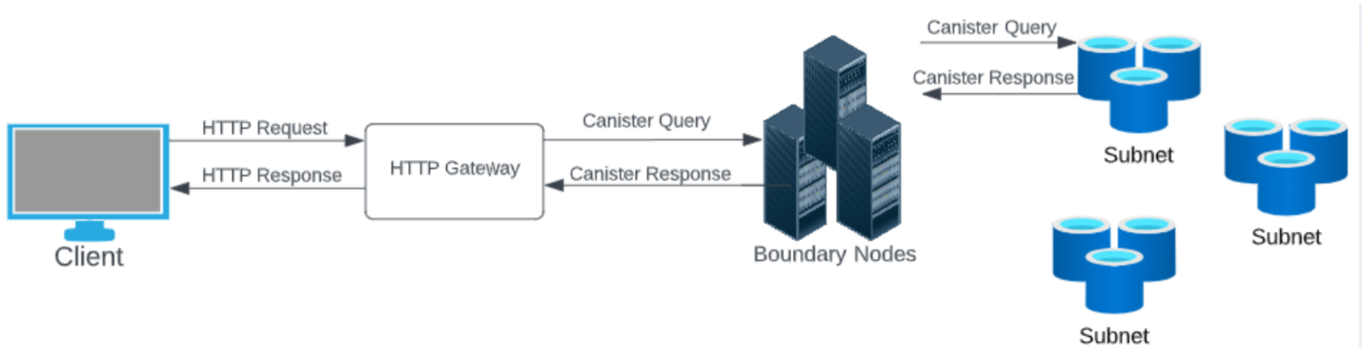
In this section, the author will delve deeper into how smart contracts facilitate web services. The focus will be on analyzing how the Internet Computer (IC) can host entire distributed applications (dapps), including both frontend and backend components. This analysis will explain the technology that enables this feature.

An introduction to contract-search

The Internet Computer (IC) stands out as the only blockchain capable of hosting an entire distributed application (dapp), including its frontend, backend, and data. As previously mentioned, canisters on the IC are bundles that contain both code and data. These canisters are capable of storing and delivering HTML, CSS, and JavaScript pages, as well as responding to API requests. Remarkably efficient, canisters can serve web pages in as little as 200ms, according to the creators. Furthermore, they can store up to 96GB of data. These features make browsing dapps on the Internet Computer akin to the traditional browsing experience on Web2, with applications hosted in the cloud.

The workflow

So to understand how this functionality it is possible to the Internet Computer we need to see first the picture and analyze its workflow.



- Client: As in every web application we need a user witch is the client to send some HTTP request.

- HTTP Gateway: Implements the HTTP Gateway protocol[13], this protocol converts HTTP requests in a format more understandable by canisters. And of course when a canister wants to send a response the HTTP Gateway it is the responsible to convert the response to HTTP response. The HTTP Gateway can be run either on the client site or on the boundary nodes or on independent specific server.
- Boundary nodes: These nodes are responsible to keep track the architecture of the Internet Computer structure, more specifically they keep track of subnets, the nodes on each subnet and the canisters that are running in each subnet. We can think the Boundary nodes as the master nodes of the structure or main nodes,(you can see how GFS works with it's master node).
- Canister: The final component of the workflow is nothing else than the Canister. We have already explain what a Canister is in previous sections. The only think that needs to be added here is: Tha anyone can send queries to the canister and it will executes the specific methods that it needs.

The deploy in Internet Computer

If a developer wishes their canister to serve a web application, they must implement a method capable of processing an HTTP request and returning an HTTP response. The canister can include and send back HTML, CSS, and JavaScript as part of this HTTP response. To enable this functionality, the developer needs to utilize the `http_request` method provided by the ICP. More information about this method can be found at reference [14].

Finally there is a simpler way to deploy a canister with a static web, and its called "asset canister". Read more about it on:[15]

HTTP Gateway Protocol

Since browsers send HTTP requests that canisters cannot directly interpret, the developers of the Internet Computer created the HTTP Gateway[13]. This software plays a crucial role by receiving the URL, converting it into an HTTP request, extracting the canister's ID, and then transforming the HTTP request into a canister request. After this conversion, it forwards the request to the boundary nodes.

Reflection

In this section the author will express personal thoughts and views around the IC and ICP, the actual project but also the white paper and their webpage.

Firstly, I must express that the entire Internet Computing project has made a remarkable impression on me. It has sparked my interest in blockchain technology, showcasing its potential for large-scale projects that integrate existing technologies, enhancing their efficiency and appeal. Internet Computing is an expansive project, extending its influence across various domains of blockchain and computer science. This includes areas like cloud computing, cryptography, distributed systems, software engineering, networking, and many more.

While the Internet Computer (IC) is an advanced and groundbreaking project, its vast scale can make it challenging to navigate. Often, readers might find themselves overwhelmed by its extensive documentation and the array of technologies it utilizes. Personally, I would appreciate a more clear and user-friendly website, offering a more straightforward approach to presenting information.

In conclusion, the Internet Computer is undoubtedly one of the most exciting projects currently available, boasting a vast technology stack appealing to computer scientists of all specializations. I highly recommend that everyone take a moment to explore it. As a final thought, I leave you with a question to think about it: Do we truly need the involvement of major players in our lives, or can the community forge a decentralized future on its own?

References

[1] Bitcoin: A Peer-to-Peer Electronic Cash System

[2] Dapp: https://en.wikipedia.org/wiki/Decentralized_application

[3] Node Provider Page: https://wiki.internetcomputer.org/wiki/Node_Provider_Documentation

[4] Node providers: https://internetcomputer.org/node-providers

[5] Actor model: https://en.wikipedia.org/wiki/Actor_model

[6] wasi2ic tool: https://github.com/wasm-forge/wasi2ic

[7] WebAssembly Languages: https://github.com/appcypher/awesome-wasm-langs

[8] Microservices: https://en.wikipedia.org/wiki/Microservices

[9] chain-key cryptography: https://internetcomputer.org/how-it-works/chain-key-technology/

[10] Architecture of ICP: https://internetcomputer.org/how-it-works/architecture-of-the-internet-computer/

[11] Message Routing: https://internetcomputer.org/how-it-works/message-routing/

[12] WebAssembly(Wasm) : https://webassembly.org/

[13] HTTP Gateway protocol: https://internetcomputer.org/docs/current/references/http-gateway-protocol-spec

[14] IC HTTP request method: https://internetcomputer.org/docs/current/references/ic-interface-spec/#ic-http_request

[15] IC host static webpage: https://internetcomputer.org/docs/current/samples/host-a-website/