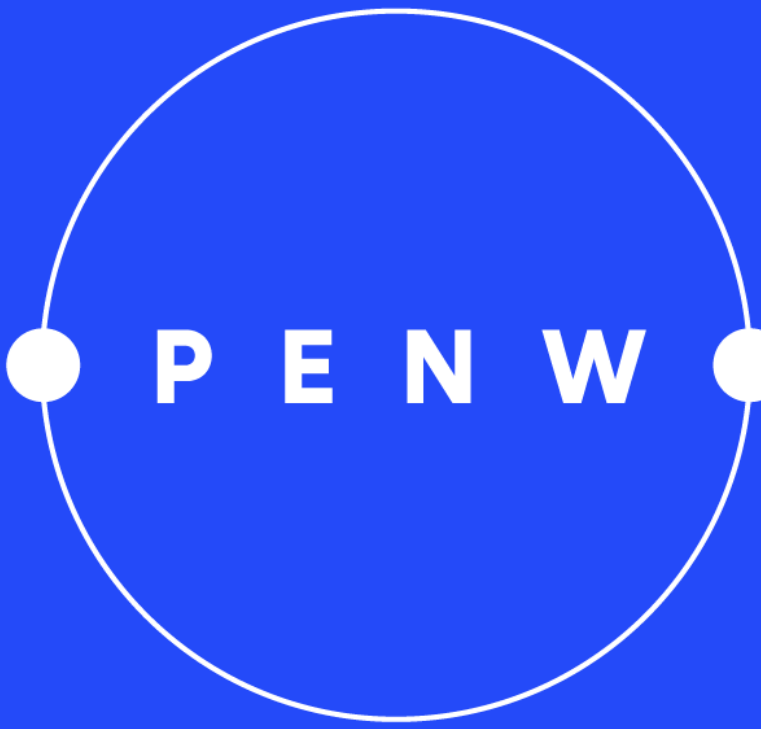


OpenWork Paper 2023

By Armand Poonawala

Contents

Introduction

Human beings are naturally a co-dependent species, and from as early as history can document, we have organised ourselves into groups using the technology available in order to perform some form of work.

Unlike other animals such as ants, which have always organised themselves in groups based on biological evolution, human beings re-organise when technology evolves.

The more significant the technological evolution, the more we re-organise.

One of the most significant evolutions of technology in recent history has been the advent of the internet.

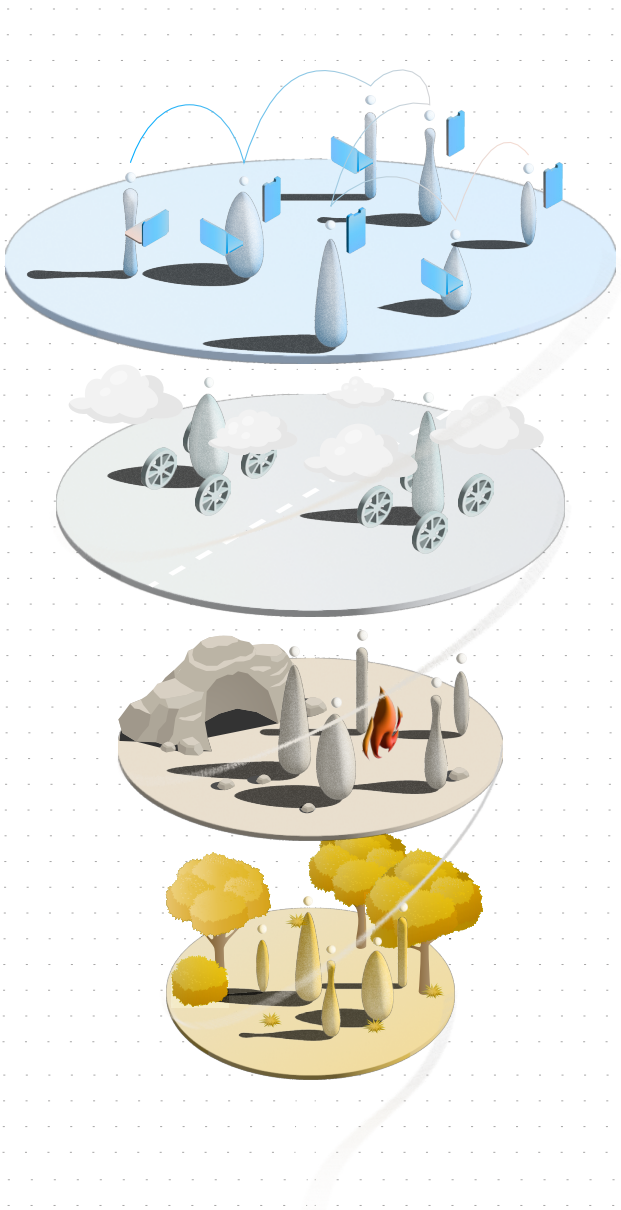
Pre-internet, we couldn't communicate digitally and needed to meet physically for most communication.

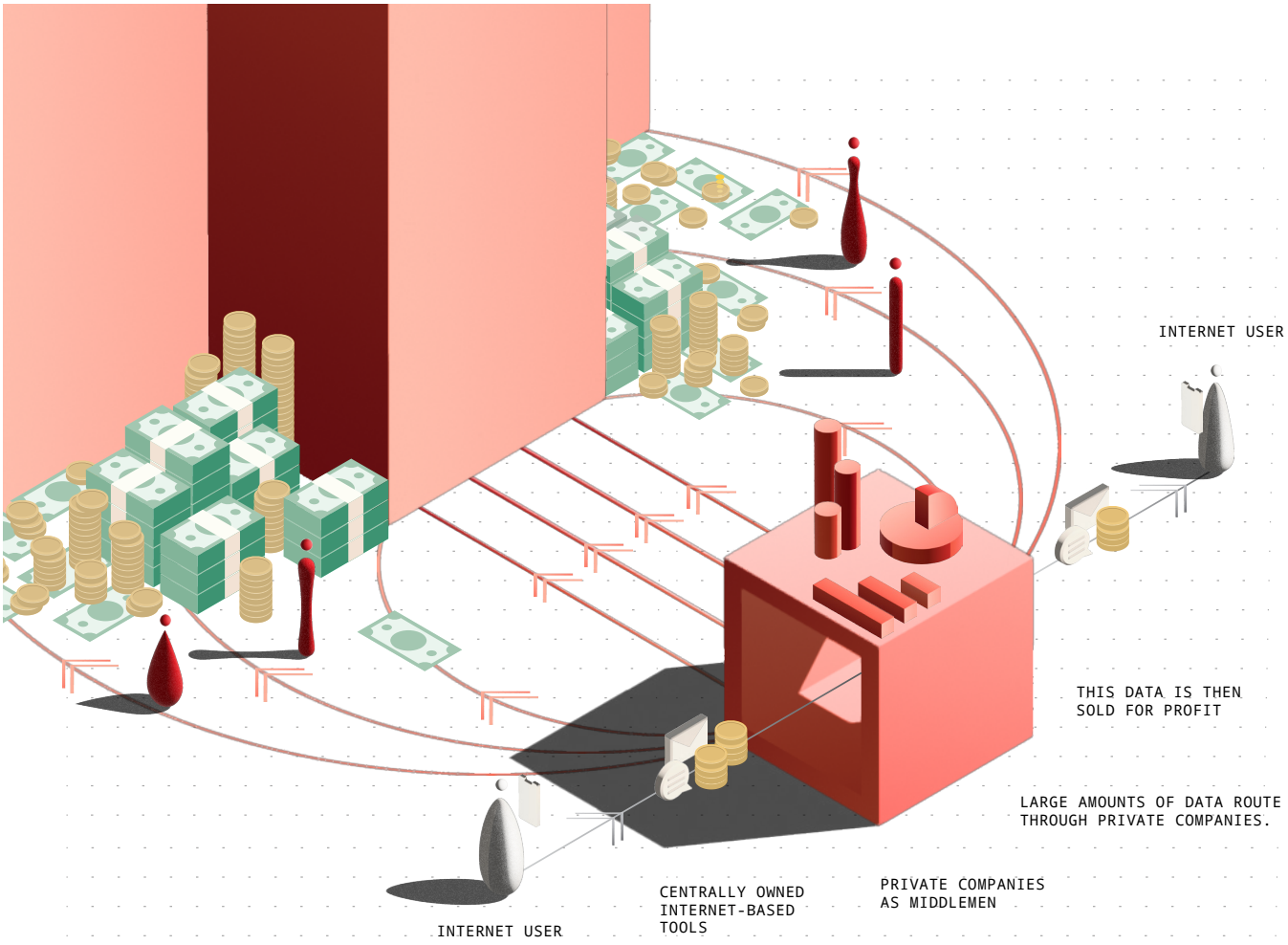
Post-internet, we can seamlessly communicate with each other remotely.

With the birth of the internet, which is not owned by one organisation but is open-source, or de-centralised, came all sorts of centrally owned internet-based tools that were created.

For example, sending a simple message or email goes through a centralised company on the internet, even today. Since these tools solved major problems (in comparison to no tool existing at all), they allowed for centralised organisations to extract most of the value for themselves despite being fully dependent on the users. Most of them collect a large amount of data on user behaviour that only they can access and profit off of.

Humans reorganizing in response to technology





Centralised companies acting as middlemen

As a result, a ridiculously disproportionate amount of the new wealth created in the world from the advent of the internet, was made mainly by the owners of these centralised organisations. From 2020 to 2022, the richest 1 percent grabbed nearly two-thirds of all new wealth worth $42 trillion, almost twice as much money as the bottom 99 percent of the world's population, reveals a new Oxfam report.

Humans using the internet by default became customers of these centralised organisations that exploited them and although the tools themselves were completely dependent on them, there was no way for them to reorganise themselves without the private companies acting as middlemen - until the introduction of blockchain technology.

Simply put, blockchain technology allows for humans to function on the internet without the need for centralised companies acting as middlemen and extracting most of the value.

It does this by creating autonomous, internet-based systems that are user-owned. These systems work smoothly through incentives for positive actors, and disincentives for negative actors thus eliminating the need for any centralised (and potentially exploitative) organisation to be involved.

The technology usually involves a transparent, distributed and immutable database that keeps records of all actions that occur on the system so that everyone on the internet can access, use and trust the system as a whole. This proves to all that it is scientifically accountable and gives equal data to all.

Through this system, the users can collectively own a fair portion of the wealth generated by the technology and the adoption of it can lead to a more equitable world.

Blockchain technology's first application was in decentralising how humans store value or transact through a decentralised currency called Bitcoin. Ethereum, which came next, was created to easily allow all sorts of decentralised tools to be built on the internet using a steady base - without needing to reinvent the wheel of basic blockchain layer 1 technology.

Thousands of decentralised projects have recently sprung up as a result of this, and the technology is right at the cusp of being adopted by humanity as a whole right across industries.

Since the technology is in its early stages, it's yet to decentralise how humans work on the internet, which is precisely the intention behind OpenWork. OpenWork intends to reorganise how humans work on the internet without any need for middlemen, where the users gain fair value from the adoption of the technology.

If successful, no longer would only a handful of companies have exclusive access to information on how humans are working on the internet, which currently allows them to exploit the most value from it. Freelance platforms like UpWork and Fiverr process millions of jobs a year or a majority of all freelance jobs done online, and keep all the data to themselves, possibly exploiting it for further profits.

The information asymmetry allows them to become gatekeepers of access to job givers or seekers on the internet. Anyone looking to use their platforms to connect with job givers or seekers must pay commission structures of 20% on average and give the platforms all the data on the job, which they do not disclose to the public despite dependence on users on both ends.

The Fundamentals of OpenWork

Layer 1 for Decentralised Work

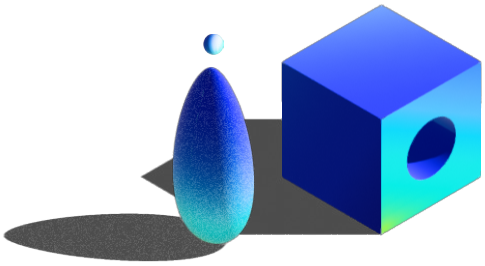
As a first step, OpenWork proposes to create a decentralised, transparent & immutable database documenting how people are working on the internet through blockchain technology. This can be thought of as a "layer 1 for decentralised work" on the internet.

The OpenWork Blockchain

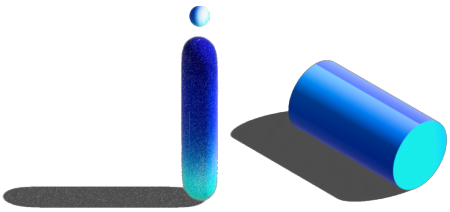
Job givers or takers create their own accounts which are accessed by private keys on the OpenWork blockchain. The owner of the private key owns & operates their own work account which has a public key that exists on the OpenWork blockchain. No centralised authority can access, own or manipulate any data associated with it.

A job giver can post a job requirement on OpenWork which initiates the creation of a potential entry on the database. Job takers can apply to take up the job. If the job giver agrees, the job then commences and a payment amount is locked in using a smart contract. Once the job taker delivers the job and if both agree it is done, the payment is released through the smart contract. *(Fig.1 - TD.1)*

Review details from both ends are then added to the block. The job requirement and fulfilment details are then added to the OpenWork blockchain with associated public keys. Keys decide on how much detail of the job is revealed to the public as some information could be sensitive.



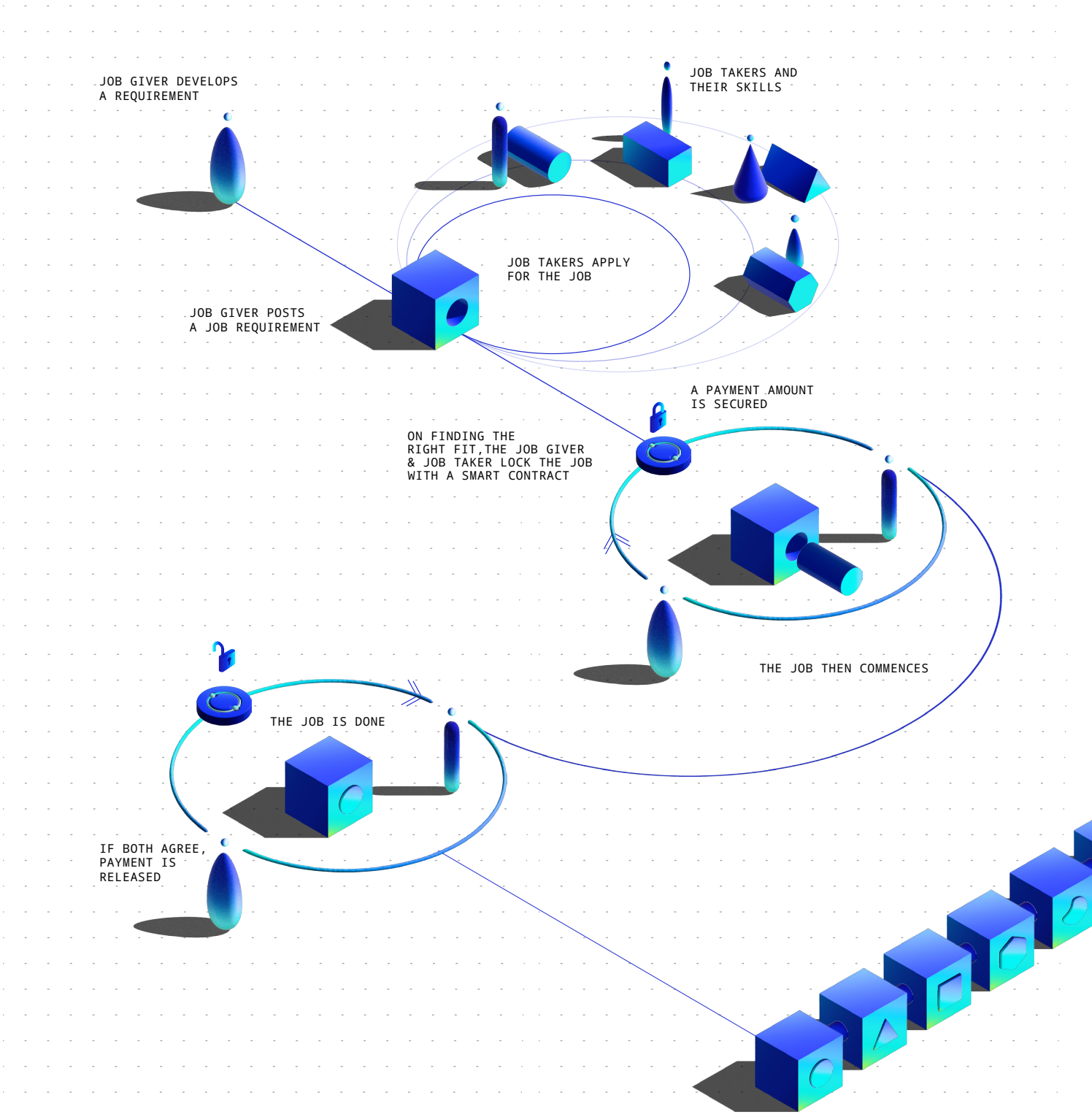
Job Giver and their Requirement



Job Doer and their Skill

After multiple jobs are completed involving the job givers or takers, accounts or public keys earn credibility and other keys can start to more easily enter into contracts with them.

Fig.1 - TD.1



Imagine anyone on the internet having equal access to a transparent database of every job that was given to someone to do, containing - who it was given to, who the job giver was, what skill they had, what the outcome was and how much they got paid for it.

For example, a graphic designer who has delivered 1000 jobs to 1000 different people would have all details of each job done stored on the blockchain where there is proof of the quality of any work done in the past. This is a good indicator of how this graphic designer would function in the future.

Further, any key that performs an action risks the credibility of their profile as any incorrectly handled job would also leave a permanent mark on their record, thus the incentive to not do bad work is high.

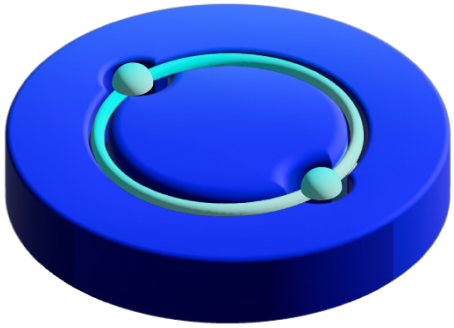
The graphic designer then can enter into a new contract for a new job and then the job giver knows that the outcome of the job will be added onto the blockchain, increasing trust.

As opposed to centralised platforms with 20% commissions, near-zero commissions would be applied on each job on OpenWork.

Once keys start to gain decentralised credibility for job giving or taking, multiple applications on top of OpenWork can be built using this credibility as a base which is what forms layer 2.

The OpenWork Token

Although the OpenWork blockchain will be owned by everyone and belongs to the internet itself, in order for people to come together to govern, contribute, grow & upgrade OpenWork, the OpenWork token is created.



OpenWork Token

On-Chain Smart Contracts

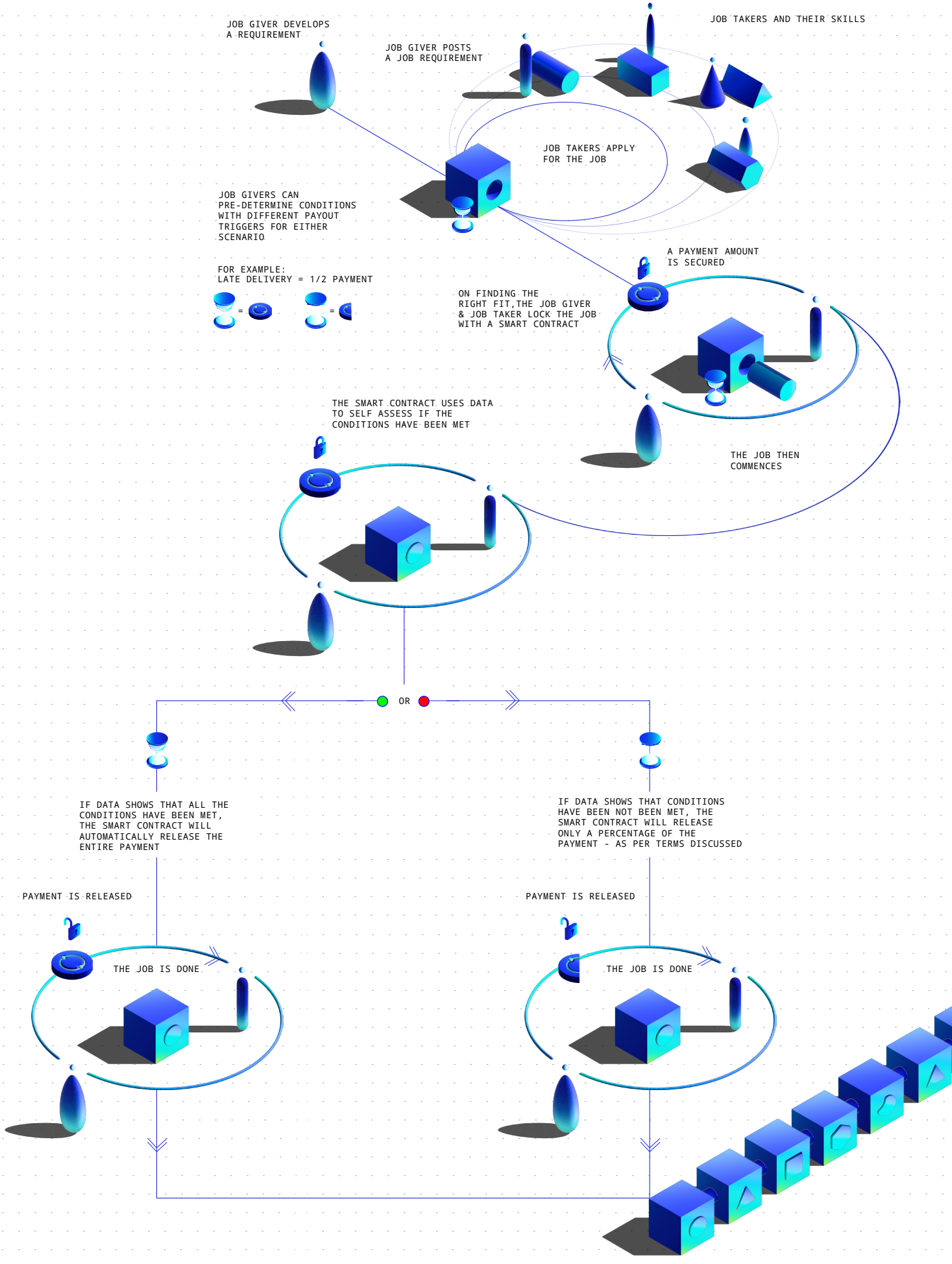
Smart contracts, or self-executing contracts based on pre-determined conditions being met, can be formed between job giver and job taker. These essentially make contracts programmable, without needing a third party to review if conditions are met or not based on which terms execute. They are considered on-chain when all data needed to execute exists on the blockchain itself, like time of delivery. (Fig. 2)

An example smart contract would be “If delivery of the job is done by 10th February 2023, pay $100. For every day the job has been delayed, pay $10 less”. Here, the job giver and doer just need to agree on if the job was delivered or not. The smart contract itself assesses when the same happened and pays accordingly. On-chain smart contracts work seamlessly when the job giver and job taker themselves have no dispute regarding the delivery of the job itself.

The OpenWork DAO

The OpenWork DAO will govern the OpenWork project based on token ownership.

Fig. 2 - On-Chain Smart Contracts



Athena - Skill Oracles

The problem with a simple decentralised system where one node pays another for work done based on mutual consent occurs when the parties do not consent.

There is a situation where for example the job doer believes the job is complete whereas the job giver disagrees. If the job doer is objectively correct, it would be unfair for the job doer not to be paid. Therefore there is a need for a decentralised dispute mechanism that would solve the fairness problem.

We propose to create Athena, a protocol that works with "skill oracles." Before we come to the solution applied here, here is a breakdown of how skill-oracles work:

1. We now introduce the concept of a "skill oracle". Job doer keys on the platform are assigned skills through a process later described here. A group of keys with a particular skill assigned, combined with the use of a smart contract oracle that only they can give inputs to, form what we call a skill oracle. For example, there can be a skill oracle for "Basic English to French translation" consisting of keys with that skill. Only these keys can give inputs related to this particular skill.

2. The skill oracle can be asked any question relating to the skill and will give an answer back that is most likely correct due to the incentive system in place which is described here: If a question is posed to the oracle, any of its members can stake their tokens and provide an answer. If the answer is thought to be incorrect by other members, they can stake their tokens in turn and dispute the same with another answer. Other members at this point can stake their tokens and side with one of the answers or provide another. Each answer or side taken counts as 1 vote. The answer with the most votes wins and those that side with it are rewarded tokens while losing answers and those that side with them lose tokens.

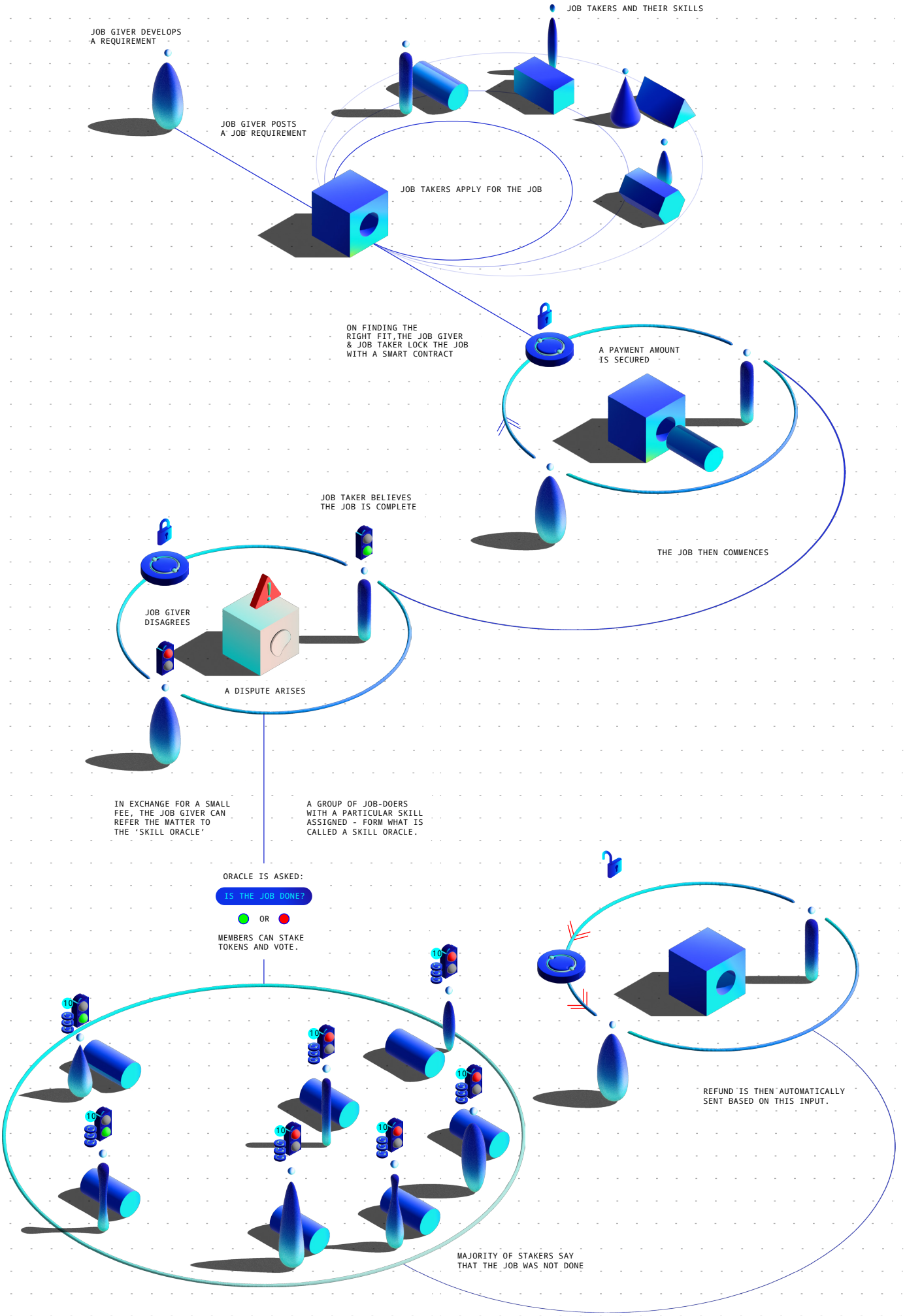
   This ensures that in almost all cases, the answer if given by someone with tokens to win or lose in the exchange will be right. (*Fig.3 - TD.2)*

3. With the use of the skill oracle, if there is a dispute, the job giver can consent to send the dispute over to the skill oracle to resolve for a small percentage fee that is taken from the locked amount. The dispute would go to the skill oracle, members of which can stake tokens and vote on the solution needed which can be directly executed by the smart contract. Winning stakers are awarded the fee and any losing staker tokens.

   For example, if the oracle says the job is simply not done by the job doer, the funds would be refunded. A real world example would be as follows: a job giver needing a word document translated from English to French. A contract is then entered into and the job doer claims to submit the translation from English to French. The job giver however, believes the translation was not done as per the brief where basic French translation was done incorrectly. The job giver can then send the dispute to the oracle which in turn assesses it. Keys with the English to French translation skill then assess, stake their tokens. Majority of stakers say that the job was not properly translated and the refund is due. The refund is then automatically sent based on this input.

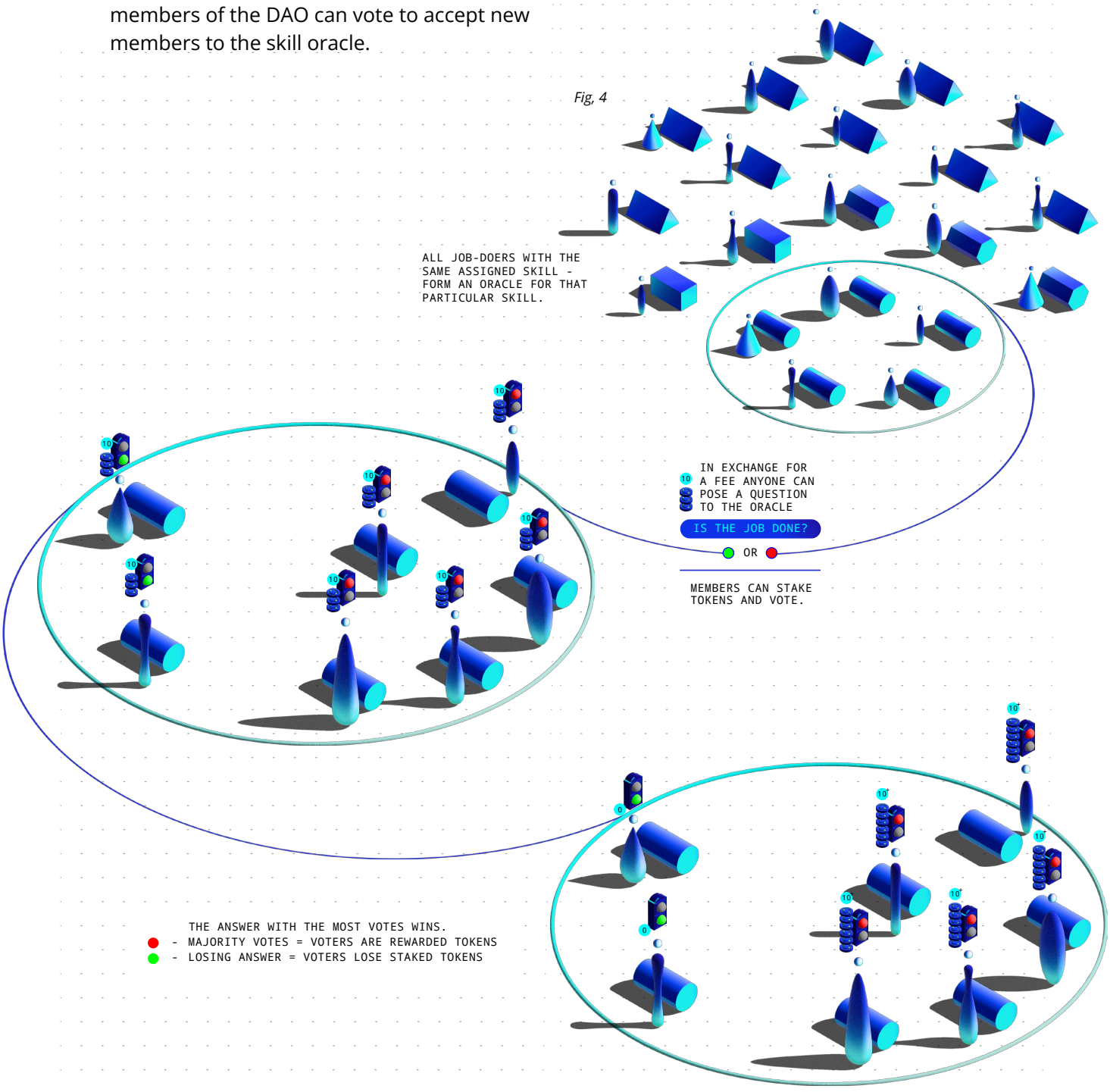
4. In order to create a new skill oracle, the DAO must decide on the creation of one for a new skill, for example "Basic English to French translation". From here, any member of the DAO can stake their tokens and propose to the DAO that they would like to add a new key to this skill oracle. If the DAO accepts the key, this key gets the skill assigned and the proposing member gets an incentive. If the DAO rejects the member, the staked tokens are taken away. Through the incentive, anyone proposing to add a member does so after carefully verifying their skill personally.

Fig.3 - TD.2


JOB TAKERS AND THEIR SKILLS
JOB GIVER DEVELOPS A REQUIREMENT
JOB GIVER POSTS A JOB REQUIREMENT
JOB TAKERS APPLY FOR THE JOB
ON FINDING THE RIGHT FIT,THE JOB GIVER & JOB TAKER LOCK THE JOB WITH A SMART CONTRACT
A PAYMENT AMOUNT IS SECURED
THE JOB THEN COMMENCES
JOB TAKER BELIEVES THE JOB IS COMPLETE
JOB GIVER DISAGREES
A DISPUTE ARISES
IN EXCHANGE FOR A SMALL FEE, THE JOB GIVER CAN REFER THE MATTER TO THE 'SKILL ORACLE'
A GROUP OF JOB-DOERS WITH A PARTICULAR SKILL ASSIGNED - FORM WHAT IS CALLED A SKILL ORACLE.
ORACLE IS ASKED:
IS THE JOB DONE?
OR
MEMBERS CAN STAKE TOKENS AND VOTE.
REFUND IS THEN AUTOMATICALLY SENT BASED ON THIS INPUT.
MAJORITY OF STAKERS SAY THAT THE JOB WAS NOT DONE

5. Once the skill oracle is formed with minimum 20 active members or keys with that skill, it is considered functional. A member is considered active if they have been added within the last 30 days or perform a job on the OpenWork network or function on the oracle within the last 30 days. *(Fig, 4)* Initial activation entails acceptance into the oracle. If no actions exist on the oracle for the member to perform, they can mark themselves as active through a button. Once a skill oracle is active, only members with the skill and no longer just members of the DAO can vote to accept new members to the skill oracle.

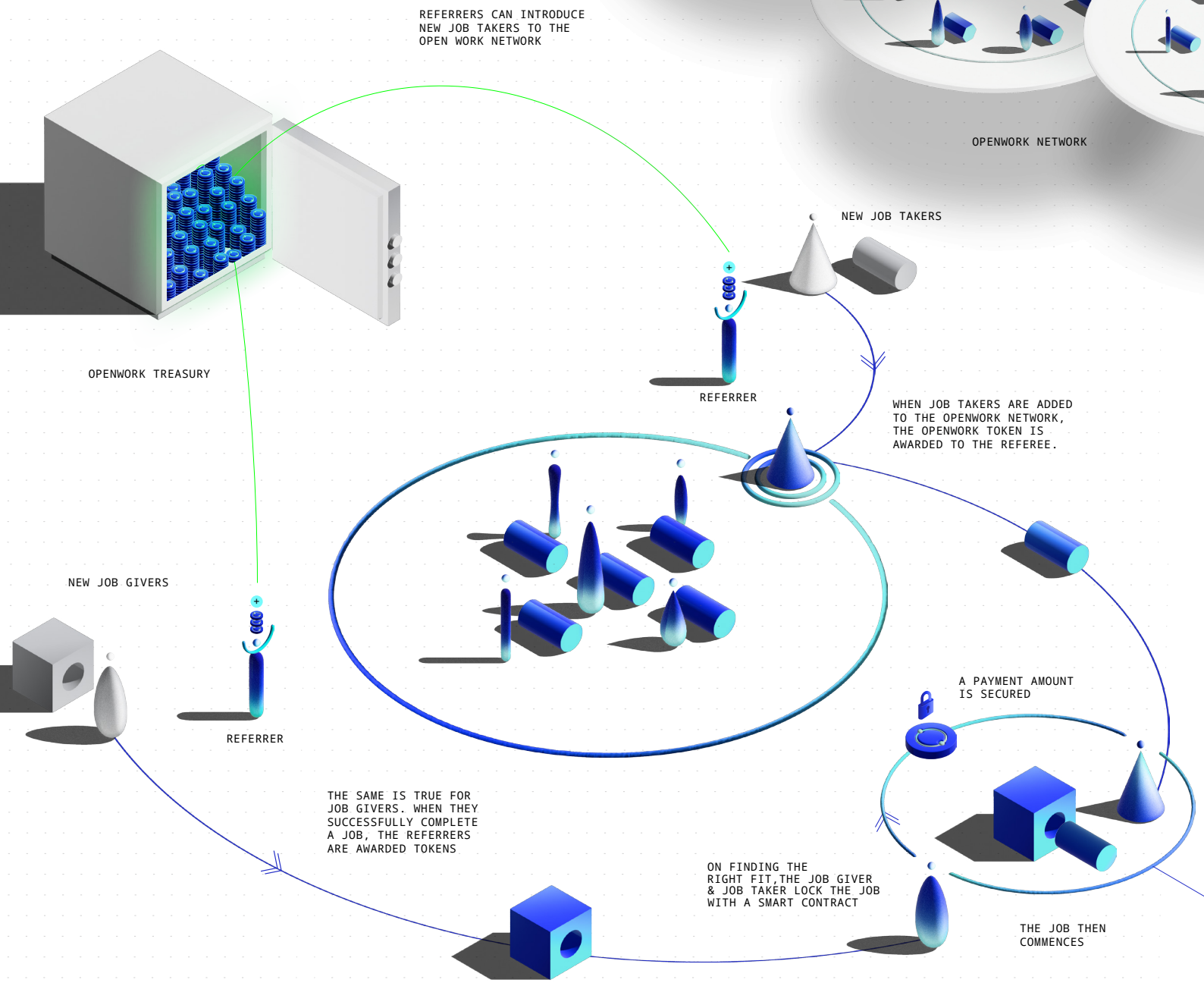
Fig, 4



Tokenomics that incentivise growth

Centralised job networks depend on centralised growth efforts to acquire both job givers and job doers.

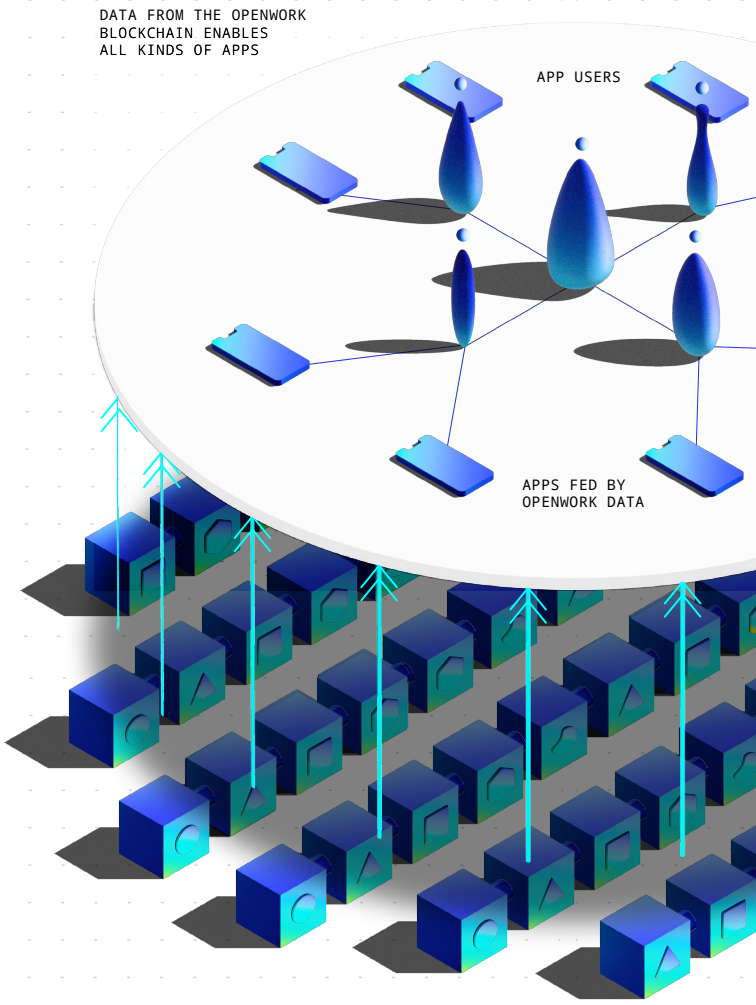
OpenWork's network will be self-growing as all public keys are incentivised to attract more job givers and job takers to the network by being awarded the OpenWork token if they do so as "referrers". When job takers are added to the network and either pass the vetting process of the skill oracle for a particular skill, or execute a job on the OpenWork network, the OpenWork token is awarded to the referee. When job givers successfully complete a job, the same occurs.



The OpenWork Ecosystem

Layer 2 for Decentralised Work

The OpenWork ecosystem would enable all kinds of applications that could be built on top of the OpenWork blockchain.



Dapps that create Programmable Work - The power of Smart Contracts using Skill Oracles

Athena's skill oracle can be used in more than just dispute resolution. Athena can make work contracts with humans programmable through smart contracts. All sorts of smart contracts can be entered into using skill oracles. The skill oracle can give any truthful answer pertaining to a skill and therefore allows contracts that execute based on the answer to self-execute. The following applications can use the skill oracle:

1. The job giver has the ability to give performance based payment for a job at hand. For example it can outline 2 objectives of a job. If both are met as per the skill oracle, the full payment is given. If one is met, half. Sticking to our translation example, the contract can trigger a 75% payment if there are more than 10 objective grammar mistakes and 100% payment if there are none. *(Fig. 5)*

Fig. 5 - Performance based Payments

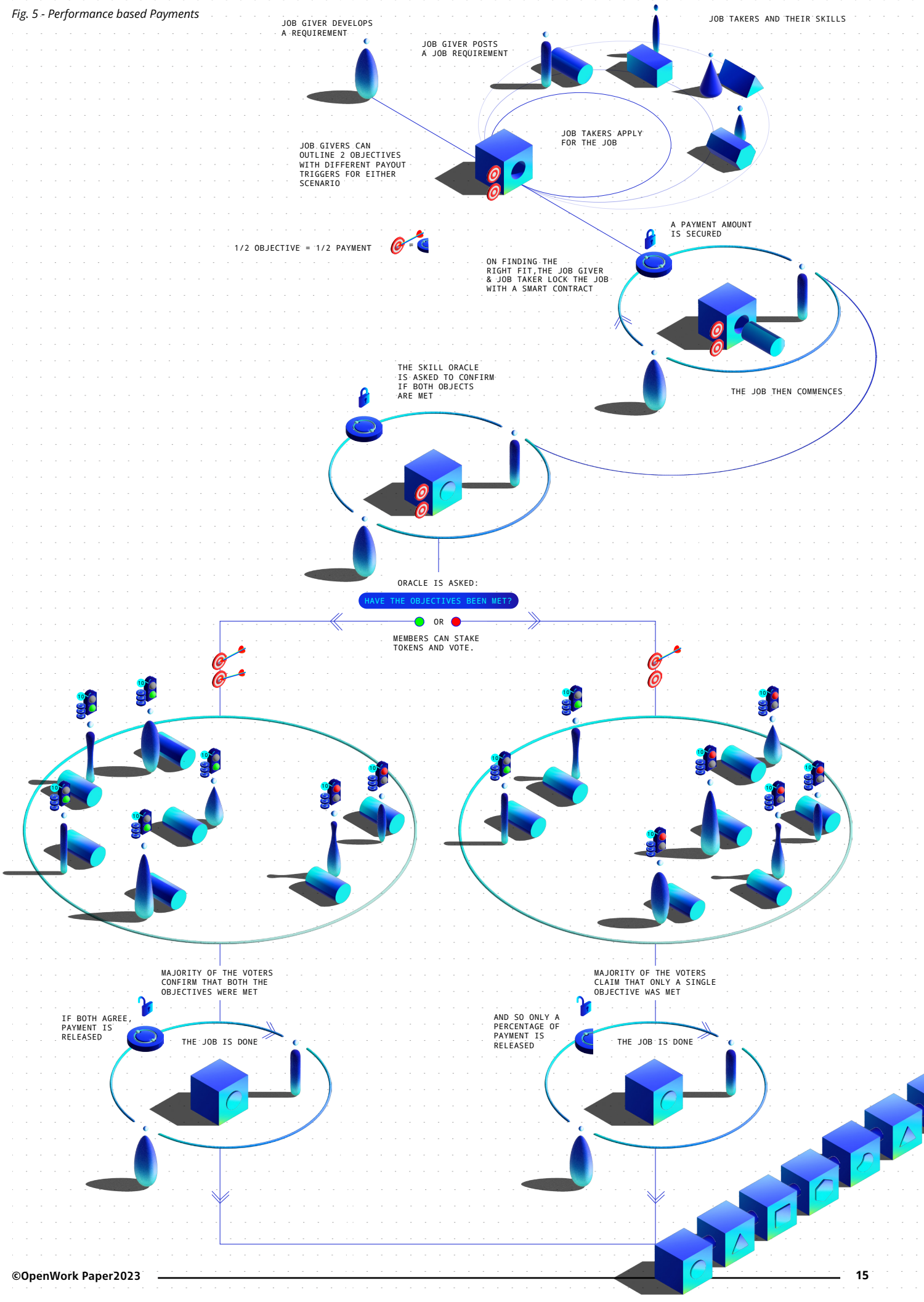
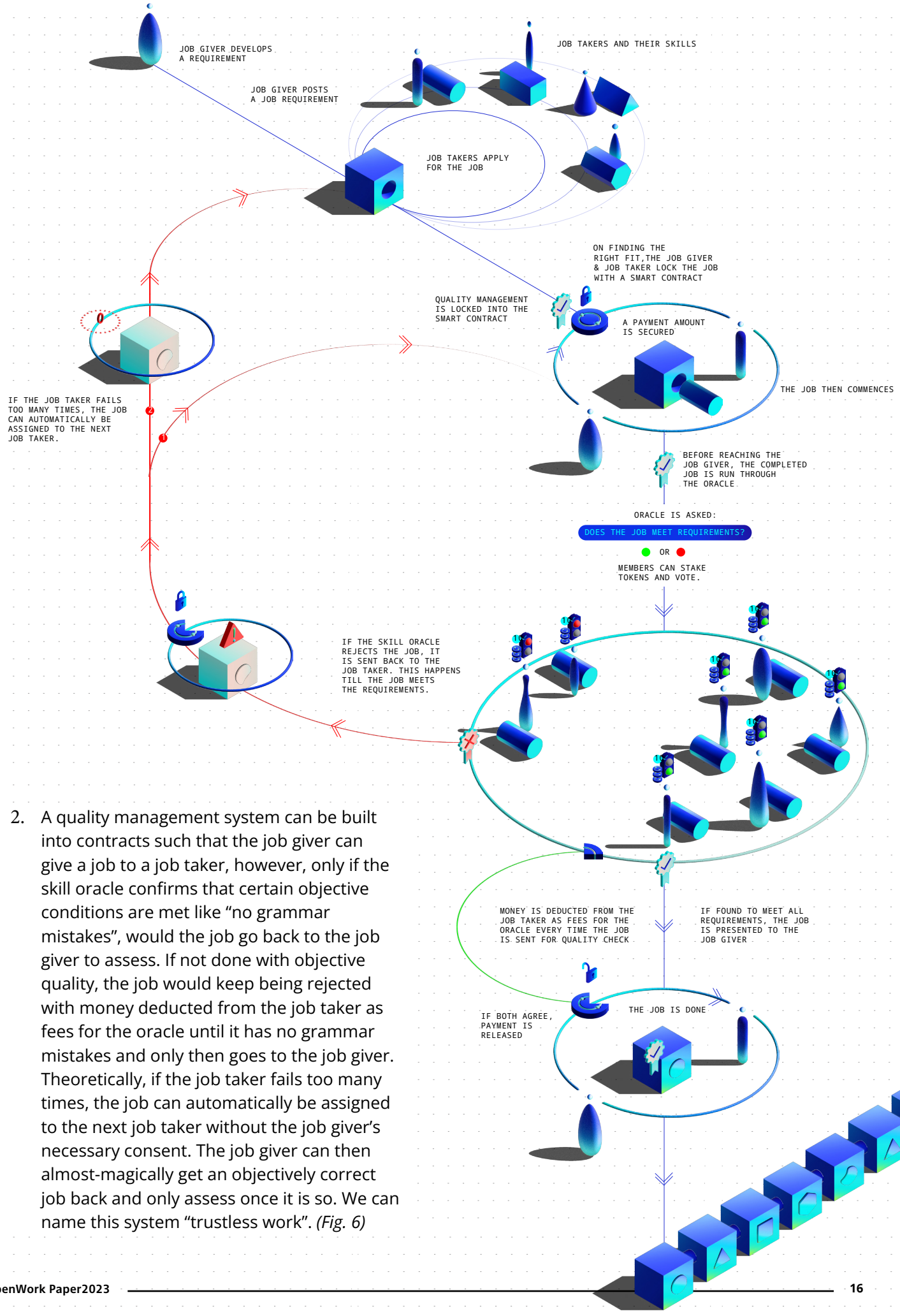


Fig. 6 - Trustless Work



2. A quality management system can be built into contracts such that the job giver can give a job to a job taker, however, only if the skill oracle confirms that certain objective conditions are met like “no grammar mistakes”, would the job go back to the job giver to assess. If not done with objective quality, the job would keep being rejected with money deducted from the job taker as fees for the oracle until it has no grammar mistakes and only then goes to the job giver. Theoretically, if the job taker fails too many times, the job can automatically be assigned to the next job taker without the job giver’s necessary consent. The job giver can then almost-magically get an objectively correct job back and only assess once it is so. We can name this system “trustless work”. *(Fig. 6)*

3. An interesting example that has practical application for skill oracles even outside a job giver and job doer situation are contracts that use lawyers. For example, a contract can say “If you legally own this painting as per the art lawyer skill oracle’s assessment, then I buy it from you.” Depending on the answer given, the transaction goes through or not.

Creating Complex Organisation Structures based on Skill Chains

Through the system of trustless work, the job giver can be assured to a high degree of certainty that the job given to job doer to complete is completed to a standard when it comes to objective work.

A dependable outcome of work given without the need for human intervention outside the skill oracle can result in an autonomous chain of jobs that work one after the other to complete larger complex jobs all through the smart contract. This is what we can call a skill-chain.

For example, the job can extend like a chain to execute the following - consider a web page that needs to be translated from English to French and then updated. First the job would go to the job taker key who performs English to French translation through the trustless work mechanism such that the job only gets passed once approved by the oracle. Then the job is passed on to a web developer who then updates the website. The job giver then only assesses the job once complete by both parties with objective quality met. *(Fig. 7 - Skill Chains)*

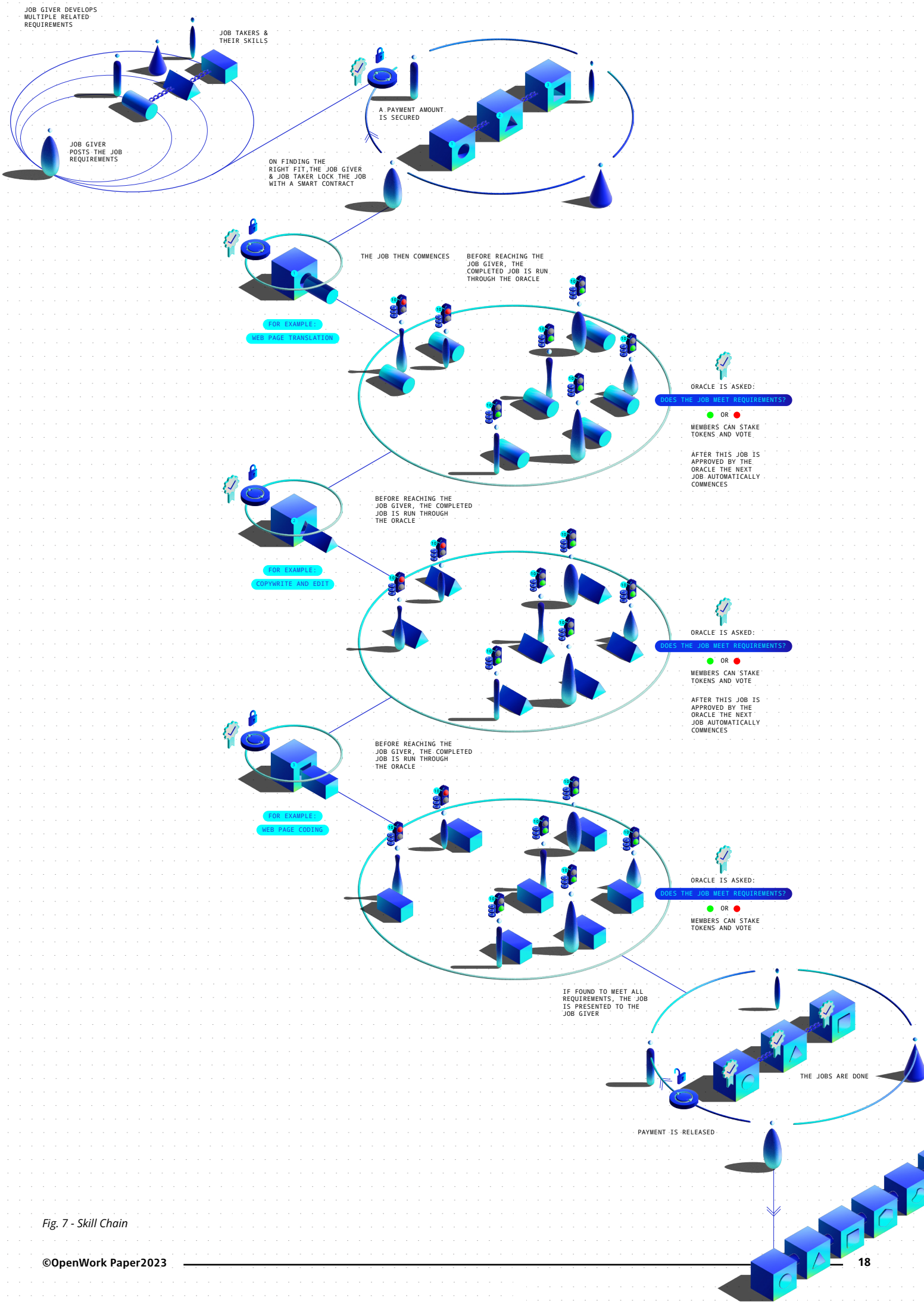
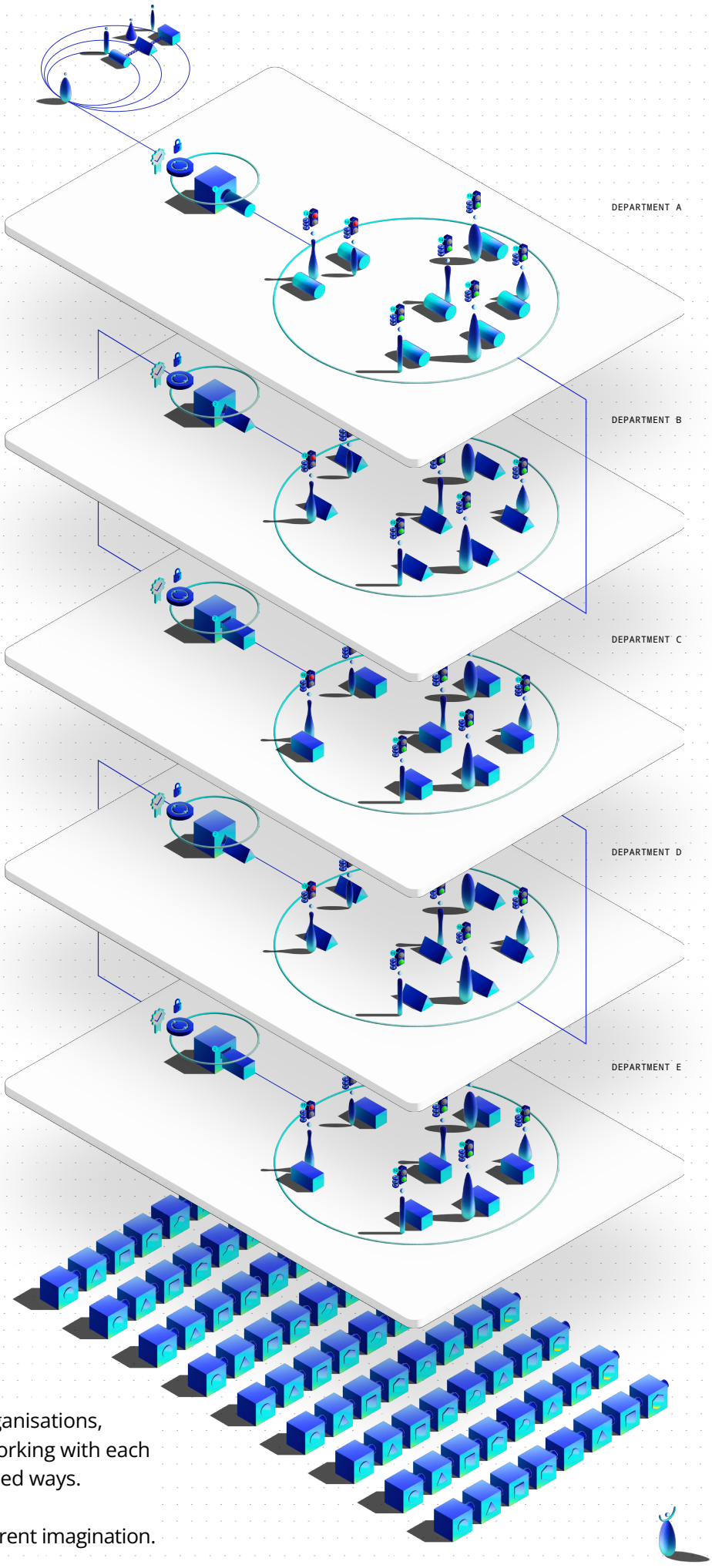


Fig. 7 - Skill Chain



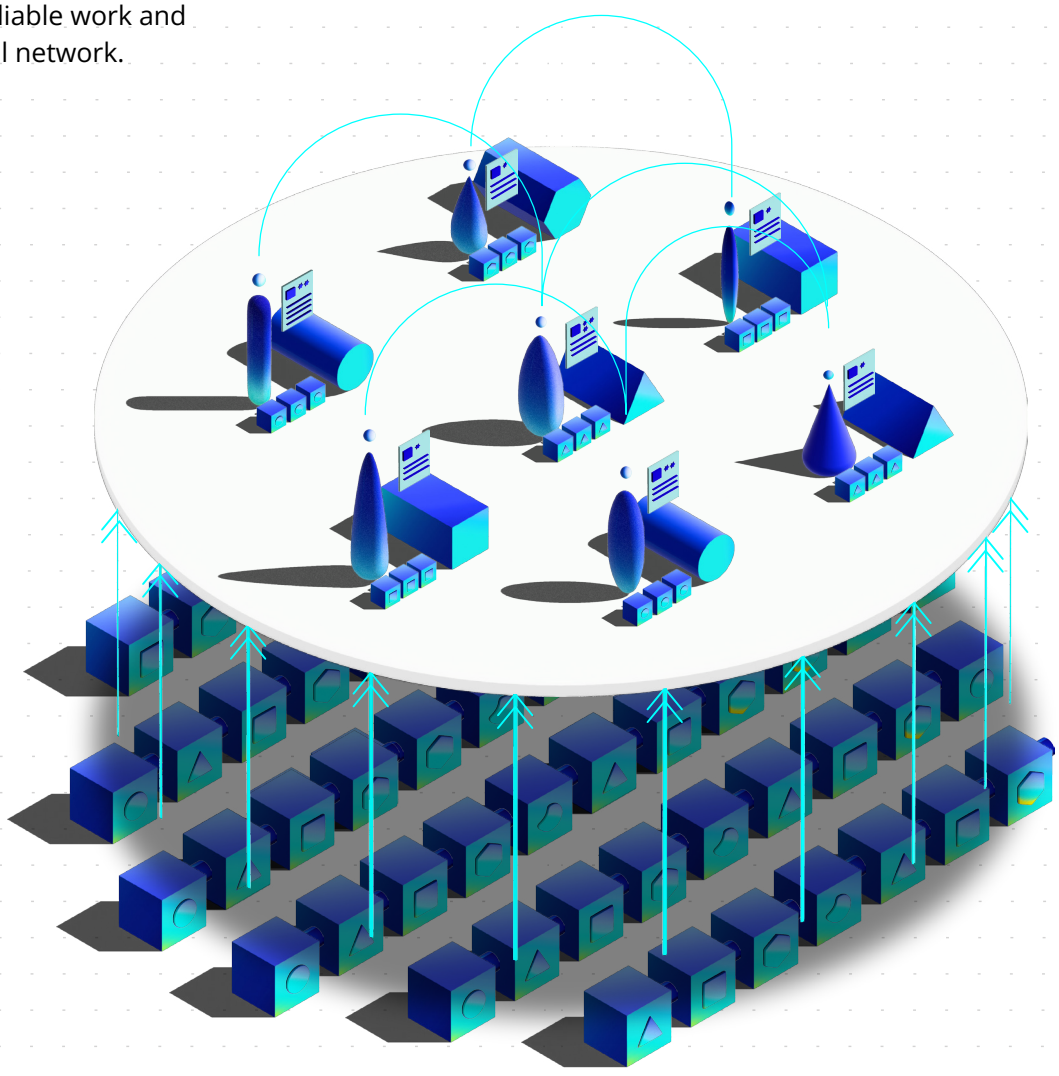
This can theoretically result in entire organisations, which are largely members with skills working with each other, running in completely decentralised ways.

The use cases extend far beyond our current imagination.

Other concepts that can be built off OpenWork

Decentralised LinkedIn

A social network built on the OpenWork blockchain as a base so that keys on the network have a record of reliable work and can connect through a social network.

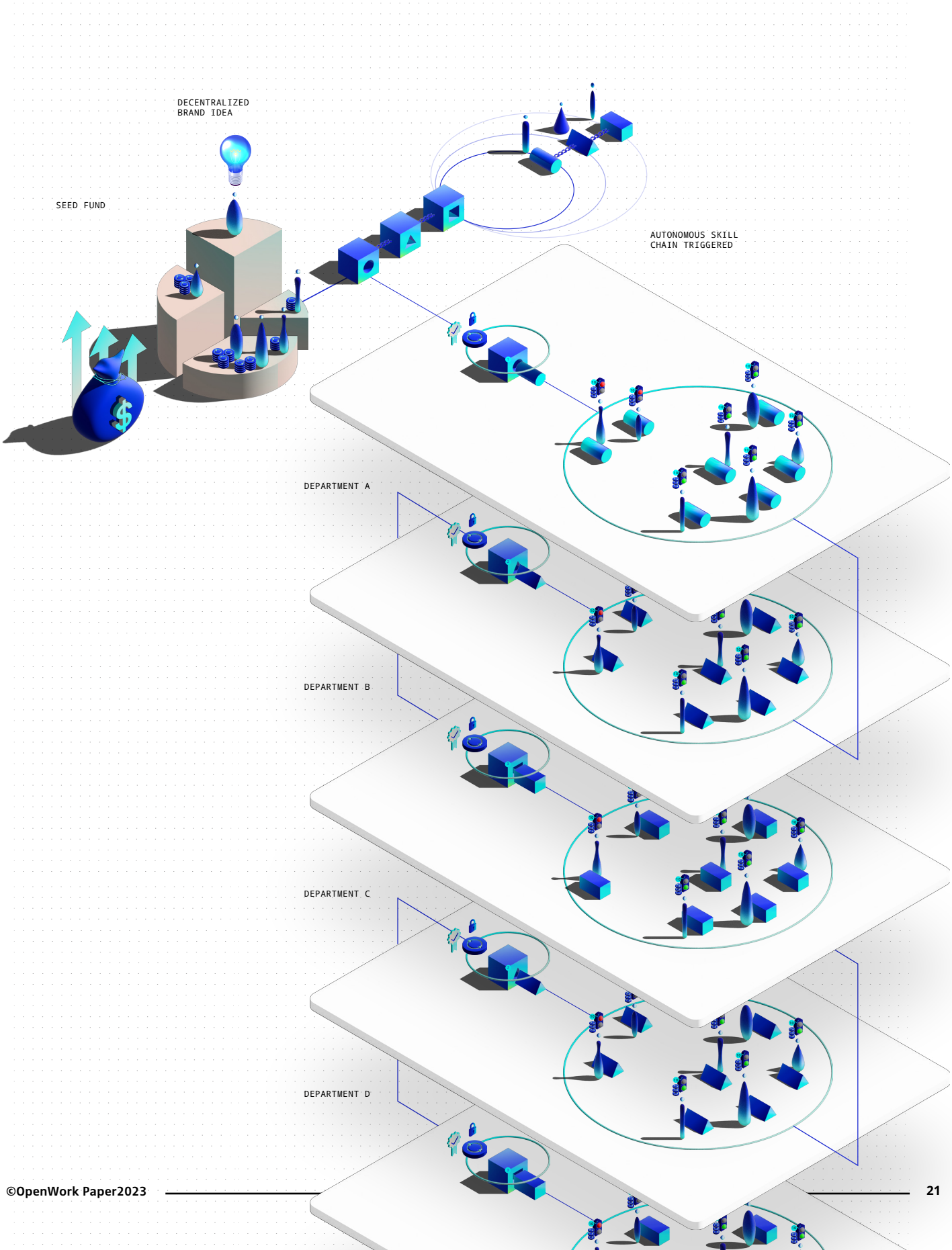


DATA FROM OPENWORK
CAN POWER A SOCIAL NETWORK.

THIS KEEPS RECORDS OF ALL
TRANSACTIONS AND REVIEWS
PROVIDING RELIABLE WORK
AND CONNECTIONS.

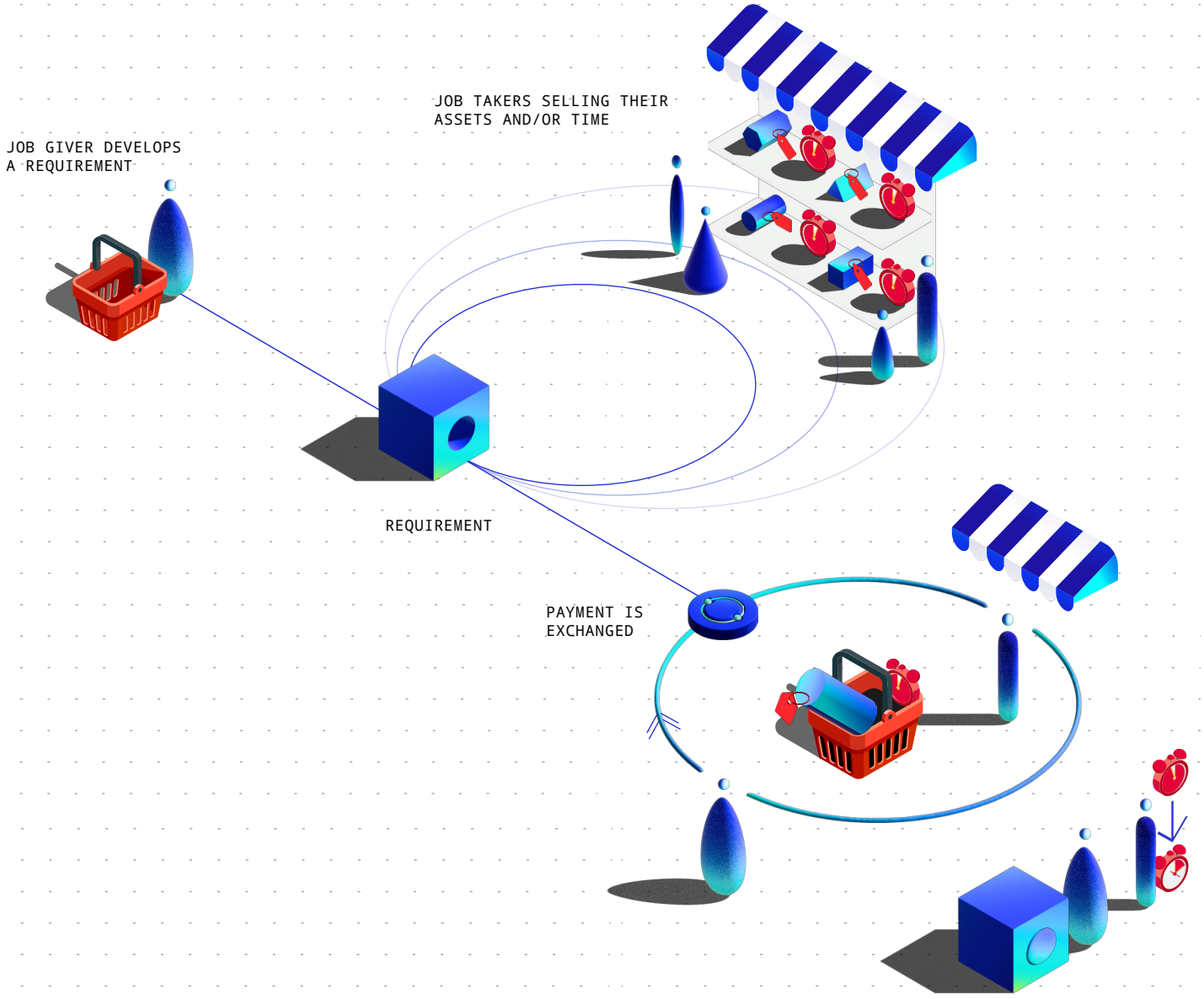
Drands

Creating decentralised brands or “drands” through a Kickstarter - like process to ensure minimum funding that would autonomously work through a skill-chain, to run an entire real-world decentralised business.



Expertise as Assets

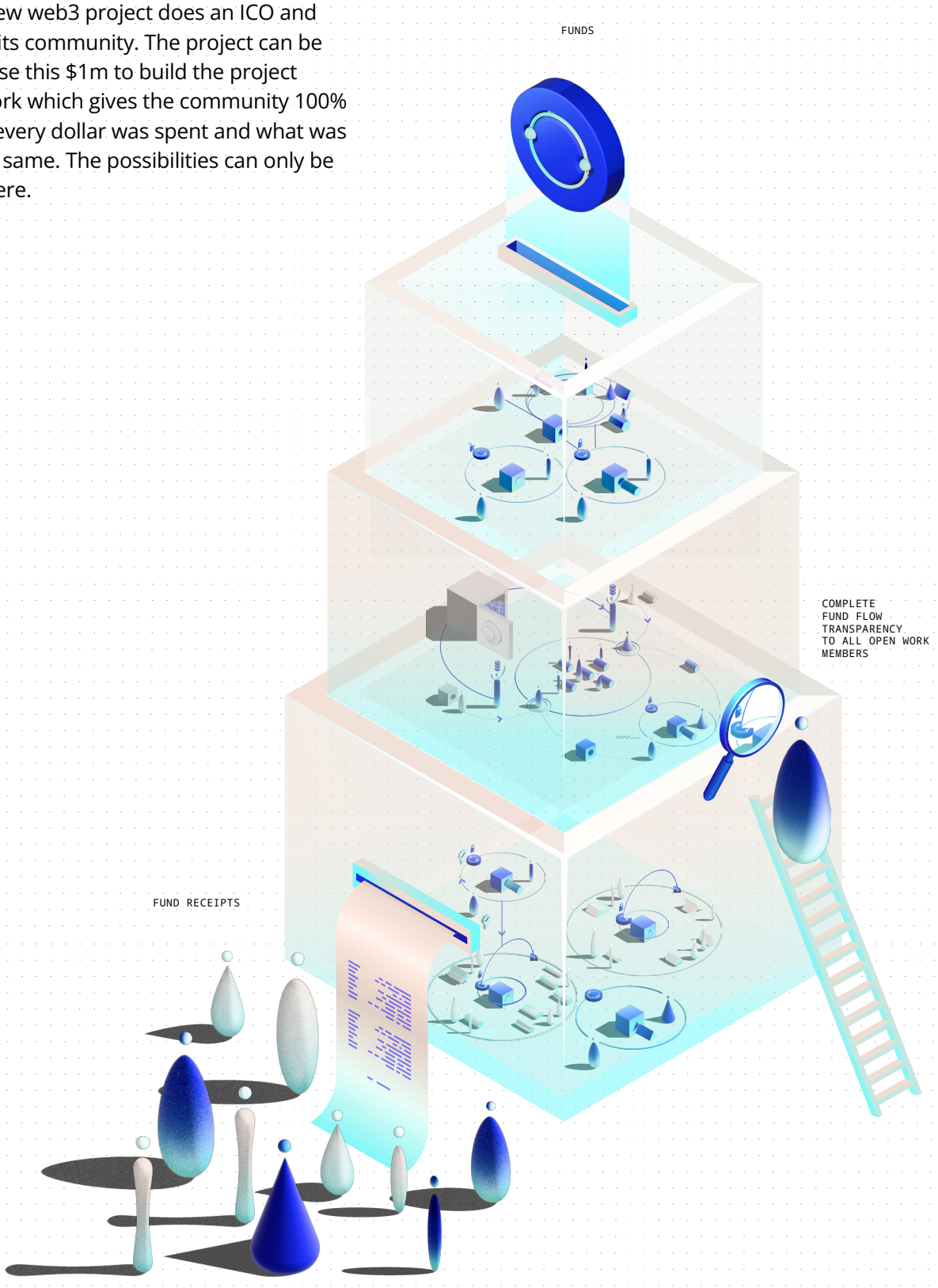
Tokens through which job doers can sell their time, or services in advance as an asset with accountability brought by the OpenWork Blockchain.



Fund Transparency

An application through which DAOs can run with more accountability with regards to how all funds are used. OpenWork can become the gold standard for transparency in how funds are used at a web3 organisation to help build the project as the fund flow in exchange for work would be completely transparent.

For example, a new web3 project does an ICO and raises $1m from its community. The project can be allowed to only use this $1m to build the project through OpenWork which gives the community 100% visibility on how every dollar was spent and what was built through the same. The possibilities can only be imagined from here.



More Context and Notes

The Context on Smart Contracts:

Smart Contracts are self executing contracts or programs that exist on a blockchain that run when predetermined conditions are met through a change in the data received.

If this data is to be gotten on-chain or through the same blockchain it exists on, it's self-evident that the types of contracts that can be used are limited to the data that exists on the chain. For example, a smart contract can exist on ethereum's blockchain that performs the logic of "if the number of ethereum holders is 1 million, then burn 100 ethereum." Here all the data is gotten from the ethereum chain itself.

If smart contracts need to interact with data outside of the chain they use oracles. Oracles are **entities that connect blockchains to external systems**, thereby enabling smart contracts to execute based upon inputs and outputs from the real world.

Even though theoretically oracles can bring all kinds of real world data to smart contracts, their current applications are mainly limited to finance and api related data. This is the reason why smart contracts are barely replacing regular contracts being used in the real world- the oracles just aren't in place to give the right data to enable them to run.

The most basic contract isn't smart

One of the most common "contracts" used in the world, and probably from the beginning of time, is one between a contractor and a service provider. In simpler words, a person paying another person to perform a service. This type of contract is not yet a smart contract.

If we break down what it entails we can demonstrate how it can turn into a smart contract. These are the key steps needed for this type of contract to be completed successfully:

- The contractor has a job requirement and the service provider has the skill(s) to execute the same.
- The two parties agree that the contractor must pay the service provider a certain amount if the job is performed by the service provider.
- Once the contractor claims to have done the job, the contractor assesses if job is complete or not.
- If complete, the service provider is paid.

In the web2 era, centralised start ups like Fiverr and UpWork were created and adopted as trusted third parties to solve problems in these steps.

Platforms like these connect contractors with service providers, hold funds in a centralised account and release them once the job is complete. In case of a dispute, the platform acts as a centralised authority on what happens next.

It's self-evident that, "trustless" work has not been achieved. A trustless system eliminates the need for participants to trust each other for the system to work.

Trustlessness means no central figure (like a bank) has authority or control over the said system.

What is Trustless Work?

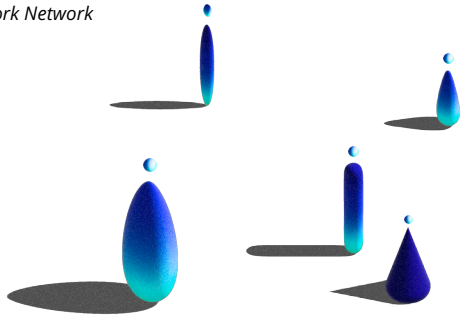
Our solution creates Trustless Work, which entails a contractor giving a service provider some work in exchange for an agreed compensation if the work is complete correctly, each without needing to trust the other nor a centralised third party.

In order to achieve a state of trustless work, the following is required-

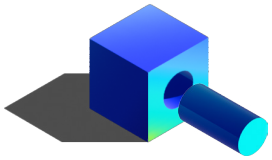
- Job givers can find the appropriately skilled talent through a platform that is owned and governed by a decentralised community.
- Job givers have the ability to hand over a job to a contractor and have work delivered back with expected minimum quality, paying if the job meets the same and not if it does not, without a trusted third party.

Openwork Icon Key

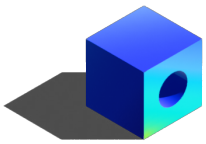
Members of the Openwork Network



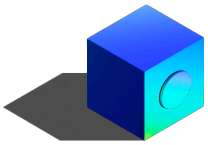
Meeting the Job Requirements



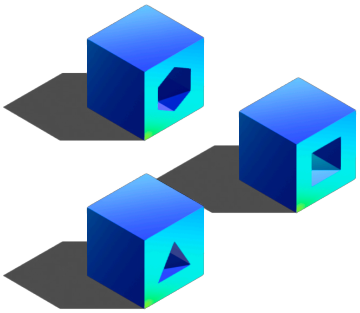
Job Requirement



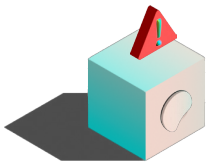
Job Done



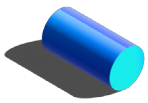
Unique Job Requirements



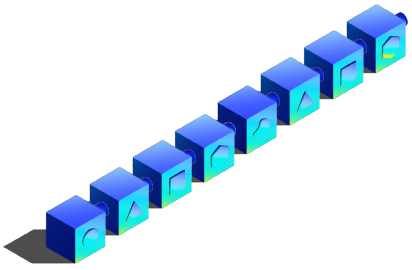
Job Not Done



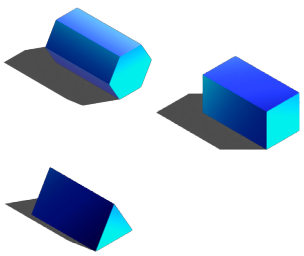
Skill



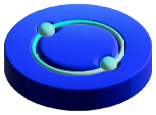
Openwork Blockchain



Unique Skills

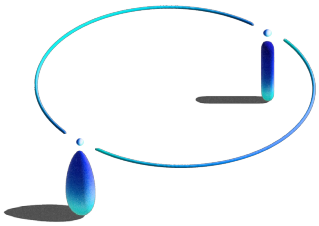


Openwork Token

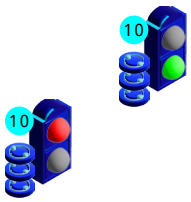


OpenWork Icon Key

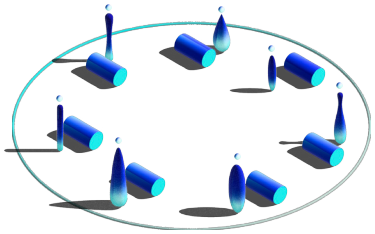
Smart Contract



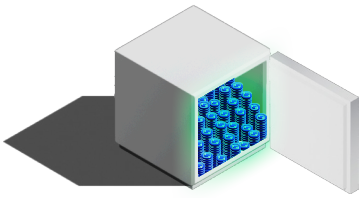
Oracle Voting Tools



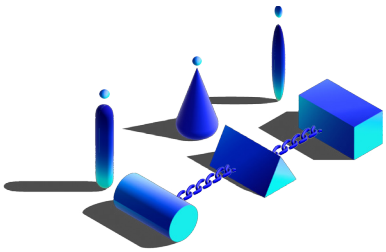
Skill Oracle



Openwork Treasury

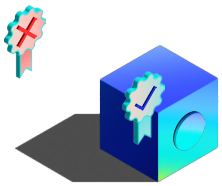


Skill Chains

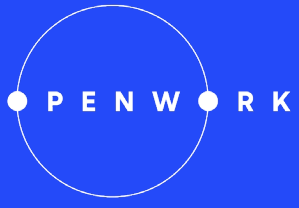


Job Objectives

Quality Markers



Non Openwork Member



OpenWork Paper 2023
By Armand Poonawala