Chapter 2: Decentralization

System Types

Centralized systems are conventional (client-server) IT systems in which there is a single authority that controls the system, and who is solely in charge of all operations on the system. All users of a centralized system are dependent on a single source of service.

In a **distributed system**, data and computation are spread across multiple nodes in the network.

What's the difference between distributed systems and parallel computing?

The main difference between these systems is that in a *parallel* computing system, computation is performed by all nodes simultaneously in order to achieve the result; for example, parallel computing platforms are used in weather research and forecasting, simulation, and financial modeling.

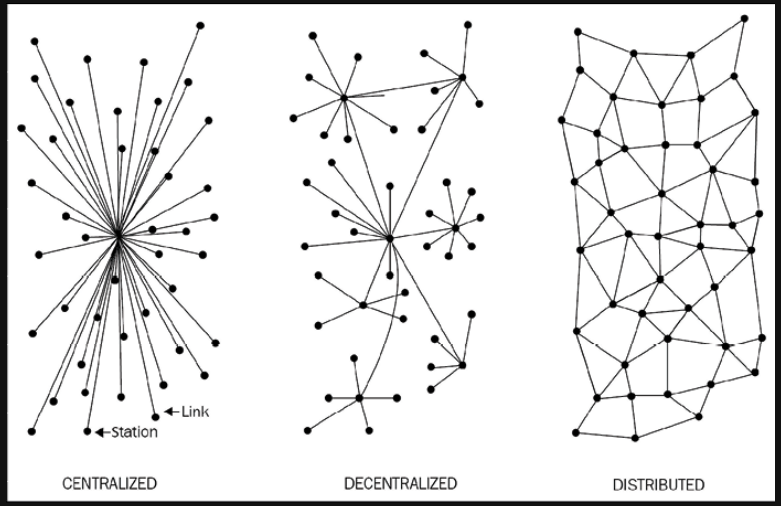
On the other hand, in a *distributed* system, computation may not happen in parallel and data is replicated across multiple nodes that users view as a single, coherent system.

- Variations of both of these models are used to achieve fault tolerance and speed. In the parallel system model, there is still a central authority that has control over all nodes and governs processing. This means that the system is still centralized in nature.

Decentralized Systems

A **decentralized system** is a type of network where nodes are not dependent on a single master node; instead, control is distributed among many nodes.

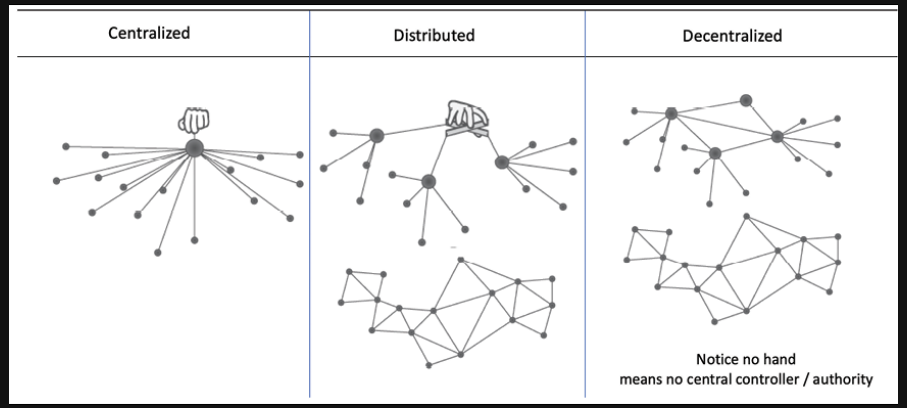
The basic idea of decentralization is to distribute control and authority to the peripheries of an organization instead of one central body being in full control of the organization.



Decentralized vs. Distributed

A decentralized system may look like a distributed system from a topological point of view, but it doesn't have a central authority that controls the network.

The critical difference between a decentralized system and distributed system is that in a distributed system, there is still a central authority that governs the entire system, whereas in a decentralized system, no such authority exists.



Centralized vs. Decentralized Systems (Networks/Applications)

Feature	Centralized	Decentralized
Ownership	Service provider	All users
Architecture	Client/server	Distributed, different topologies
Security	Basic	More secure
High availability	No	Yes
Fault Tolerance	Basic, single point of failure	Highly tolerant, as service is replicated
Collusion resistance	Basic, because it's under the control of a group or even single individual	Highly resistant, as consensus algorithms ensure defense against adversaries
Application Architecture	Single application	Application replicated across all nodes on the network
Trust	Consumers have to trust the service provider	No mutual trust required
Cost for consumer	Higher	Lower

What does this have to do with blockchains?

By design, blockchain is a perfect vehicle for providing a platform that does not need any intermediaries and that can function with many different leaders chosen via consensus mechanisms.

- This model allows anyone to compete to become the decision-making authority. A consensus mechanism governs this competition, and the most famous method is known as Proof of Work (PoW).
- Decentralization can be viewed from a blockchain perspective as a mechanism that provides a way to remodel existing applications and paradigms, or to build new applications, to give full control to users.

The technology exists to allow anyone to start a decentralized system and operate it with no single point of failure or single trusted authority.

- It can either be run autonomously or by requiring some human intervention, depending on the type and model of governance used in the decentralized application running on the blockchain.

But still, why decentralize?

There are many benefits of decentralization, including:

- Transparency
- Efficiency
- Cost saving
- Development of trusted ecosystems
- In some cases privacy and anonymity.

But decentralization is not without it's challenges

Before embarking on a journey to decentralize everything using blockchain and decentralized applications, it is essential that we understand that not everything can or needs to be decentralized.

Some challenges, such as *security requirements*, *software bugs*, and *human error*, need to be examined thoroughly.

Additionally, there are many other issues to consider, such as *latency*, *choice of consensus mechanisms*, *whether consensus is required or not*, and *where consensus is going to be achieved*.

- If consensus is maintained internally by a consortium, then a private blockchain should be used; otherwise, if consensus is required publicly among multiple entities, then a public blockchain solution should be considered.

Other aspects, such as *immutability*, should also be considered when deciding whether to use a blockchain or a traditional database.

So, before we decentralize, we need to ask ourselves a few very important questions, like:

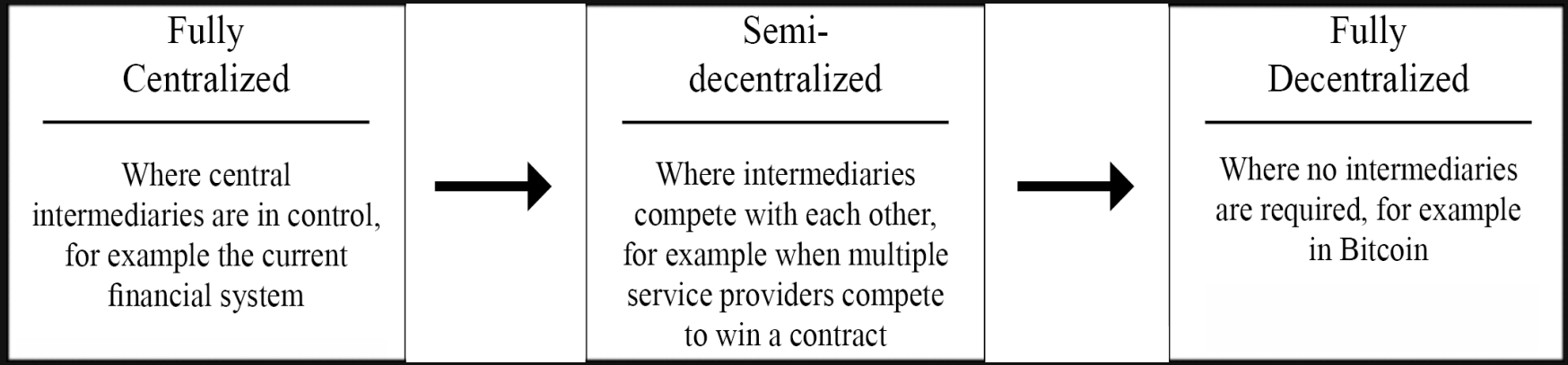
- Is a blockchain really needed?
- When is a blockchain required?
- In what circumstances is a blockchain preferable to traditional databases?

Question	Yes/No	Recommended solution
Is high data throughput required?	Yes	Use a traditional database.
	No	A central database might still be useful if other requirements are met. For example, if users trust each other, then perhaps there is no need for a blockchain. However, if they don't trust cannot be established for any reason, blockchain can be helpful.
Are updates centrally controlled?	Yes	Use a traditional database.
	No	You may investigate how a public/private blockchain can help.
Do users trust each other?	Yes	Use a traditional database.
	No	Use a public blockchain.
Are users anonymous?	Yes	Use a public blockchain.
	No	Use a private blockchain.

Question	Yes/No	Recommended solution
Is consensus required to be maintained within a consortium?	Yes	Use a public blockchain.
	No	Use a public blockchain.
Is strict data immutability required?	Yes	Use a blockchain.
	No	Use a central/traditional database.

Methods of Decentralization

Two methods can be used to achieve decentralization: **disintermediation** and **competition**.



1. Disintermediation

> Disintermediation refers to removing any intermediaries in a process and thus removing the need for a middle-man that might "cause" centralization.

Example:

Imagine that you want to send money to a friend in another country.

Centralized:

You go to a bank, which, for a fee, will transfer your money to the bank in that country. In this case, the bank maintains a central database that is updated, confirming that you have sent the money.

Decentralized:

With blockchain technology, it is possible to send this money directly to your friend without the need for a bank.

- All you need is the address of your friend on the blockchain. This way, the intermediary (that is, the bank) is no longer required, and decentralization is achieved by disintermediation.

2. Competition, or Contest-driven decentralization

> In the method involving competition, different service providers compete with each other in order to be selected for the provision of services by the system.

This paradigm does not achieve complete decentralization.

- However, to a certain degree, it ensures that an intermediary or service provider is not monopolizing the service.

In the context of blockchain technology, a system can be envisioned in which smart contracts can choose an external data provider from a large number of providers based on their reputation, previous score, reviews, and quality of service.

This doesn't mean this method will fully decentralize the system in question, but it allows smart contracts to make a *free choice* based on these criteria. This way, an environment of competition is created.

How to Decentralize a Pre-Existing System?

Arvind Narayanan and others have proposed a framework in their book *Bitcoin and Cryptocurrency* technologies that can be used to evaluate the decentralization requirements of a variety of issues in the context of blockchain technology.

This framework raises four questions whose answers provide a clear understanding of how a system can be decentralized:

1. What is being decentralized?
2. What level of decentralization is required?
3. What blockchain is used?
4. What security mechanism is used?

An Example of a Decentralized Framework

The four questions discussed previously are used to evaluate the decentralization requirements of this application. The answers to these questions are as follows:

1. Money transfer system
2. Disintermediation
3. Bitcoin
4. Atomicity

- The responses indicate that the money transfer system can be decentralized by removing the intermediary, implemented on the Bitcoin blockchain, and that a security guarantee will be provided via atomicity.
- Atomicity will ensure that transactions execute successfully in full or do not execute at all. We have chosen the Bitcoin blockchain because it is the longest established blockchain and has stood the test of time.
- Similarly, this framework can be used for any other system that needs to be evaluated in terms of decentralization. The answers to these four simple questions help clarify what approach to take to decentralize the system.
- To achieve complete decentralization, it is necessary that the environment around the blockchain also be decentralized. We'll look at the full ecosystem of decentralization next.

Blockchain and full ecosystem decentralization

The blockchain is a distributed ledger that runs on top of conventional systems. These elements include **storage**, **communication**, and **computation**.

Storage

Data can be stored directly in a blockchain, and with this fact it achieves decentralization.

- However, a significant disadvantage of this approach is that a blockchain is not suitable for storing large amounts of data by design. It can store simple transactions and some arbitrary data, but it is certainly not suitable for storing images or large blobs of data, as is the case with traditional database systems.

Two primary requirements here are **high availability** and **link stability**, which means that data should be available when required and network links also should always be accessible.

- An example of this is IPFS (Inter-Planetary File System) proposed by Juan Benet. The mechanisms behind IPFS will be further explained in later chapters, but this is a good introduction: link

There needs to be an incentive for nodes to join the network, and provide their storage and bandwidth.

The incentive mechanism for storing data is based on a protocol known as *Filecoin*, which pays incentives to nodes that store data using the Bitswap mechanism. The Bitswap mechanism lets nodes keep a simple ledger of bytes sent or bytes received in a one-to-one relationship. Also, a Git-based version control mechanism is used in IPFS to provide structure and control over the versioning of data.

> Bitswap is a message-oriented exchange protocol used to request and send content blocks between peers in a P2P content-addressable network.

Communication

In the current model of the internet, unconditional trust is placed in a central authority (the service provider) and users are not in control of their data.

- Even passwords are stored on trusted third-party systems.

Additionally, access to the Internet (the communication layer) is based on Internet Service Providers (ISPs) who act as a central hub for Internet users.

- If the ISP is shut down for any reason, then no communication is possible with this model.

Now imagine a network that allows users to be in control of their communication; no one can shut it down for any reason. This could be the next step toward decentralizing communication networks in the blockchain ecosystem. It must be noted that this model may only be vital in a jurisdiction where the Internet is censored and controlled by the government.

Blockchain has revived the vision of decentralization across the world, and now concerted efforts are being made to harness this technology and take advantage of the benefits that it can provide.

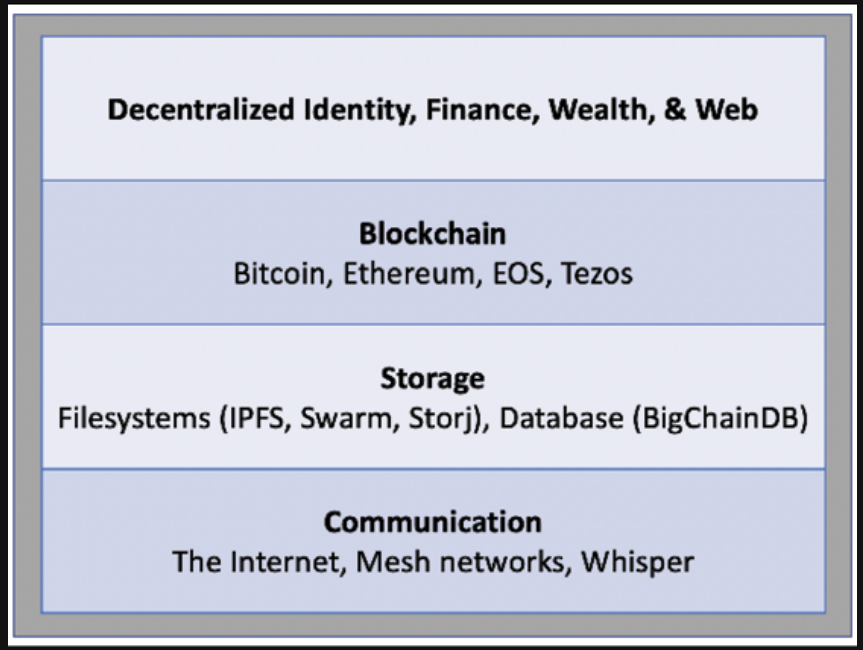
Computing Power and Decentralization

> Decentralization of computing or processing power is achieved by a blockchain technology such as Ethereum, where smart contracts with embedded business logic can run on the blockchain network.

Other blockchain technologies also provide similar processing-layer platforms, where business logic can run over the network in a decentralized manner.

The following diagram shows an overview of a decentralized ecosystem:

- In the bottom layer, the Internet or mesh networks provide a decentralized communication layer.
- In the next layer up, a storage layer uses technologies such as IPFS and BigChainDB to enable decentralization.
- Finally, in the next level up, you can see that the blockchain serves as a decentralized processing (computation) layer.
- Blockchain can, in a limited way, provide a storage layer too, but that severely hampers the speed and capacity of the system.
- Therefore, other solutions such as IPFS and BigChainDB are more suitable for storing large amounts of data in a decentralized way.
- The Identity and Wealth layers are shown at the top level.



The blockchain is capable of providing solutions to various issues relating to decentralization. A concept relevant to identity known as Zooko's Triangle requires that the naming system in a network protocol is secure, decentralized, and able to provide human-meaningful and memorable names to the users.

Conjecture has it that a system can have only two of these properties simultaneously.

However, recently, there have been strides to achieve all three at once, but not without it's caveats. See Namecoin.

But still, decentralization is not always the solution

Decentralization may not be appropriate for every scenario. Centralized systems with well-established reputations tend to work be er in many cases. For example, email platforms from reputable companies such as Google or Microsoft would provide a better service than a scenario where individual email servers are hosted by users on the Internet.

Terminology

Going further, we'll need to know some terms that'll pop up from here on out. It's vital to understand what these are, as later concepts will build off of these terms.

Smart Contracts

> A smart contract is a software program that usually runs on a blockchain. A smart contract usually contains some business logic and a limited amount of data. The business logic is executed if specific criteria are met. Actors or participants in the blockchain use these smart contracts, or they run autonomously on behalf of the network participants.

Autonomous agents

> An Autonomous Agent (AA) is an artificially intelligent software entity that acts on the behalf of its owner to achieve some desirable goals without requiring any or minimal intervention from its owner.

Decentralized organizations

> DOs are software programs that run on a blockchain and are based on the idea of actual organizations with people and protocols. Once a DO is added to the blockchain in the form of a smart contract or a set of smart contracts, it becomes decentralized and parties interact with each other based on the code defined within the DO software.

Decentralized autonomous organizations

> Just like DOs, a decentralized autonomous organization (DAO) is also a computer program that runs on top of a blockchain, and embedded within it are governance and business logic rules.

- In a DAO, the code is considered the governing entity rather than people or paper contracts. However, a human curator maintains this code and acts as a proposal evaluator for the community. DAOs are capable of hiring external contractors if enough input is received from the token holders (participants).

DOs vs. DAOs

The main difference between these two is that DAOs are autonomous, which means that they are fully automated and contain artificially intelligent logic.

- DOs, on the other hand, lack this feature and rely on human input to execute business logic.

How legal are DAOs?

DAOs do not have any legal status, even though they may contain some intelligent code that enforces certain protocols and conditions.

However, these rules have no value in the real-world legal system at present.

One day, perhaps an AA (that is, a piece of code that runs without human intervention) commissioned by a law enforcement agency or regulator will contain rules and regulations that could be embedded in a DAO for the purpose of ensuring its integrity from a legalistic and compliance perspective.

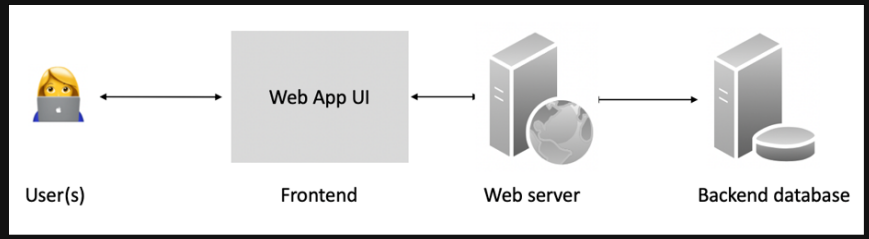
DApps

> A DApp—pronounced Dee-App, or now more commonly rhyming with app—is a software application that runs on a decentralized network such as a distributed ledger.

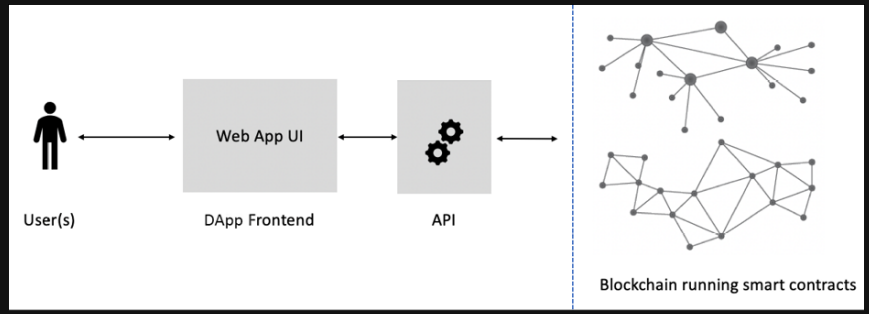
All the ideas mentioned up to this point come under the broader umbrella of decentralized applications, abbreviated to DApps.

DApp vs. Traditional App

Traditional apps commonly consist of a user interface and usually a web server or an application server and a backend database. This is a common client/server architecture.



A **DApp** on the other hand has a blockchain as a backend and can be visualized as depicted in the following diagram. The key element that plays a vital role in the creation of a DApp is a smart contract that runs on the blockchain and has business logic embedded within it.



In the last few years, the expression DApp has been increasingly used to refer to any end-to-end decentralized blockchain application, including a user interface (usually a web interface), smart contract(s), and the host blockchain.

The clear distinction between different types of DApps is now not commonly referred to, but it does exist.

Often, DApps are now considered just as apps (blockchain apps) runningon a blockchain such as Ethereum, Tezos, NEO, or EOS without any particular reference to their type.

DApp types

DApps at a fundamental level are software programs that execute using either of the following methods. They are categorized as **Type 1**, **Type 2**, or **Type 3** DApps:

1. *Type 1*: Run on their own dedicated blockchain, for example, standard smart contract based DApps running on Ethereum. If required, they make use of a native token, for example, ETH on Ethereum blockchain.
2. *Type 2*: Use an existing established blockchain. That is, make use of Type 1 blockchain and bear custom protocols and tokens, for example, smart contract based tokenization DApps running Ethereum blockchain.
3. *Type 3*: Use the protocols of Type 2 DApps; for example, the SAFE Network uses the OMNI network protocol.

DApp Example

Another example to understand the difference between different types of DApps is the USDT token (Tethers).

- The original USDT uses the OMNI layer (a Type 2 DApp) on top of the Bitcoin network.
- USDT is also available on Ethereum using ERC20 tokens. This example shows that a USDT can be considered a Type 3 DApp, where the OMNI layer protocol (a Type 2 DApp) is used, which is itself built on Bitcoin (a Type 1 DApp).
- Also, from an Ethereum point of view USDT can also be considered a Type 3 DApp in that it makes use of the Type 1 DApp Ethereum blockchain using the ERC20 standard, which was built to operate on Ethereum.

Requirements of a DApp

According to the definition provided in a whitepaper by *Johnston et al. in 2015, The General Theory of Decentralized Applications, DApps*:

For an application to be considered decentralized, it must meet the following criteria.

1. The DApp should be fully open source and autonomous, and no single entity should be in control of a majority of its tokens. All changes to the application must be consensus-driven based on the feedback given by the community.
2. Data and records of operations of the application must be cryptographically secured and stored on a public, decentralized blockchain to avoid any central points of failure.
3. A cryptographic token must be used by the application to provide access for and incentivize those who contribute value to the applications, for example, miners in Bitcoin.
4. The tokens (if applicable) must be generated by the decentralized application using consensus and an applicable cryptographic algorithm. This generation of tokens acts as a proof of the value to contributors (for example, miners).

DApp Example: Decentralized Finance (DeFi)

Traditionally, finance is a business that is almost impossible to do without the involvement of a trustedthird party.

- It can either be a bank or some other financial firm; consumers have to trust a central authority to do business on their behalf and provide the services.

There are rules, policies, procedures, and strict regulations that govern this ecosystem.

This control and management is of paramount importance for the integrity of the entire financial ecosystem.

Still, since there is always a central party that is required in every single transaction consumers do, this approach has some disadvantages.

Centralized Finance (CeFi)

Some main disadvantages are listed as follows:

- Access barrier: Access to financial services and banking requires a rigorous onboarding process involving KYC and other relevant checks and procedures. Even though this is extremely important for the integrity of the existing financial system, it can become a barrier at times for millions of unbanked people all around the world, especially in third world countries.
- High cost: Financial services can be costly in some scenarios, especially investment-related activities, and could be seen as a barrier toward entering the financial services industry.
- Transparency issues: There are concerns around transparency and trust due to the proprietary nature of the financial industry.
- Siloed: Most current financial industry solutions are proprietary and are owned by their respective organization. This results in interoperability issues where systems from one organization are unable to talk with another.

DeFi

With the decentralization of finance though, not only are these barriers removed, but a number of advantages are added as well, such as inclusion, easy access and cheaper services.

You can find the latest rankings and analytics of the DeFi protocol on this excellent website: DeFi Pulse.