**Ocean: A Decentralized Data Exchange Protocol**

Ocean Protocol Foundation[[1]](#footnote-0)

with BigchainDB GmbH[[2]](#footnote-1) and Dex[[3]](#footnote-2)

DRAFT Sept 15, 2017. **DO NOT DISTRIBUTE**

**Abstract**

This paper presents a decentralized data marketplace protocol and network called Ocean, on which data marketplaces can be built.

The world has recognized the value of data, but it’s been very difficult to establish a price for the data (especially non-fungible data) while reconciling privacy concerns. Many enterprises have tremendous amounts of data, but have difficulty exploiting it. Conversely, many startups have deep expertise in artificial intelligence (AI), but lack the data to make their AI models perform. To address this problem, data marketplaces have emerged, but they are silos themselves.

Ocean is a protocol and network to ease building of decentralized data marketplaces. Ocean handles storing of the metadata (who owns what), links to the data itself, and data IP licensing information. On top of Ocean there can be thousands of data marketplaces and exchanges, all accessing the same data. Each marketplace acts as the last mile in connecting buyers and sellers. Crucially, pricing offers are at the shared Ocean layer, not at the marketplace layer, to help liquidity and avoid lock-in of a dataset offer to a given marketplace. Ocean incentivizes uploading data, especially for data commons. Each dataset is controlled by the respective rights holder, with privacy measures. Ocean provides programmable market mechanics, making fair yet flexible pricing easy. Ocean is designed for industrial-scale usage.

**Contents**

[**1. Introduction**](#_e14lkjabesm5) **3**

[**2. Overview of Solution**](#_qn0u5c8oibis) **3**

[2.1. Token Design](#_n9yo5283wndh) 3

[2.2. System Architecture](#_kxoy5c3qoca) 3

[2.3. Protocol Sketch](#_woze5ltdio2v) 4

[**3. Approach to Token Design**](#_d1q0pyvfs9y1) **5**

[**4. Use Cases**](#_olfgbdtn3evj) **6**

[4.1. Autonomous Vehicles](#_fpmnrcn9cg5j) 6

[4.2. Medical Research](#_vyu3bsmr63oz) 6

[4.3. Global Data Commons](#_hrlgd8b91d0g) 6

[4.4. Other](#_xe5yn2jsyryx) 6

[**5. Pricing**](#_tewqwqa7hgfr) **6**

[**6. Stakeholders**](#_ycrhs4dq95e0) **7**

[**7. Key Goals**](#_4dbalbb4i6zl) **9**

[**8. Token Design**](#_5ezmzpj91iic) **9**

[8.1. Candidate Designs](#_duficxh2s11g) 9

[8.2. Registry on Data](#_jg7h4iwr7czf) 9

[8.3. Free Pricing](#_58d6fpvofwey) 9

[8.4. Non-Free Pricing](#_wl74obsa99wd) 10

[8.5. Short Term Governance (Keepers)](#_stwj86jvufjn) 10

[8.6. Medium Term Governance (Fixing Bugs)](#_p1ow9uwzxtdu) 11

[8.7. Long Term Governance (Protocol Updates)](#_fpp5wzb0x9rs) 11

[8.8. Network value](#_kstalper49p5) 12

[**9. Technical Dates**](#_j86o2ab961zp) **12**

[9.1. Dates leading to Ocean](#_xluzaze9kti2) 12

[9.2. Software in Ocean Ecosystem](#_45v38ta5vm0p) 14

[9.3. Ocean Software Releases](#_y3jsdt29jy74) 14

[**10. Conclusion**](#_vfd7hdpyh926) **15**

[**11. Acknowledgements**](#_7it7zjqoqj71) **15**

[**12. References**](#_agtlfo7ucebs) **16**

[**13. Appendix: Ocean in the Decentralized Stack**](#_8h6sio8glqxs) **16**

[**14. Appendix: Target Specifications**](#_yukzzbhfoc3t) **16**

[**15. Appendix: Realization of Target Specifications**](#_kt65pjxov1w0) **17**

[**16. Appendix: Detailed Protocol Sketches**](#_tevxsby8sw76) **24**

[16.1. Self-driving cars with fixed price](#_s2qvcyulkn1z) 25

[16.2. Parkinson’s with fixed price and on-premise compute](#_7knc1pj22s9d) 25

[**17. Appendix: Token Design Goals, Questions, and Tools**](#_3lvi6nkx0idq) **25**

[17.1. Buy/Sell/Incent Data](#_zewff7r5jlra) 25

[17.2. Economics](#_kurix9idiud) 27

[17.3. Towards Good Acting via Staking, Id, Reputation](#_m8x1zgo182n3) 27

[17.4. Short Time Scale Governance (Keepers)](#_8g8jnwijr9az) 30

[17.5. Medium Time Scale Governance (for Bugs)](#_br1aatggbj3g) 32

[17.6. Long Time Scale Governance (for Protocol Updates)](#_iexmxvoobdc9) 32

[**18. Appendix: Candidate Token Designs**](#_q9r187ywys9c) **33**

[18.1. Candidate Design 1: Incentivize on Sale Price](#_5i02z31nrdcp) 33

[18.2. Candidate Design 2: Curation Market Pricing on Commons Data](#_5xqghb9jv9n2) 35

[18.3. Candidate Design 3: Free Pricing on Commons Data](#_w2vbnd3gvba9) 36

[18.4. Comparison of Candidate Designs](#_xevtijkhv2ng) 37

[**19. Appendix: Token Design Full Checklist**](#_93hxh049vsh9) **38**

[19.1. Main goals](#_31mgnlp192ax) 38

[19.2. Buy/Sell/Incent Data](#_ew3awxlm1wjk) 38

[19.3. Token Economics](#_aaket0mwpaxs) 39

[19.4. Towards Good Acting via Staking, Id, Reputation](#_oha332vbt4oy) 39

[19.5. Short Time Scale Governance](#_ja8ys3vvgjx8) 39

[19.6. Medium Time Scale Governance](#_rfe07cj79izj) 39

[19.7. Long Time Scale Governance](#_h2mb33kdr2z7) 40

[19.8. Software / Complexity](#_a3niqff4l682) 40

[19.9. Usability for Buyer / Seller / Referrer](#_d850u6ik85ox) 40

[19.10. Attacks](#_rygioyv1ka2b) 40

[19.11. External Checklist](#_mde1h3wkj7ak) 41

# 1. Introduction

The cover story of the May 6, 2017 edition of *The Economist* began, “The world’s most valuable resource is no longer oil, but data.” [[Economist1](#xpj55h2jpd9j)] Why? The technology to gather, store and use massive amounts of data has only become affordable recently. The amount of data being gathered keeps growing. Data storage and computing keep getting cheaper. And the tools to manage and make use of data have improved. Data, and the ability to use it, have become differentiating factors in business.

One way to use data is to build models of the world, so as to make improved predictions of the future, or to better-understand the past. Artificial intelligence (AI) practitioners use data to “train” models. Recently, they found that they could improve model accuracy without spending time developing clever new algorithms: all they need is *more data*. That discovery is what made modern computer vision and natural-language understanding (like Siri) accurate enough to be useful.

However, all is not well. Not everyone has data. Without data, the model is not accurate. With no accuracy, the model is unusable. The winners so far have been companies with vast data resources *and* internal expertise, like Google and Facebook. Others have been less fortunate, having one but not the other. Some startups have amazing algorithms but are starving for data. Conversely, some enterprises are drowning in data but have comparatively less expertise. Few companies have both the engine and the oil.

This paper presents a decentralized data marketplace protocol and network called **Ocean**, on which data marketplaces can be built. In Ocean, the network control is decentralized, but the data & privacy control is in users’ hands. The market mechanics are fair yet flexible; in fact they are customizable by higher level marketplaces and exchanges. Ocean is designed for industrial-scale usage. Ocean’s token model incentivizes growing the data in Ocean, especially the data commons.

# 2. Overview of Solution

This section contains an overview of the solution. Later sections elaborate on various parts.

## 2.1. Token Design

Ocean maintains a registry of data that is accredited as non-fraudulent by Ocean token holders. Anyone can challenge the poster’s claim at any point (with stake); you lose stake if you’re wrong.

Different data pricing schemes sit on top. Free data has a reputation set by Curation market staking, which implicitly incentivizes referrals. Non-free data must be staked. There can be fixed pricing, auction pricing, royalties, or something programmed by developers. All staking and purchase actions are made in Ocean tokens.

Ocean rewards keepers for reliably serving up data, with rewards proportional to the data’s stake (free: reputation; non-free: price).

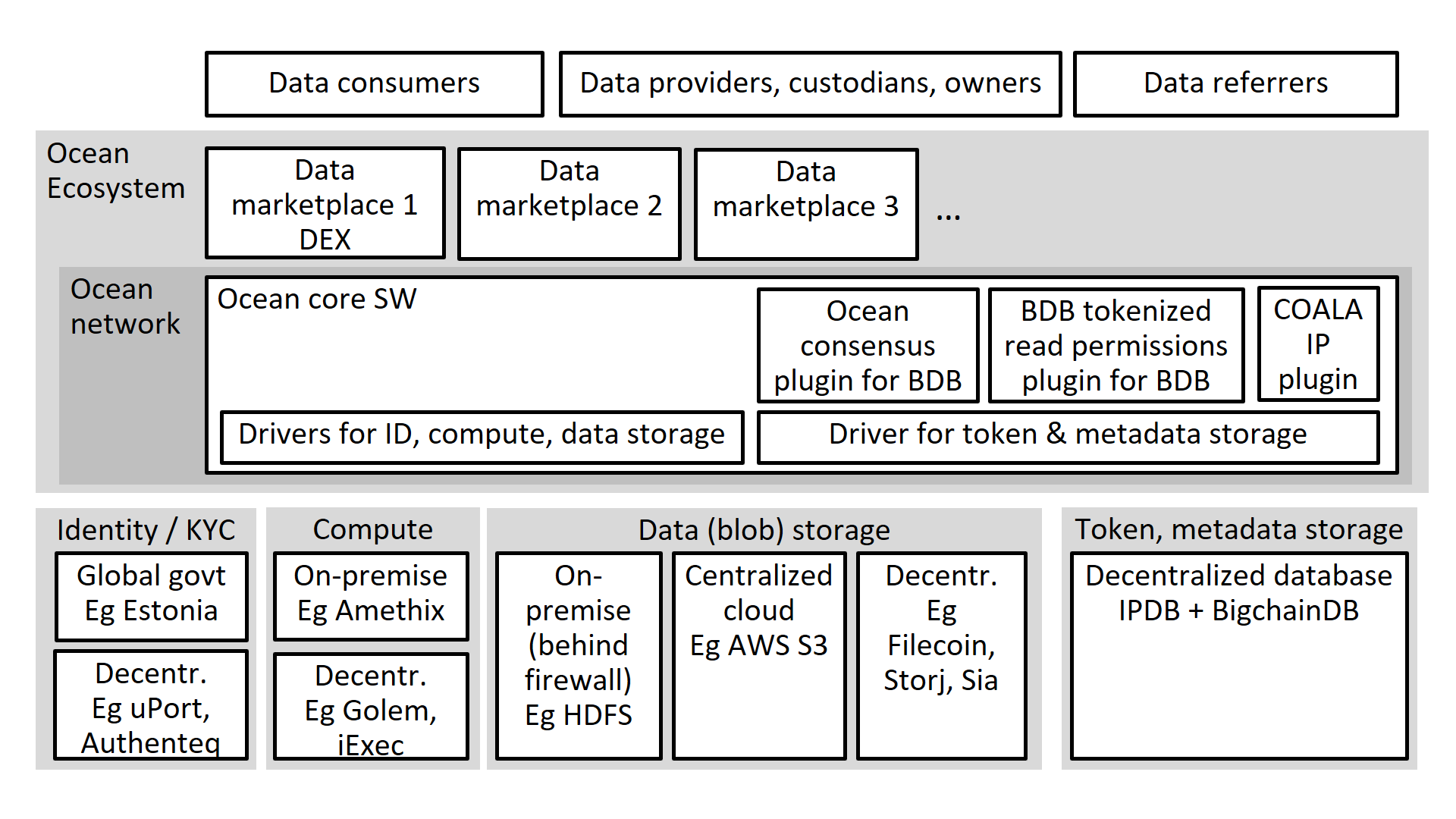
On-chain governance based on token holdings and reputation is grandfathered in over time, from the core dev team. Ocean foundation governance is separate.

## 2.2. System Architecture

The figure below shows the overall architecture. At the top are the users: data consumers, providers, custodians, owners, and external referrers. They can always interface directly with Ocean network if they like; however for convenience with graphical user interfaces and jurisdiction-specific data governance regulations, they will usually interface through third-party data marketplaces like DEX.

Marketplaces and exchanges are typically how providers and consumers interact with Ocean network, for last-mile convenience. To catalyze marketplaces, we are building a reference data marketplace (DEX) having permissive open source licenses to allow other startups to use the code as a starting point.

The Ocean network itself is composed of a set of Ocean keeper nodes, each running Ocean core software. Keepers act as watchdogs for correct settlement: ensuring that the data is sent to the buyer, and that the tokens are sent to the seller. This is their “work”.



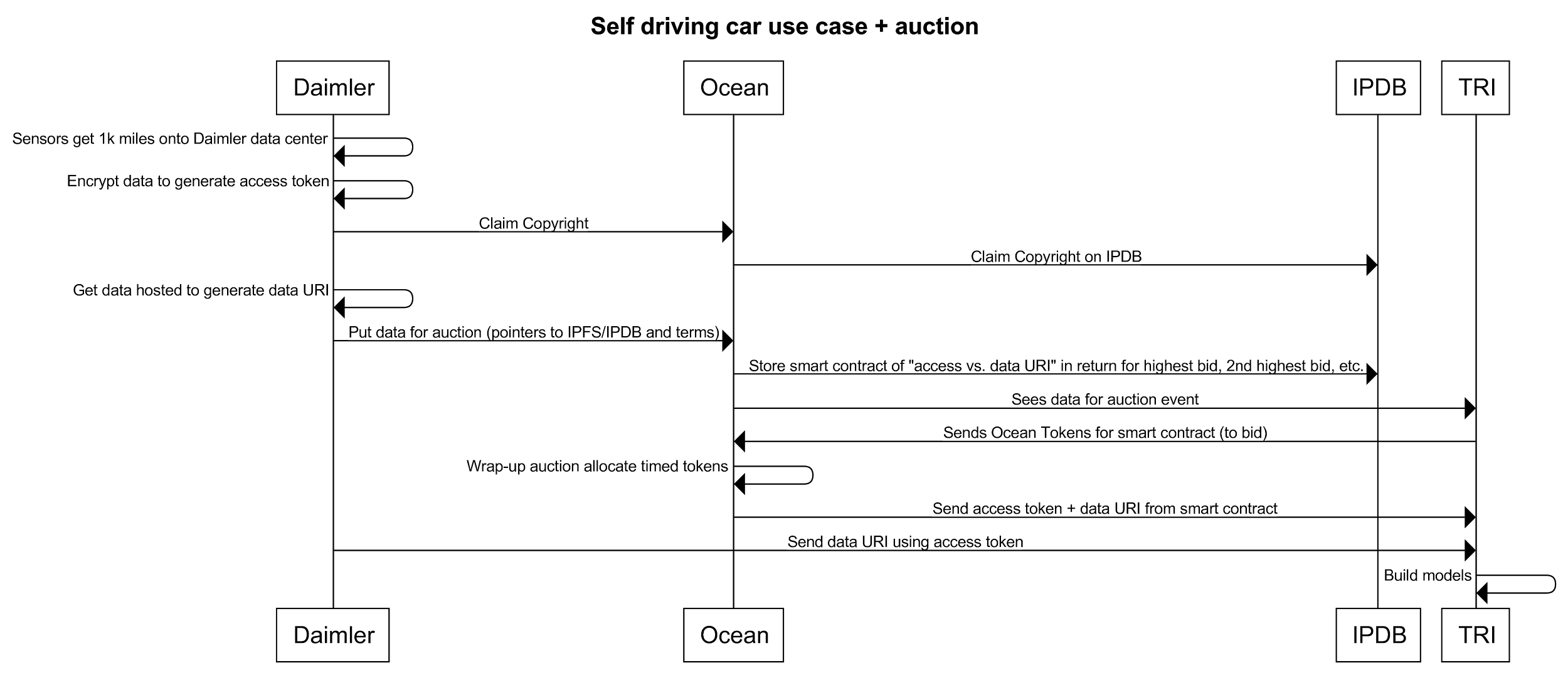
Ocean is a Delegated Proof of Stake (DPOS) system, which nicely balances performance with decentralization, and gives optionality for governance. There are two types of keepers:

* **Active keepers.** These do validation (i.e. vote on whether a tx is valid), issue block rewards, computation as needed, etc. There will be on the order of 100 of these; more just adds latency and uses higher bandwidth.
* **Read-only keepers.** These simply mirror the data, and are on standby to become active keepers.

Keepers use third-party software and networks for identity, privacy-preserving compute, data (blob) storage, and token & metadata storage. Keepers leverage BigchainDB heavily: they peer with each other using an Ocean-specific consensus plugin, they use another plugin for tokenized read permissions (key to data privacy), and a third plugin for COALA IP (giving flexibility and legal teeth to IP licensing).

## 2.3. Protocol Sketch

Here, we show the sequence of steps for actors interacting, for the use case of self-driving cars with auction-style pricing. Daimler supplies the data, TRI is the buyer, and in between are Ocean network and IPDB (for metadata). Ocean must support all the input and output signals coming to it; IPDB must support all the input and output signals coming to it. This is merely a sketch. A later section fleshes out protocols further, for example the use case of fixed-price on Parkinson’s accounting for privacy requirements.



This concludes the high-level overview of Ocean. We now start to elaborate on Ocean’s design methodology, design options, and design itself.

Protocol details can be found in the Appendices.

# 3. Approach to Token Design

**Steps.** Ocean is a tokenized ecosystem. Token design is about designing incentives for stakeholders to grow the value of the network. Here are the steps we follow:

1. **Use Cases.** Identify key use cases to guide us.
2. **Pricing.** Identify key possible pricing schemes to support. (Specific for a marketplace protocol.)
3. **Stakeholders.** Identify the stakeholders, and their “gives” and “gets”. Choose which stakeholders we want to incentivize in the system itself.
4. **Goals.** Choose the main goal. In our case, it’s getting data into the system. Document questions and tools for that goal. Then, select other related goals, questions, and tools. Identify attack vectors.
5. **Checklist.** Distill the goals into a detailed checklist of questions.
6. **Candidate designs.** Propose candidate designs. For each, compare it against the checklist (especially key goals).
7. **Iterate to final.** Propose a final design. Get feedback from others, and repeat previous steps as needed.

Subsequent sections will cover these steps.

**Optimization framing.** We can frame token design as trying to design a mathematical optimization problem that the live network is trying to solve. The first few steps are about identifying the optimizer’s goals, in terms of objectives (things to maximize and minimize) and constraints (which must be met); and the design space (what can actors in the system do, to improve against the goals).

In particular, the network is a lot like an evolutionary algorithm, where the coordinating agent (optimizer) has little direct control over the actors; actors can take actions like mutating and crossing over; all the optimizer can do is keep the good and discard the bad. But if this gets repeated over time, i.e. evolved, then the system will arrive at a state with many good actors.

# 4. Use Cases

These use cases guide our design.

## 4.1. Autonomous Vehicles

We (at BigchainDB) started working with Toyota Research Institute (TRI) in early 2017 for self-driving cars. The accident rate of self-driving cars reduces with higher model accuracy. Models get more accurate with more data. TRI had calculated that it needed 500 billion to 1 trillion miles driven in order to get models that are sufficiently accurate for production deployment of self-driving cars (and this is on top of orders of magnitude more miles driven in world simulators). TRI saw that each automaker faced similar challenges; and that it would be prohibitively expensive for each automaker to generate that much data on its own. Why not pool the data, via a data marketplace?

Driving data illustrates how not all data is fungible: a mile driven in a blizzard in Canada is worth more than a mile driven on an empty, sunny desert highway. But one mile in the blizzard is fungible with other miles in blizzards. The system must account for both fungible and non-fungible data.

## 4.2. Medical Research

We (at DEX) are working with hospitals in Munich, Singapore, and elsewhere on a Parkinson’s research study in the Connected Life initiative. The goal is to build models based on patient data spanning these hospitals. However, German data protection laws prevent the Munich Hospital from transferring the raw data out of Germany. To give a flavor of possible solutions, one option is to process the data heavily (e.g. obfuscate the data) and then to transfer the processed data. Another option is to send software (e.g. model-training algorithms) to the data, to process the data in Germany, with only the final models leaving Germany.

## 4.3. Global Data Commons

[ImageNet](https://en.wikipedia.org/wiki/ImageNet) is an open dataset with over 10 million tagged images—much larger than previous open image datasets. It’s allowed AI researchers to train image classifiers with radically less error than before, for dozens of computer vision applications.

Similarly, the world would greatly benefit if eventually all datasets for training autonomous vehicles becomes available at an accessible cost to anyone in the world. Even if data is bought and sold at first, ideally the barrier to entry for these datasets to become low, i.e. free (as in beer).

## 4.4. Other

We are iterating with organizations in more than a dozen verticals, including mobility / supply chains, energy, hedge funds, and more.

# 5. Pricing

**Free Data.** We want to encourage a growing data commons for the world. The “as in beer” part is key: there must be an element of participating in the community. The token design elaborates on the incentive structure.

**Non-Free Fungible Data.** Pricing of fungible data is easy(ish): just use an exchange. Exchanges are low friction and let the market determine the price. We plan to support data exchange functionality in the Ocean Protocol.

**Non-Free Non-Fungible Data.** Some data is not fungible. One could simply fix a price. However, it’s often really hard to tell in advance how much value the data has. Price too low, and it’s lost revenue opportunity. Price too high, and no one buys. To address these concerns, we explored several pricing schemes and distilled them into three: fixed price, auction, and royalties. Each has unique mechanics and requirements from a crypto standpoint.

Ocean will make these pricing schemes easy to use by default, but also provides the flexibility to allow developers of marketplaces to easily construct their own.

|  |  |  |
| --- | --- | --- |
| **Scheme** | **Scheme Details** | **Pros & Cons** |
| ***Free*** |  |  |
| **Free (as in beer)** | Retrieve, but must be participating in the community in some way. | Pro: a world data commons |
| ***Non-Free Fungible*** |  |  |
| **Exchange** | Rights holders place asks. Buyers place bids. Exchange matches. | Pro: simple; easy onboarding  Pro: auto pricing, by market  Con: only works on fungible |
| ***Non-Free Non-Fungible*** |  |  |
| **Fixed price** | One fixed price by rights holder. Buyer buys at that price. | Pro: simple; easy onboarding  Con: Worry about “fair” price. |
| **Auction** | Fixed term auction. Highest bidder gets data now. Second-highest in one month. Third-highest in two months, etc, until at 6-mo mark data is free. | Pro: Easy onboarding  Pro: Opens up access to more data over time.  Pro: seller can auction data as often as he/she wishes, e.g. more often for valuable data |
| **Royalties + auto impact** | Data is offered at a royalty % on final value computed in deployment, distributed among data providers. Amount to a provider is proportional to the (auto-computed) impact of that data on value. | Pro: AI models use *way* more data sooner, accuracy↑  Pro: only pay for what helps, so more “fair” pricing  Con: Heavier onboarding  Con: Complexity |

# 6. Stakeholders

The table below lists all stakeholders in the network, what value they can provide the network, and what they might get in return.

The first four rows of the table are the most important. Since Ocean is for data marketplaces, we need suppliers (first row) and incentivized referrers (first row). We also need buyers (third row). Finally, we need keeper nodes to run the network. We have highlighted them, for emphasis.

The remaining rows are stakeholders that add ancillary value, for which they might get rewards from the network in return. It’s a *might* because the network does not need to consider each stakeholder explicitly. It can be simpler for stakeholders to just hold stake (tokens) such that when they contribute to the network, the token value goes up accordingly. Or, if a 3rd party (such as Ocean Exchange Foundation) sees that the stakeholder provides value, then it could contract with or invest in that stakeholder.

Therefore in designing tokens, we are choosing which stakeholders the network sees. We need to account for the highlighted rows in the network itself, e.g. with mining tokens. The rest is optional.

|  |  |  |
| --- | --- | --- |
| **Stakeholder** | **What value they can provide** | **What they might get in return** |
| **Data provider, data custodian, data owner** | Data (market’s supply) | Mining tokens, non-mining tokens, investment |
| **Data referrers. Includes exchanges and other application-layer providers.** | Data (via a data provider etc) | Mining tokens, non-mining tokens, investment |
| **Data consumer** | Tokens | Data (market’s demand), investment |
| **Keepers** | Run nodes in network | Mining tokens |
| **Identity vetters / certifiers** | Improve quality of buyers | Mining tokens, non-mining tokens, contracts, investment |
| **Data vetters / certifiers** | Improve quality of data supply | Mining tokens, non-mining tokens, contracts, investment |
| **Channel partners / affiliate marketers beyond data referral** | New channels for network usage | Non-mining tokens, contracts, investment |
| **Application-layer users** | New channels for network usage | Non-mining tokens, investment |
| **Ocean Exchange core developers** | Improve network capabilities, governance | Non-mining tokens, contracts |
| **Technology suppliers** | Improve network capabilities | Non-mining tokens, contracts, investment |
| **Regulators / government** | Oversight in data governance mechanisms | More widespread enforcement of well-governed data |
| **Founding partners** | Lead network growth | Tokens |
| **Ocean foundation itself; board members; caretakers** | Lead network growth, governance | Tokens |
| **Token holders (ideally, overlaps with all of above)** | Promote network | Rise in token value |

# 7. Key Goals

The main goal of Ocean network is a large supply of data, both of “commons” data and priced data. Here are our key criteria for evaluating tokens:

* For non-free data: incentive for supplying more? Referring?
* For non-free data: good spam prevention?
* For free data: incentive for supplying more? Referring?
* For free data: good spam prevention?
* Does token give higher marginal value to users of the network, vs external investors? Eg Does return on capital increase as stake increases?
* Are people incentivized to run keepers?
* Is it simple? Is onboarding low-friction? Where possible, do we use incentives/crypto rather than legal recourse?

The appendix elaborates on how we arrive at these goals and lists several tools to help solve these issues, from identity tools to reputation tools to governance tools; as well as building blocks in the stack.

The appendix also includes a thorough checklist of questions beyond these core questions, including enumeration of attack vectors (fraud, sybil attacks, more).

# 8. Token Design

## 8.1. Candidate Designs

To address the key goals, we explored a broad variety of token designs. Three main candidates emerged:

1. Incentivize on price data is sold for. adChain registry for actors and data.
2. Incentivize free-as-in-beer data. There’s a price, set through a curation market.
3. Incentivize truly free data. It’s free to download. Incentivize curation. Incentivize CDN-style delivery.

The appendix describes each of these in detail, and compares them.

We chose the third candidate design. We describe it here.

## 8.2. Registry on Data

We use an [adChain](https://medium.com/@AdChain/introducing-the-adchain-registry-cc5b8b831a7e)-style registry of data which are “accredited as non-fraudulent by Ocean token holders.” There are no further mechanisms for actor identity, actor reputation, or data identity.

Someone can only post data if they are the rights holder, or if the data is public domain / CC licensed. Anyone can challenge the poster’s claim at any point (with stake). There is a vote, where “yes” means “data is not junk and rights are ok”. If the majority votes “yes”, the challenger loses staked tokens. If the majority votes “no” then the poster loses their staked tokens (on this dataset), and gets removed from the ocean registry. They do not lose *all* their staked tokens; that would be too harsh.

Different data pricing schemes sit on top.

## 8.3. Free Pricing

We use a [Curation Market](https://medium.com/@simondlr/introducing-curation-markets-trade-popularity-of-memes-information-with-code-70bf6fed9881) for reputation of data. Think of each dataset submitted as a Reddit-style post that can get upvoted.

If you believe a dataset will be useful (downloaded more, or gain in popularity), then you will (a) stake your belief in its value, curation-market style (b) serve it up when asked.

If you don’t serve it up when asked (or fail on other keeper functionality), you will lose your entry in that dataset’s curation market. It’s ok if you are just retrieving it last-minute from S3 or another miner; it’s more reward as a CDN rather than as proof of storage (unlike Filecoin).

Anyone can use data anytime for free, downloading from any of the nodes serving up the data.

The submitter of the dataset stakes 1 Ocean token with an implicit upvote of 1. Someone else can upvote the dataset by staking 2 tokens. The next upvoter must stake 3 tokens, and so on to N upvotes total. At any time, a prior staker can exit and get back (1+2+...+N)/N tokens; there is a downvote. These mechanics incentivize participants to submit quality data, and gives them an opportunity to make money too.

This means that overall, each actor has “holdings” in terms of stake (belief) of the relative value of different datasets. If an actor is early to understand the value of a dataset, they will get high relative rewards.

Curation markets implicitly incentivize referrals.

## 8.4. Non-Free Pricing

Here, the Ocean token is used as a currency for buying and selling. Data is priced in currency of vendor’s choice (e.g. USD or EUR) then converted just-in-time to a token price, according to crypto exchange rates (like Golem).

We explicitly have *no* token mechanisms to incentivize data supply or referral. The data will already get priced according to traditional supply/demand market mechanics; we don’t need to do more. Price is a proxy for data reputation. Compare this to the free data where the curation market signals data reputation.

However, we do need to discourage bad acting. The starting point is the data registry. But going one step further, we use staking. In order to sell a dataset for amount x, the vendor must stake amount x. To sell it twice, they must stake 2x. The stake will get locked up for 28 days, and cannot be re-used during that period.

Someone can only post data if they are the rights holder. Anyone can challenge the poster’s claim at any point (with stake). There is a vote. If the majority votes “data ok”, the challenger loses staked tokens. If the majority votes “data not ok” then the poster loses their staked tokens (on this dataset), and gets removed from the ocean registry. They do not lose *all* their staked tokens; that would be too harsh.

## 8.5. Short Term Governance (Keepers)

**Choosing keepers**

Anyone can download and start running a keeper node. To start with, it is read-only (passive). To become an active keeper, it must get voted active, where vote strength is a function of stake. Therefore becoming an active keeper is a combination of stake and reputation.

Details:

* Votes choose which keepers gets to be active keepers.
* Votes can be delegated to other identities; delegation can be removed anytime (liquid democracy).
* Who can vote: anyone in the ocean registry.
* Vote score by participant *i* = *vote\_scorei = log10(stake).* Therefore higher stake helps a bit, but not much. Basically “one identity one vote” with a small bias to more stake.
* Vote tally for keeper *j* = *vote\_tallyi  = ∑participant\_i (vote\_scorei  if i voted for j, 0.0 otherwise)*
* Vote tallies are taken at the same time interval as block rewards are calculated and divvied up.
* Active keepers are chosen as the keepers with the 100 highest vote\_tally values

Active keepers vote on transactions with one vote each. A transaction goes through if the majority of votes are yes.

As soon as any of a keeper’s votes on transactions are out of alignment with the majority of votes, then the keeper loses their stake, and reward goes to zero. Another keeper automatically replaces them.

**Keeper Incentives**

We aim to incentivize people to run keepers, and to ensure they have sufficiently high performance for the system to make progress.

“Stick” incentive: If you have staked on data, you need to run a keeper and be able serve up that data. If someone queries you for the data and you don’t serve it up, you lose the stake in the data, for both free-as-in-beer and non-free cases.

“Carrot” incentive: Block rewards are distributed across all stakers of all datasets pro rata according to (withdrawal price) \* log10(number of downloads). We use log10 so that less popular datasets continue to be hosted, and new datasets added in.

This pushes the responsibility of storing the data to the data suppliers; incentivizes for data uptime; and has an auto CDN-like functionality to scale up for popular data.

## 8.6. Medium Term Governance (Fixing Bugs)

Anyone can report a bug, as an issue to the appropriate GitHub repo.

Anyone can issue a patch as a pull request to the Ocean codebase, which is then vetted by Ocean core devs. Assuming the core devs accept it, then they will merge the PR and communicate (via social media etc.) a recommendation to the keepers to update the software. Of course someone could fork the code and try to convince the Keepers to update to the forked version. If they’re successful, well perhaps that’s a sign that the core devs or Foundation are doing a poor job, and perhaps the fork should be accepted!

The initial core devs will be core devs for BigchainDB and DEX. After that, core devs are added and removed in a democratic process by votes from existing core devs.

## **8.7. Long Term Governance (Protocol Updates)**

Anyone can propose an improvement, as an issue to the appropriate GitHub repo, and labelled as an “Ocean Improvement Proposal”, and numbered, e.g. OIP 203.

The voting mechanism for choosing keepers is the exactly the same for voting on OIPs, with liquid democracy, weighted voting, and the like. The same stake is used, etc. There are two exceptions: 28 day period; and a grandfathering mechanism. Let’s elaborate. The differences of long time scale to short time scale voting are marked with a “[LT]”.

* An OIP is submitted to the network. Votes are “accept” or “reject” the OIP. The software implementation for both “accept” and “reject” votes must be ready [LT]. FIXME: do we *really* need to have code ready beforehand? Perhaps two levels - proposal and code. Look at BIPs and EIPs and elsewhere.
* Votes can be delegated to other identities; delegation can be removed anytime (liquid democracy).
* Who can vote: any participant who has an identity and a stake within the system. This can be keepers but also simple users of the system.
* Vote score by participant *i* = *vote\_scorei = identity\_score \* stake\_score*, where
  + *identity\_score = {1.0 if e-res, 0.85 if govt, .., 0.0 if none}*
  + *stake\_score = log10(stake).* Therefore higher stake helps a bit, but not much. Basically “one identity one vote” with a bias to identity and more stake.
  + Stake is the *minimum* stake held by the participant in the overall voting period [LT].
  + *voting\_reputation\_score = 1.0 if in “voting” whitelist registry, 0.0 otherwise.* This is the *same* registry as for short time scale voting.
  + Vote tally for “accept” = vote\_tallyaccept  = ∑participant\_i (vote\_scorei  if *i* voted “accept”, 0.0 otherwise). Similar for “reject”.
  + Then, *accept\_rationet = vote\_tallyaccept/(vote\_tallyaccept + vote\_tallyreject).* [LT]
* We have a grandfathering mechanism, where the Ocean Foundation has more control at the beginning, giving time for the network participants to bootstrap in. The foundation itself has governance to include stakeholders by data vendors, data buyers, etc.
  + *accept\_ratiofoundation =* % of foundation caretaker votes that accept, versus reject.
  + *accept\_ratiooverall = Wnet \* accept\_rationet + (1 - Wnet) \* accept\_ratiofoundation*
  + Where *Wnet*=0.0 for the first year after launch of the production net (this buys time to finalize & test governance code), 0.333 for the next year, 0.667 the following year, and 1.0 thereafter.
* Vote tallies are taken at 28 days (four weeks) after the proposal is submitted. This gives enough time for discussion about whether to include a feature or not. [LT]
* The OIP is accepted if *accept\_ratiooverall*  > 0.5. If the OIP is accepted, the keepers automatically update with the new protocol. [LT]

## 8.8. Network value

The network value is the total value of:

* Actor whitelist registry value
* Stakes / belief in data reputation for free data, via curation markets
* Stakes / belief in data value for non-free data, via “must stake x to sell x”

# 9. Technical Dates

## 9.1. Dates leading to Ocean

|  |  |  |
| --- | --- | --- |
| Date | What | Notes |
| June 2013 | DEX project announced - a (centralized) data exchange. | www.dex.sg |
| July 2013 | ascribe project started, for IP on the blockchain. www.ascribe.io | Includes SPOOL, a Bitcoin overlay protocol for IP |
| Feb 2015 | ascribe production release |  |
| July 2015 | BigchainDB project started - scalable blockchain database. | www.bigchaindb.com |
| Oct 2015 | IPDB project started - a public net for BigchainDB | www.ipdb.io |
| Oct 2015 | COALA IP project started - a flexible IP protocol | www.coalaip.org  A collaboration w/ Juan Benet (IPFS/FileCoin), Simon de la Rouviere (Consensys/Ujo), Primavera de Filippi (COALA), more |
| Feb 10, 2016 | BigchainDB v0.1 released | First public announcement |
| Oct 2016 | IPDB testnet released to lead users |  |
| Oct 2016 | COALA IP v1.0 released |  |
| Dec, 2016 | BigchainDB articles on blockchain for AI and big data. | In articles, all roads lead to decentralized data exchanges. |
| Jan 2017 | Ocean project started - general decentralized data exchange | Collaboration between DEX & BigchainDB. |
| Jan 2017 | AVDEX project started - a autonomous vehicle data exchange | Collaboration between Toyota Research Institute & BigchainDB. With an eye to reconcile with Ocean later. |
| May 2017 | AVDEX MVP released | Public announcement of AVDEX. Links to code etc. |
| May 2017 | Ocean technical design starts | Use cases, pricing options, token & governance design, sequence diagrams / protocols, more. Intensive iterations with thought leaders. |
| July 6, 2017 | BigchainDB v1.0 released |  |
| Aug 2017 | Ocean Exchange Foundation formed |  |
| Sept 2017 | IPDB production net released to lead users |  |
| Sept 2017 | Ocean is announced |  |
| Oct, 2017 | Ocean whitepaper v1 released |  |

## 9.2. Software in Ocean Ecosystem

Here are key software components to build:

* Ocean core (Keeper) software. This includes:
  + Support the designed token dynamics, including token storage and smart contracts business logic. Support for free, non-fungible, fungible, and programmable pricing schemes
  + Individual identity / KYC, individual reputation, data identity, data reputation
  + Support for intellectual property claims & licensing (e.g. COALA IP)
  + Integrate metadata storage, w/ privacy considerations (e.g. integrate zero knowledge proofs)
  + Integrate blob storage (on-premise, cloud, decentralized), w/ privacy considerations
  + Integrate compute (on-premise, cloud, decentralized) , w/ privacy considerations (e.g. integrate secure containers, homomorphic encryption)
  + Short time scale governance / consensus (e.g. is a transaction valid)
  + Long time scale governance (e.g. how to update protocol)
  + Well-defined http API and drivers (JS, Py, etc)
  + Individual node deployment tools (e.g. Kubernetes, nginx)
  + This includes work to improve SW of building blocks as needed. This means BigchainDB + IPDB (metadata storage), but also work on IPLD (data interoperability), ILP (value interoperability), COALA IP, IPFS, Ethereum etc. as needed
* Management & analytics of Ocean network
  + dashboard for management & analytics of individual nodes
  + dashboard for analytics on overall network
  + dashboard for voting in long time scale governance
* Data marketplace template software & legals (to be used by DEX; open-sourced for others). This includes:
  + GUI for discovering data
  + GUI for making commons data available in Ocean
  + GUI for buying & selling data according to various pricing schemes
  + Data compliance frameworks (e.g. GDPR)
  + Data quality frameworks
  + Well-defined http API and drivers (JS, Py, etc)
* Software & support to support the ecosystem and catalyze the community
  + Low friction for crypto wallets (e.g. follow latest token protocol)
  + Low friction for crypto exchanges
  + Hooks into other data networks

## 9.3. Ocean Software Releases

Note: our software releases are named after Ocean planets [found in science fiction](https://en.wikipedia.org/wiki/Planets_in_science_fiction#Ocean_planets).

FIXME this needs to be fleshed out

|  |  |
| --- | --- |
| **Name / Date** | **Functionality** |
| Thalassa  Nov, 2017 | +Ocean MVP for sandbox at Singapore data authorities (IMDA) |
| Hydros  Q1 2017 | +Testnet  +Support vertical: self-driving cars  +Pricing: Fixed.  +IP contracts |
| Tethys  Q2 2017 | +Pricing: Auction. |
| Muur  Q3 2017 | +Production net  +Support vertical: health with privacy  +Privacy w/ on-premise compute with secure containers or HE |
| Manaan  Q4 2017 | +Pricing: Royalty-based; auto compute impacts of data to price. |
| Solaris  Q2 2018 |  |
| Kamino  Q4 2018 |  |
| Venus  Q2 2019 |  |
| Earth  Q4 2019 |  |
| TBD  Q4 2020 |  |
| TBD  Q4 2021 |  |
| TBD  Q4 2022 |  |

# 10. Conclusion

This paper presented Ocean, a protocol and network to power decentralized data exchanges.

# 11. Acknowledgements

The lead authors (Trent McConaghy, Troy McConaghy, Tim Daubenschütz, ) would like to thank everyone who gave feedback, comments or other contributions to this paper, particularly Ryan Zurrer, Fred Ehrsam, Chirdeep Chhabra, Bruce Pon, Maciek Laskus, Alberto Granzotto, Dimi de Jonghe, and the rest of the BigchainDB & DEX teams. FIXME: more people to thank.

# 12. References

FIXME add in refs that we currently only have linked.

[COALA IP] <https://www.coalaip.org/>

[Crevier1993] Crevier, Daniel. *AI: The tumultuous history of the search for artificial intelligence*. Basic Books, 1993: 148–150.

[Economist1] “[The world’s most valuable resource is no longer oil, but data.](https://www.economist.com/news/leaders/21721656-data-economy-demands-new-approach-antitrust-rules-worlds-most-valuable-resource)” *The Economist*. 6 May 2017.

[Filecoin] <https://filecoin.io/>

[Heath1956] Heath, Thomas Little, ed. *The thirteen books of Euclid's Elements*. Courier Corporation, 1956.

[IPDB] <https://ipdb.foundation/>

[IPFS] <https://ipfs.io/>

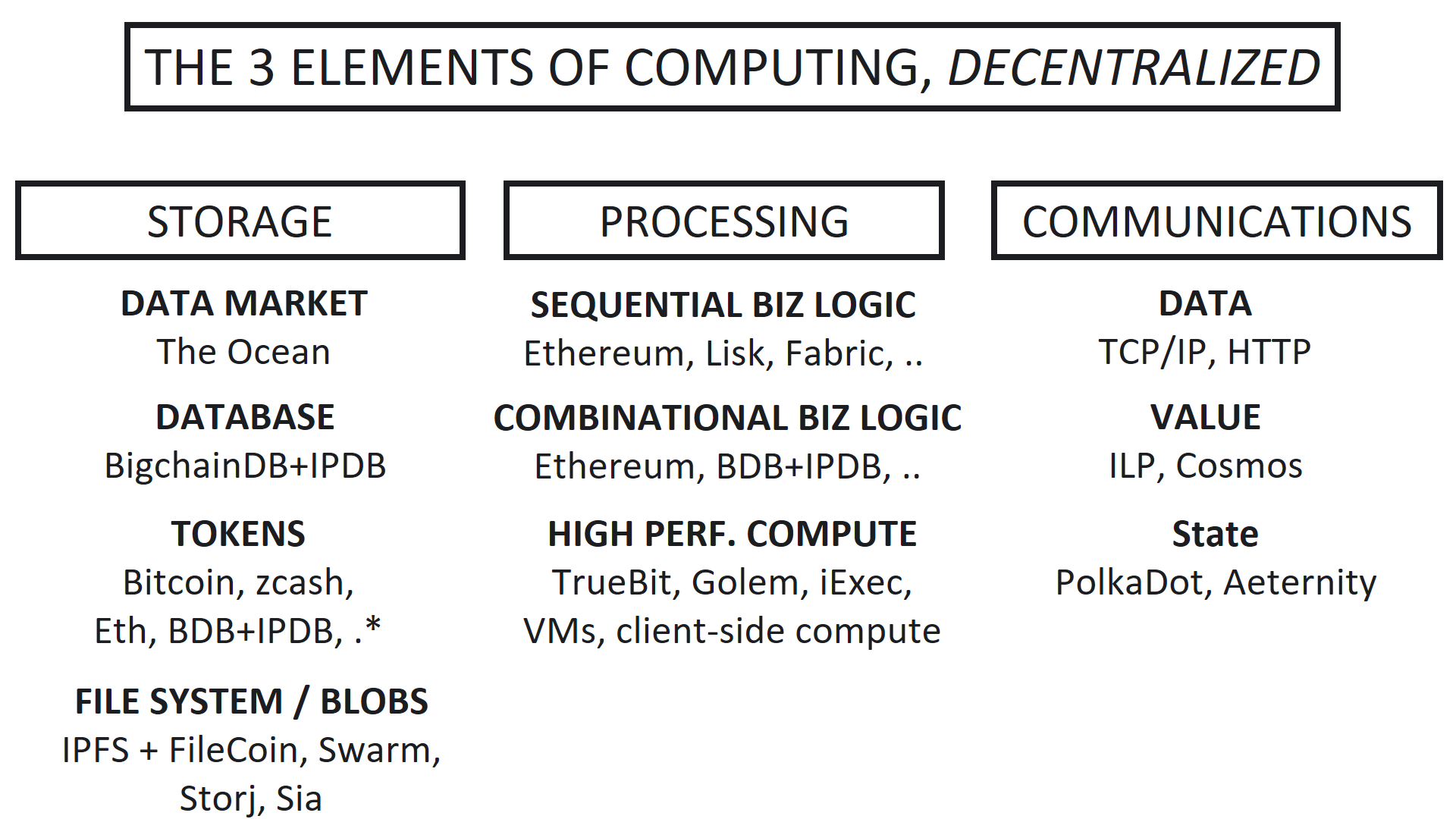
[Moor2003] Moor, James, ed. *The Turing test: the elusive standard of artificial intelligence*. Vol. 30. Springer Science & Business Media, 2003.

[Storj] <https://storj.io/>

[Swarm] <http://swarm-gateways.net/bzz:/theswarm.eth/>

# 13. Appendix: Ocean in the Decentralized Stack

Because equalizing access to data is key to the future of the internet, we see that data marketplace protocols are a key storage-related block in the [blockchain infrastructure landscape](https://blog.bigchaindb.com/blockchain-infrastructure-landscape-a-first-principles-framing-92cc5549bafe).



# 14. Appendix: Target Specifications

We have several target specifications, which we group into four categories.

**A. Network Control is Decentralized**

* **A medium to connect** the data-haves with data-consumers. Enterprise meets AI startup.
* **Decentralized** control, yet one logical place.
* **Aligned** **community** with skin in the game. Staked tokens. Incentivize quality data supply.
* **Transparent governance** with clear rules

**B. Data Control is User-Specific**

* **Data rights & privacy** are first-class citizens. Bring compute to the data. Tokenized read permissions.
* **Compliance with data laws**, regulations and policies.Work closely with regulators.
* **Tokenized data as IP, with legal teeth.** COALA IP protocol, baked-in contracts.

**C. Fair, Flexible Market Mechanics**

* **Low friction to pricing.** Good defaults for selling non-fungible and fungible data.
* **Avoid “data escapes”.** Owners not fraudsters get paid.
* **Programmable pricing schemes.**

**D. Designed for Industrial-Scale Big Data and AI Applications**

* **Industrial-scale building blocks.** Efficient computation on private data. Identity with E-Residency, more. Scalable blockchain database BigchainDB.
* **Visible provenance of data** and models.
* **Control the upstream of your data** & models.
* **Easier data pooling** for better & qualitatively new AI models

# 15. Appendix: Realization of Target Specifications

Here, we describe each target characteristic in even further detail, and discuss at a high level how Ocean realizes that characteristic. Later sections of the paper elaborate further.

**A medium to connect the data-haves with data-consumers**

This baseline problem deserves re-stating. Many large enterprises have tremendous amounts of data, but do not know how to unlock its full potential. Conversely, many startups with deep AI expertise are starving for data. If we can create a medium to connect these parties, it will unlock tremendous value for both sides. We can generalize this to pure-play data creators and users (without involving AI).

The high level approach is to provide a data marketplace that connects the data-haves (typically enterprises) with data-consumers (typically AI startups).

**Decentralized control, yet one logical place.**

A large number of data providers must participate to make an ocean worth of data available for sale. For that to happen, the marketplace must first feel *fair*. It can’t be controlled by any single entity that might pull the plug at any minute or change the rules. We need to get past centralized infrastructure with dictatorship-style control.

Fortunately, new technology makes this possible: decentralization technology spreads the marketplace infrastructure (and its control) among many parties. No single person or entity owns or controls the network. There’s no all-powerful administrator. It works because the nodes all follow the same rules (protocols). The rules are encoded in openly-readable software. (Actually, some of the nodes could break the rules, but as long as enough of the nodes follow the rules, then the system will work properly. There are strong incentives to encourage participants to follow the rules.)

All finalized transactions are recorded in an cryptographically-signed, immutable public record, which makes them auditable by anyone.

Users don’t have to trust some third party to act fairly based on their reputation. Trust arises from open protocols, cryptography, immutable records and strong incentives.

The network nodes can be spread all over the world, so a natural disaster or change in laws won’t affect the entire network. The system is resilient.

**Aligned Community with Skin in the Game**

To be truly successful, Ocean needs to have a vibrant, aligned community with strong network effects. Towards that end, we ask several questions. How can we align incentives among participants in the marketplace, rather than having vendors compete in zero-sum games (when one wins, the other must lose)? How can we incentivize sellers and buyers to stick around? How can we incentivize referrals?

Within the decentralization community, *tokens* have emerged as a potent answer to these questions. Here, we use data tokens called Ocean Tokens for the community involved in Ocean’s data marketplace. Ocean Tokens act as a currency to buy and sell data. We also want data buyers and sellers have skin-in-the-game, and to de-incentivize particular bad behavior. To this end, market participants must *stake* tokens to participate. Staking also plays a role in network governance.

Because Ocean Tokens are crypto tokens living on a decentralized blockchain database, they can be traded on crypto exchanges and have their value rise as the value of the marketplace rises. The ecosystem participants (holders of Ocean Tokens) then have an aligned incentive to increase the value of Ocean Tokens; this makes participation in the ecosystem a positive-sum game (win-win) rather than a zero-sum game.

**Transparent Governance with Clear Rules**

Not all data marketplaces are operated with clear rules or transparent governance. In centralized marketplaces, terms of service can change suddenly and unexpectedly, possibly for the worse. Ocean must be operated transparently and openly, so that all stakeholders can see what’s going on, have a say in decision making, and know what to expect. We must avoid centralization of power, and of whales taking control. There must be means for fast decision-making

Ocean will have governance in two places:

* Governance of Ocean Foundation. It covers long-term network guidance (not governance), community building, managing foundation funds, etc. This largely draws on our experience in designing governance for the decentralized database IPDB; which in turns has roots in the governance of the DNS.
* Governance of Ocean network. How a transaction gets validated, how protocol gets updated, etc.

To design these, we are drawing on our own experiences in designing. We also draw on other learnings (good and bad) in decentralized governance from the last few years, from Bitshares to TheDAO to Maker, acknowledging the role of both staking and reputation, and the need to move fast yet the desire to avoid an oligarchy.

**Data rights & privacy**

Privacy is a first-class citizen. The data providers have both rights and control on the use of their data, with verifiable audit. Ensure that data itself can stay with the data owners, by e.g. bringing the compute to the data, behind a firewall. Also, allow clean control over purchased data, e.g. by tokenizing read permissions for encrypted data.

There are several good reasons when it’s not a good idea for the data to leave the premises:

1. The data cannot leave the jurisdiction for legal reasons, like German medical data
2. The data provider feels uncomfortable to release the data
3. The dataset is so massive that it doesn’t make sense to transmit it over the network.
4. The data is highly sensitive even years from now, and should not get out in the advent of quantum computing. Here’s the scenario. Imagine if some sensitive data was encrypted and posted publicly. Perhaps it’s secure now. However, when quantum computing gets good enough for dedicated hackers to use (in say 5-20 years) then we want to make sure they don’t decrypt that sensitive data. We can’t just remove the encrypted data from the network either, because it may be stored immutably (IPFS/Filecoin style) or savvy hackers may already be making their own copies in preparation. In short: if data is still sensitive after a few years, storing it encrypted on a public net is not enough.

Several technologies can help. First, we can leverage crypto and blockchain technologies for access control.

* **Traditional crypto.** Encryption for data safety. Digital signatures for proof that X did Y.
* **On-premise storage & firewalls.** Store data behind firewalls. This way, when quantum computing goes mainstream, public encrypted data doesn’t get exposed.
* **Role-Based Access Control (RBAC)** aka Tokenized Read Permissions. Users are assigned roles as tokens. Those roles are assigned read permissions as tokens. This is a BigchainDB add-on built in Aug-Sep 2017.
* **Access provenance.** Here, the blockchain stores provenance of what data was accessed when, by whom. This is a BigchainDB add-on built in Aug-Sep 2017, and is key part of the IMDA Sandbox for Nov 2017.

Next, we can bring compute next to the data, i.e. on-premise compute:

* **Federated learning.** Update an AI model by sending a training code & initial parameters to each data-holder, computing data-side, and sending back a gradient update to the model. The model must have decentralized control so that a single entity can’t choose privacy-compromising initial parameters (e.g. a vector of zeros). Example: OpenMined.
* **Homomorphic encryption (HE)** with federated learning. Make model gradient updates on *encrypted* local user data. HE is efficient with thoughtful choice of AI algorithms, e.g. rectified linear units rather than sigmoids. Example: OpenMined.
* **Secure containers.**  One container can see the data, but not the network. Another container does model building and can’t see the network. To talk to the network it must talk locally with a blockchain-powered “gateway” container that *can* see the network. Example: Amethix.

Finally, we can anonymize and obfuscate. There are more techniques as well.

* **Anonymization / Obfuscation.** From a raw dataset, compute a new dataset from which the original dataset cannot be inferred, while retaining the ability to train a model. Example: Numerai’s use of GANs.

**Compliance with Data Laws, Regulations and Policies**

Data owners, licensors, buyers, and licensees usually want to comply with many data-related laws, regulations and policies. They might be external regulations, imposed by a government (for example) or the internal policies of a company. Examples include:

* The Health Insurance Portability and Accountability Act of 1996 (HIPAA) in the USA
* The General Data Protection Regulation (GDPR) in the EU
* The Sarbanes-Oxley Act of 2002 (SOX) in the USA
* The internal data policies of a company.

Not all data-related regulations are data privacy regulations. Some require increased data auditability or transparency, for example.

Ocean is working with regulators and enterprises to ensure that Ocean facilities compliance with these rules. And, Ocean can help all parties with compliance.

Identity (potentially as high as KYC) may be required for certain data jurisdictions. In general, each of these identity providers provides a public key, and some secure means of signing with a private key.

* **Government.** One key option is a government-backed identity which can be used globally; currently that is only Estonia through its E-Residency program. Estonia issues E-Residency cards which have inbuilt private keys. We are hopeful that in the future, other national governments will follow Estonia’s lead.
* **Decentralized.** Another key option is decentralized identity providers, such as Authenteq (which provides super-simple KYC in many jurisdictions), uPort (which has especially good traction in the Ethereum community), constent.global (identity based on social media accounts), or Civic (token-incentivized identity).

**Tokenized data as IP, with legal teeth**

Explicitly treat data as intellectual property. Have an appropriate protocol for data-as-IP within a blockchain context. Tie in appropriately with IP & contract laws.

Details. If you create a data set or a specific AI model, then it’s automatically is protected by copyright and is protected by copyright laws. That is, data sets and AI models are intellectual property (IP) and can be regarded as assets. We sometimes refer to them as “data assets” or “DataIP.”

If you own some data assets, then you can license various rights to others. For example, you can license your data to someone else to train a model. Or you could train a model and license your trained model for someone to include in their mobile application. Sub-licensing, sub-sub-licensing, etc. is possible too. You can license data or models from others as well.

Copyright laws have enabled licensing for quite some time. Blockchain technology can be used to enhance the licensing process. For example:

* You can publish a cryptographically-signed claim of copyright to a global, tamper-resistant public registry (i.e. a blockchain).
* You can write a specific license to grant certain rights (with restrictions) to the owners of the license, and that license can be written to the same registry/blockchain.
* That license can be officially granted to another party (a licensee) using a blockchain-style transfer, i.e. only you can make the transfer because only you have the necessary private key.

Ocean will use [IPDB](https://docs.google.com/document/d/1O52_ug9BjkIcHfqR-nShrksejePW5YjTJOBzeSk1i0A/edit?pli=1#bookmark=id.du7tpffgkkbw) (powered by BigchainDB) as its blockchain and the [COALA IP](https://docs.google.com/document/d/1O52_ug9BjkIcHfqR-nShrksejePW5YjTJOBzeSk1i0A/edit?pli=1#bookmark=id.dckxfhzfsd6p) protocol for intellectual property licensing.

A note on terminology: when someone “buys a song,” they’re usually buying a license. They’re not buying the copyright to the song. We use similar language in this paper. For example, when we write “buy data” or “sell data,” we almost always mean buy a license or sell a license for some data assets.

**Low friction to pricing**

Easy onboarding to sell or buy data, but with more effort, it is more “fair” (e.g. auto impacts per dataset, or more fungible data).

When a data buyer comes to a data marketplace, they want to get data that’s useful to them, and at a fair price. But among all the data sets available, how can they know which will be the most useful? They need enough information to decide. Ideally, they could just get all the data and then pay in proportion to how useful that data was to them, but that’s usually not an option. The data seller may not want to, or may not be able to release the data (e.g. for compliance reasons, compliance with external or internal data protection policies or regulations). They might allow the buyer to send some code to their data to calculate something, or to get a small data sample, or to get a filtered version of the data, for example.

Somehow, a price must be determined so that both buyer and seller are happy, while satisfying their constraints. Ocean currently plans to offer four pricing options: if the data is non-fungible, then there are three options: a fixed price set by the seller, an auction, or automatic (algorithmic) pricing, if possible. If the data is fungible, and there are enough buyers and sellers, then the price can be determined by an exchange (with asks and bids).

As mentioned earlier, it’s tricky to find a price for a dataset so that both the buyer and seller feel like they have gotten a fair deal. The buyer needs enough information to decide on a fair price, but the seller doesn’t want to (or might not be allowed to) reveal too much data in advance.

**Non-fungible data.** The thorniest challenge is for datasets which are non-fungible. For example, with self-driving car data, a kilometer driven in a Canadian blizzard is worth more than a kilometer driven in sunny California. Each dataset may be completely different, from weather patterns to stocks to Twitter feeds.

**Fungible data.** For data that is fungible, the answer is simpler. We can automatically price it using a standard exchange setup: an order book of bids and asks, with an automatic matching algorithm.

We also need to support - and incentivize - **free data** leading to a healthy data commons for the planet.

**Avoid “data escapes”**

We don’t want people copying data they don’t own, or people buying data for their own use but making it available for free to everyone. (Note: this is different than industry jargon “free riding”).

**Programmable pricing schemes**

Have good default schemes for easy onboarding. But, be programmable to allow developers to construct novel pricing schemes.

Make sure we have flexibility beyond the initial four pricing schemes. Not only to support them, but to make it low barrier to entry for new schemes to enter. Ideally: don’t favor the four, rather, be generic and say “everyone is equal”. Eg EVM opcodes are very general. So is IPDB transactions, with CCs etc. So here we say “we have built the first four pricing schemes using CC + private data sharing, here are the json-ld txs, but you can build whatever else and have it in the marketplace. Marketplace “apps”. Or like Ethereum tutorials.“

Ocean’s building blocks can be assembled by developers to suit their particular needs for a given data submarket. This includes providing good primitives for bids, asks, tokenized read permissions, and more.

**Industrial-Scale Building Blocks**

An architecture for planetary scale. Can handle massive amounts of data. Thousands of transactions per second minimum. Metadata storage is fast.

Efficient computation on private data (this precludes homomorphic encryption).

Building blocks in the system are best-in-class. Includes blocks for identity, compute (on-premise, centralized, decentralized), blob storage (on-premise, centralized, decentralized), metadata / token storage & querying database (must be decentralized, therefore using BigchainDB).

Let’s elaborate on a few of these blocks.

**Compute.** In many cases, data is just bought and sold; there is no need for compute within the Ocean system. That is the simplest, of course. However, it is not every case.

* **On-Premise.** There are times when the data cannot leave the premises, as described above. In this case, we can bring the compute to the data using Amethix’s secure-container technology.
* **Decentralized.** There is an emerging set of decentralized compute providers like Golem who will be useful to handle massive compute loads more cheaply and potentially with privacy-preserving measures.

**Data (blob) Storage.** This is where the data is actually stored. The Ocean network does not have incentive to store the data for its own sake; however the providers / custodians / owners will have the data and must provide access to the data consumers in some fashion. There are at least three options, depending on the needs:

* **On-premise (behind firewall).** There are times when the data cannot leave the premises, as described above. In this case, we might use on-premise storage might be on a simple Linux file system running on a single server box, or perhaps HDFS running on an enterprise data center. It would all be behind a firewall, of course.
* **Centralized cloud.** Here, the data is stored on something like AWS S3. It provides more convenience at a cost of some security and bandwidth usage. It’s still centralized, of course. It would be behind a firewall and possible encrypted.
* **Decentralized cloud.** If one doesn’t have the constraints leading to on-premise, then storing the data decentralized might be a great option. It decentralizes control (a plus) and could be much cheaper than centralized cloud storage too. Options include [IPFS](https://docs.google.com/document/d/1O52_ug9BjkIcHfqR-nShrksejePW5YjTJOBzeSk1i0A/edit#bookmark=id.mkdcgva0alu5) plus physical storage like [Swarm](https://docs.google.com/document/d/1O52_ug9BjkIcHfqR-nShrksejePW5YjTJOBzeSk1i0A/edit#bookmark=id.bng1x61exncp), [Storj](https://docs.google.com/document/d/1O52_ug9BjkIcHfqR-nShrksejePW5YjTJOBzeSk1i0A/edit#bookmark=id.7si6l01wmxq6) or [FileCoin](https://docs.google.com/document/d/1O52_ug9BjkIcHfqR-nShrksejePW5YjTJOBzeSk1i0A/edit#bookmark=id.qhr8yq6d5zni): decentralized file systems to store the large data and model blobs.

**Token & metadata storage.** [IPDB](https://docs.google.com/document/d/1O52_ug9BjkIcHfqR-nShrksejePW5YjTJOBzeSk1i0A/edit#bookmark=id.du7tpffgkkbw) network running BigchainDB software is appropriate here. BigchainDB is a scalable blockchain database that can store both tokens (assets) and structured metadata. IPDB is its public net.

**Enable Provenance of Data and Models**

Once data (and models) are part of an ecosystem, their history of usage and ownership—their provenance—fall out. That history is valuable in its own right. It allows one to gain trust in the data and the models. Put another way, it allows data and models to get their own *reputation.*

If you train a model on garbage data, then you’ll get a garbage model. Same thing for testing data. Garbage in, garbage out.

Garbage could come from malicious actors who may be tampering with the data. Garbage may also come from non-malicious actors / crash faults, for example from defective IoT sensor, a data feed going down, or environmental radiation causing a bit flip (sans good error correction).

How do you know that the training data doesn’t have flaws? What about live usage, running the model against live input data? What about the model predictions? In short:what’s the story of the data, to and from the model? Data wants reputation too.

Blockchain technology can help. Here’s how. At each step of the process to build models, and to run models in the field, the creator of that data can simply time-stamp that model to the blockchain database, which includes digitally signing it as a claim of “I believe this data / model to be good at this point.” Let’s flesh this out even more…

**Provenance in building models:**

1. Provenance on sensor data (including IoT). Do you trust what your IoT sensor is telling you?
2. Provenance on training input/output data.
3. Provenance on model building itself, if you like, via trusted execution infrastructure, or TrueBit-style markets that double-check computation. At the very least, have evidence of model-building with the model-building convergence curve (e.g. *nmse*vs. *epoch*).
4. Provenance on the models themselves.

**Provenance in testing / in the field:**

1. Provenance on testing input data.
2. Provenance on model simulation. Trusted execution, TrueBit etc.
3. Provenance on testing output data.

We get provenance in both building the models, and applying them. The result is more trusted training data & models.

And we can have chains of this. Models of models, just like in semiconductor circuit design. *Models all the way down.*They can all have provenance.

**Benefits include:**

* Catch leaks in data supply chain (in the broadest sense), at all the levels. For example, you can tell if a sensor is lying.
* You know the story of the data and model, in a cryptographically-verifiable fashion.
* You can catch leaks in the data supply chain. That way, if an error happens, we’ll have a much better idea of how and where. You can think of it as banking-style reconciliation, but for models.

Data gets a reputation, because multiple eyes can check the same source, and even assert their own claims on how valid they believe the data to be. And, like data, models get reputations too.

**Enable Controlling the Upstream of Your Data & Models**

As explained earlier, datasets and models are protected by copyright, so they can be treated as assets, which can then be licensed. Blockchain technology can be used to record all registrations and licenses. It can also be used to enforce licenses, and to control who gets them.

For example, if you have some personal data about yourself, you could claim the copyright on that data, create some licenses, and sell those licenses. If someone used that data to develop a model, then your license might specify restrictions on how that model can be used. In short, blockchain technology can help people control the upstream of their data.

This is likely part of DeepMind’s strategy in their [healthcare blockchain project](https://news.bitcoin.com/deepmind-healthcare-ai-blockchain/). In data mining, healthcare data puts them at risk of regulation and antitrust issues (especially in Europe). But if users can instead truly own their medical data and control its upstream usage, then DeepMind can simply tell consumers and regulators “the customer actually owns their own data, we just use it under license from them.” Lawrence Lundy wrote:

*It’s entirely possible that the only way governments will allow private ownership (human or AGI) of data is with a shared data infrastructure with “network neutrality” rules, as with AT&T and the original long lines. In that sense, increasingly autonomous AI requires blockchains and other shared data infrastructure to be acceptable to the government, and therefore to be sustainable in the long term.*

**Ease Data Pooling for Better & Qualitatively New AI Models**

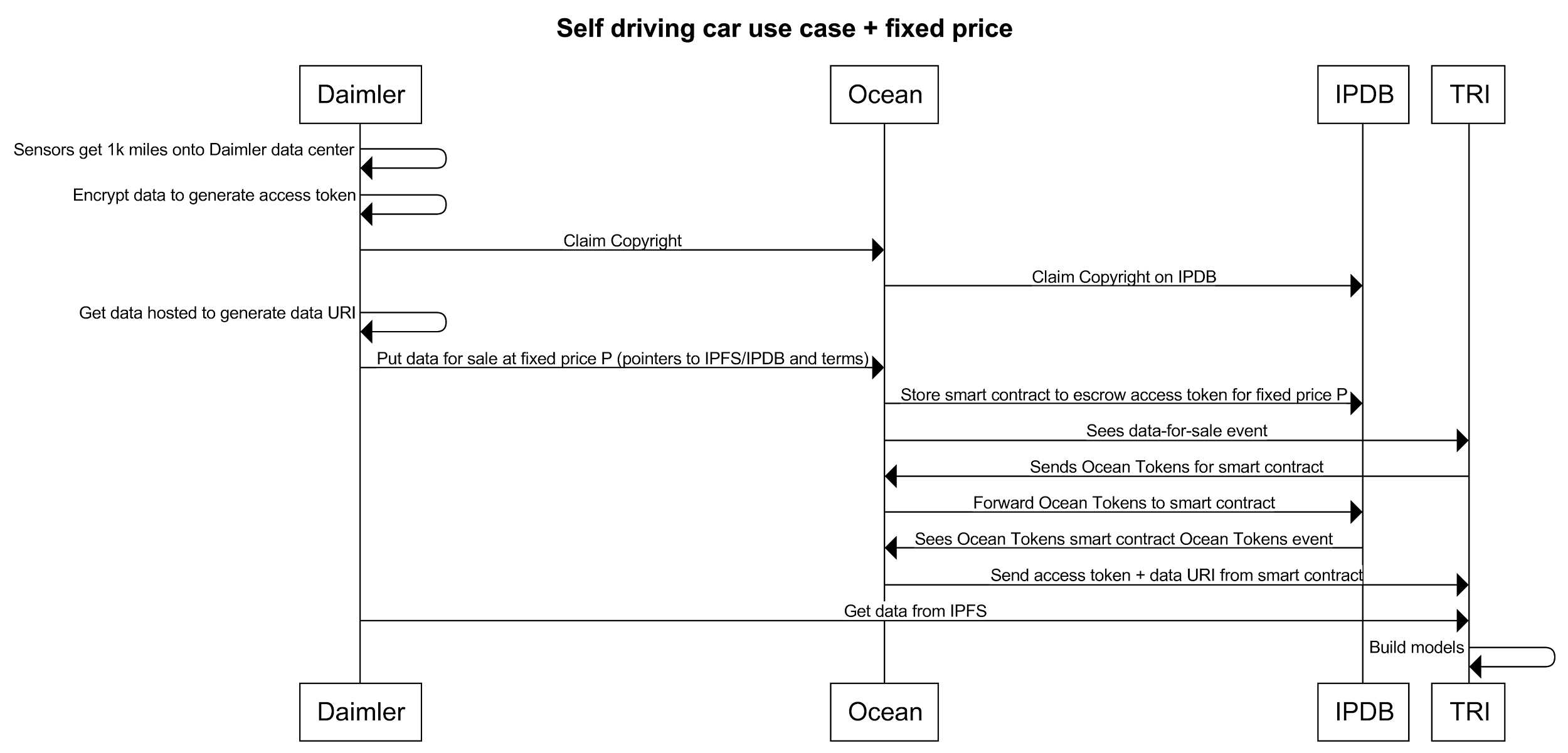
Ocean reduces the friction to getting pooling datasets -- building a model from way more datasets, which means the model accuracy will go up. In some cases, when data from silos is pooled, you don’t just get a better dataset, you get a qualitatively new dataset, which leads to a qualitatively new model. This is especially powerful with data at planetary scale.

# 16. Appendix: Detailed Protocol Sketches

The introduction provided a figure showing the protocol sketch for self-driving cars with an auction price. Here, we provide two more examples. FIXME insert our detailed sketches that flesh this out to implementation level

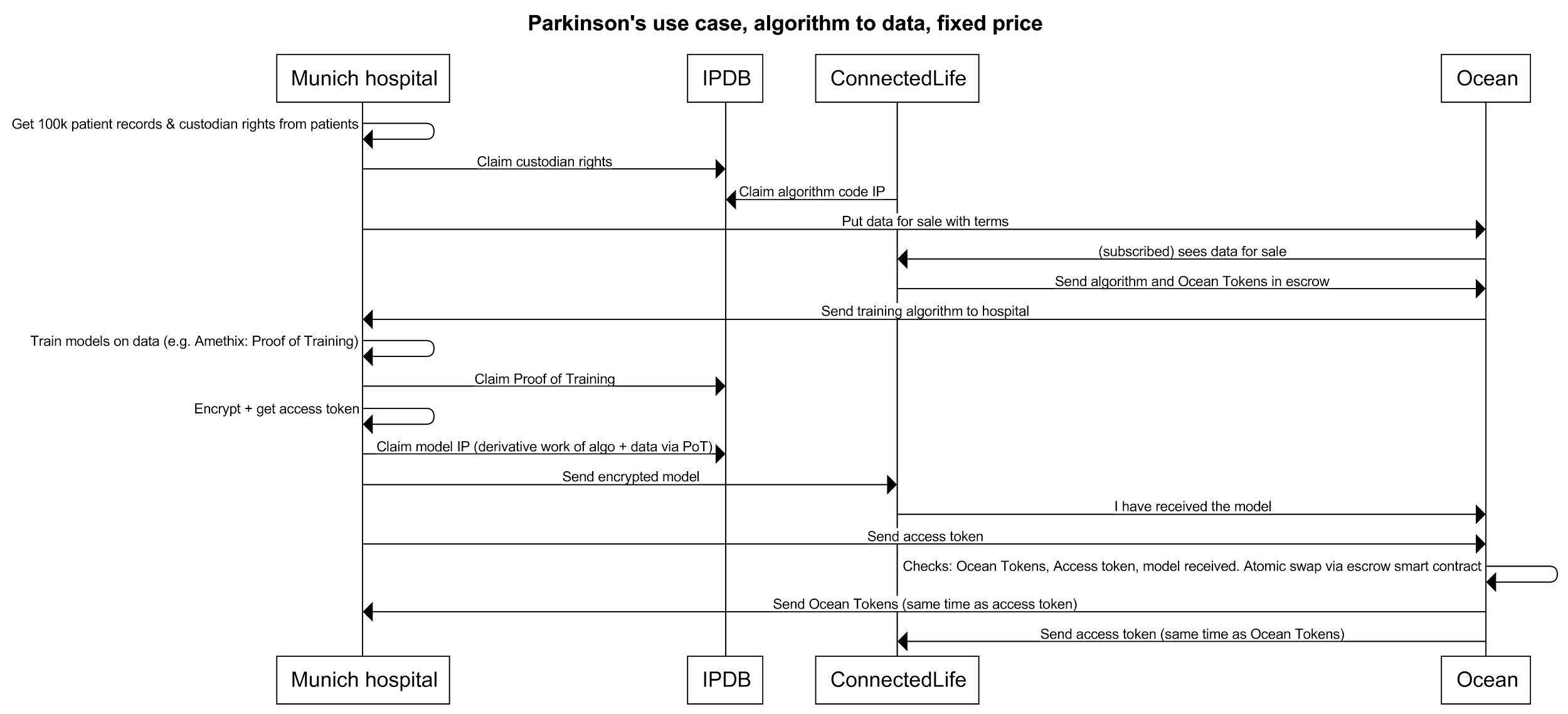
## 16.1. Self-driving cars with fixed price

This is probably the simplest use case. Privacy is not an issue, and pricing is directly set.



## 16.2. Parkinson’s with fixed price and on-premise compute

This has simple pricing. But the data must stay on-premise, for reasons of privacy, not to mention data protection regulations. Therefore the compute is brought to the data.



# 17. Appendix: Token Design Goals, Questions, and Tools

## 17.1. Buy/Sell/Incent Data

**Goals.** We group goals into categories, starting from core functionality. This and the following sections cover the goals. We will **bold** the key goals.

The main thing we want to incentivize in the network is togrow the supply of data to a massive amount.Once we have data, everything else falls out. A major tool for this is data referrals. Related goals:

* **Key goal: Is there an incentive for supplying more data?** Data of high quality? Sooner rather than later?
* **Key goal: Is there an incentive for referring others’ data?** Eg return on capital is proportional to stake.

We want Ocean to also supply data in the commons. To that end, we add two more goals:

* **Key goal: Do we incentivize for supplying and referring free data (“data commons”)?**
* **Key goal: For free data, do we prevent spam?**

Of course we also to ensure the token supports the core functionality of buying / selling, at the key pricing mechanisms (fixed price, auction, exchange). Related goals:

* **Key goal: Is the token design sufficiently general to support our target pricing schemes, free prices, and other developer-defined pricing schemes?**
* For fixed-price pricing: Is there a means to post a fixed price? Agree to buy at that price?
* For auction pricing: Is there a means to post for auction? A means to bid in that auction?
* For exchange pricing: Is there a means to make an ask? Make a bid? Eg 0x protocol.
* For free pricing: who pays for compute resources? (Storage, write bandwidth, read bandwidth)

**Tool: Referral rewards.** Here are options to incentivize supply & referral:

* **Share mining rewards pro-rata wrt selling price.** In any given time period where a total value of *x* tokens’ worth of *y* data is sold, give mining rewards pro-rata based on price of data to the suppliers/referrers of *y* data. Assume that mining rewards exponentially drop over a period of years. Issue: this is a zero-sum game among data referrers, for that period. So “sale whales” (large sales) dominate. The dynamics are different if this is lumped over 10 minutes versus one week, but in all cases the sale whales still dominate. Another issue is that: reward could be unreasonably massive if there isn’t much activity; but then again that automatically induces people to participate, so it will balance itself, and concern is gone. Related is that markets will likely be more active in the day; so savvy users would bias to transactions in the middle of the night. While this has a nice load-balancing effect, it may be an unwanted distortion.
* **Share mining rewards pro-rate wrt price\_score.** Like above, fixing the problem of “sale whales” by giving diminishing returns for sale price: price\_score = sqrt(sale price) or log10(sale price). This implicitly encourages more diversity of data for sale.
* **Reward 5%\*price of until reward pool empty.** Have a fixed pool of reward tokens that get doled out. Suppliers/referrers get rewarded 5% of data sale price. (And of course the supplier themself gets the full sale price.) When those rewards are spent, that’s it. Ideally there are enough rewards such that by the time the pool is empty, the data in Ocean is massive. Issue: This is prone to gaming; though that can be fixed with e.g. an adChain-style registry, reputation, or some other mechanism. Issue: this doesn’t reward super-early providers versus slightly later providers. Issue: sale whales will drain this faster.
* **Reward time\*5%\*price of until reward pool empty.** Like previous, but add a penalty for lateness. Referrers get (5% of data sale price) \* time\_multiplier, where time\_multiplier = 1.0 to start, but diminishes exponentially to <0.1 within two years.
* **Reward time\*5%\*price\_score of until reward pool empty.** Where price\_score is sqrt(price) for example. Addresses “sale whale” issues.
* **Min(shared mining reward pro-rata wrt selling price, 5%\*price).** Takes the best of the previous designs, by sharing mining rewards, but not squandering tokens if there isn’t much activity. Issue: more complex.

## 17.2. Economics

All these incentives do not matter if the token loses its value, or barely increases in value over time. It has to relate to some economic rights. Therefore we ask the following (drawing on [[Smith2017](https://www.smithandcrown.com/token-rights/)]):

* **Key goal: Do we address: How might the token increase in value?**
* **Key goal: Do we address: Which of the following rights does a token holder get? {Payment, access, profit or fee, contribution, block creation, governance}**

Finally, we have a bias to simplicity, potentially made possible by markets/incentives, crypto, and AI, before leaning on legals. For example, use a deterministic crypto proof rather than “go to the courts”.

* **Key goal: It simple? Is onboarding low-friction? Where possible, do we use incentives/crypto (and maybe AI) rather than legal recourse?**

## **17.3. Towards Good Acting via Staking, Id, Reputation**

**Goals**

In the case of Ocean, we want to incentivize good acting. Put another way, we want to disincentivize bad acting, including but not limited to fraud. We want “skin in the game”. Goals related to good acting:

* **Key goal: Is there a means to get high-quality metadata? Eg How do we prevent non-owners of the data from submitting that data?** (Fraud).
* **Key goal: Are we incentivizing skin-in-the-game?** E.g. Does return on capital increase as stake increases?
* Do big providers of data need to stake a lot? Consumers?
* If I have high stake but low reputation, can I make $? If I have low stake but high reputation, can I make $? If I have high stake and high reputation, can I make $$$?
* Do keepers (at least keepers with a higher level of reward or privilege) need to stake a lot?
* Is there a good threshold of individual / org identity - are they are who they say they are? At the very least, to prevent Sybil attacks. But potentially more, to adhere to data privacy regulations.
* Is there a good measure of individual / org reputation - are they a good actor in the ecosystem? (In buying, selling, keeping, etc?)
* Is there a good threshold of data identity - is the data what they say it is?
* Is there a good measure of data reputation - is the data useful?
* Does remuneration favor data freshness?

Let’s explore some key tools that can help.

**Tool: Stake**

As a design principle, the **token should give higher marginal value to users of the network, compared to external investors** (thanks to Fred Ehrsam for this framing).Staking is an excellent tool towards this principle.

Staking means that users literally have a vested interest in the system, that they have “skin in the game”. We want to ensure that key users hold (“stake”) based on using the system or validating the system. Stake needs to be intrinsic to the system.

We want staking for selling data, buying data, keeping (running a node), and maybe more.

As a design principle, one great way to give we can **make *return on capital* to be proportional to stake**, much like interest in a bank deposit (thanks to Ryan Zurrer for this framing). We treat this as a key goal in token design. This is an excellent means to achieve the key goal of “Are we incentivizing skin-in-the-game?”

Examples:

* adChain. To be part of a registry of “good” actors, you need to stake. Others can bet against you in an initial vetting process.
* Numerai. If you want to submit your model and bet on its worth, you need to bet (stake) with Numeraire. It’s locked up for one month while the trading happens.
* Eos. You get more computing power if you stake more.
* Ethereum Casper. You get inflation in your Ether holdings unless your run a validator.

Akin to “double spending”, we also need to avoid “double staking” where one token is used as collateral across many things at once. Allowing that would greatly increase the system’s risk exposure compared to the staker’s risk exposure.

Staking-related questions:

* **Key question: Does token give higher marginal value to users of the network, compared to external investors**
* Do we avoid double-staking?
* Is stake locked for a period of time?

There are at least two problems if we make *return* linearly proportional to *stake* (i.e. *return* α *stake*). First, whales get too high influence on the system, even ones that are known bad actors. Second, to make any return, you may need to have a lot of stake to start. We can use reputation to balance this; see next section. But we can largely mitigate these effects via diminishing returns for more stake, or thresholds, such as the following.

Options for stake\_score, where *return* α *stake\_score*:

* *stake\_score* = *stake*. 10x more stake gives 10x more return. 100x gives 100x more.
* *stake\_score* = sqrt(stake). 10x more stake gives 3.2x more return. 100x gives 10x more.
* *stake\_score* = log10(stake). 10x more stake gives 2x more return. 100x gives 3x more.
* *stake\_score* = 1.0 if stake > thr else 0.0. That is, to participate at all you must give a minimum threshold “security deposit”. Threshold might be fixed, or cutoff at e.g. “top 100 stakers”.
* *stake\_score* = log10(stake) if stake > thr else 0.0. That is, have a minimum security deposit, plus rewards beyond that.

Usually there is a time period associated with staking, to avoid quick in-and-out attacks. For example, staking in Numerai and Decred is on the order of one month.

Stake can be put into the system in general, or different stakes to different places, such as a stake for identity, a stake for each dataset, a stake for voting.

We could consider risk to be another factor. The higher the risk, the higher the reward. Risk might be measured as stake (assuming the stake can be lost), or it might be a separate factor. For simplicity, we will lump risk in with stake.

**Tool: Identity**

We can prevent Sybil attacks and have other benefits if the system has a better handle on identity. We don’t necessarily need to force identity, we can simply reward for having better-defined identity.

For example: if I can demonstrate that I’m just one person (vs sybil-attacking) then I get rewarded, or at least not punished. For example, I have identity\_score = 1.0 if an E-resident (of any identity-granter that meets particular criteria, Estonia being the only current one), 0.9 if govt id (e.g. Authenteq), 0.75 if social-vetted (e.g. via Civic), 0.0 if no evidence.

**Tool: Actor Reputation**

Plain staking can still have issues. For example, whales can act badly, lose stake now and then, but keep participating with influence because they haven’t felt much pain for acting badly. To address this, we can include reputation. Or, new actors in the ecosystem may not have much capital, but if there are means for them to build up their reputation, then they can still do well.

We could combine actor\_reputation\_score with stake\_scores by summing or multiplying the scores.

Like stake, reputation score can account for reputation whales via e.g. sqrt(reputation), and so on.

Question: how to measure / define reputation? Here are some options.

* You get rated after every interaction with 1-5 stars. Both buyer and seller. Like Uber, Ebay, etc. The distribution / histogram of ratings itself can be useful information.
* My reputation = what others have staked in me. Like OpenBazaar “Trust is risk” framing. This is highly interesting, however untested and feels risky.
* “I have been active recently” and that’s all. This could be defined as: activity\_score > 1.0 for the past week, where I get 0.5 points for buying or selling data, 0.5 points for every day I run a keeper node, or 0.5 points for vetting someone.
* Your payment comes after a time period, after the consumer has used your data, and vetted that “yes this is not spam” or “yes this is ok”. Your reputation / karma is a function of whether payments came through.
* Reputation = total number of purchases made, regardless of price.
* Reward minimal activity. E.g. add 0.3 for each time user has spent > $10 by one party, up to 1.0.
* Reward being in system for minimal time and minimal use. E.g. add 0.1 for each month user has been in the system and had at least $10 in sales, up to 1.0.
* adChain style registry for “you are a good actor” vs not. Has staking for a fixed time, to see if there are challengers. If none, the actor is in a whitelist.
* Total sales (buying and selling) for that actor, over all time or a fixed period of time (one week, one month). Or, diminishing returns for higher sales, such as sqrt(sales) or log10(scales).

**Tool: Data Reputation**

An actor could build up a good reputation over time, then do massive sybiling with bad data. Having reputation on data can help. Approaches include:

* Data gets rated by buyers with 1-5 stars. Overall reputation is the average or the median. The distribution / histogram of ratings itself can be useful information.
* Data reputation = what the vendor has staked in the dataset.
* Data reputation = a confidence value given by the data vendor on how useful that dataset will be. Use it similar to how numerai uses confidence for models, i.e. reward is proportional to stake / confidence.
* Data reputation = what *all* comers have staked in the dataset. Whoever stakes in the dataset may get benefit on sales of that dataset.
* “This data has been used recently” and that’s all. This could be defined as: activity\_score > 1.0 for the past week, where it gets 0.5 points every time it gets purchased.
* Total sales for that data, over all time or a fixed period of time (one week, one month). Or, diminishing returns for higher sales, such as sqrt(sales) or log10(scales).
* Reputation = total number of purchases, regardless of price. This relates to a continuous token model where price goes up with more sales.
* Reward minimal usage. E.g. add 0.3 for each time it’s bought for > $10 by one party, up to 1.0.
* Reward being in system for minimal time and minimal use. E.g. add 0.1 for each month it’s been in the system and had at least $10 in sales, up to 1.0.
* adChain style registry for “this is not a spam dataset” vs not. Has staking for a fixed time, to see if there are challengers. If none, the dataset goes into a whitelist.
* Tokenizing each dataset itself, and letting the free market decide price; reputation is then basically price. This is exactly what will happen for fungible data, where we can have an exchange to let the market auto-price the data. It’s less relevant for non-fungible data.

**Combining Tools**

We can combine the scores of identity, stake, actor reputation, and data reputation together, for example overall\_score = stake\_score \* identity\_score \* actor\_reputation\_score \* data\_reputation\_score.

Multiplying scores has several benefits over adding them. First, makes the overall score less sensitive on any single score; it’s more about the order of magnitude differences. It balances individual scores on orders of magnitude, rather than a more sensitive summation, because log(a\*b) = log(a) + log(b). If there is a tradeoff among individual scores, it naturally biases to the “knee” of that tradeoff. Furthermore, it doesn’t need apples-to-apples units. For these reasons, this approach is common in AI and optimization.

If we also make each contributing score is in [0,1] then the overall score is in [0,1] as well. This can make it easier to reason about and include as a multiplier for block rewards. But it can add complexity depending what is being measured.

## 17.4. Short Time Scale Governance (Keepers)

**Introduction.** We have these governance sub-problems:

1. *Governance of network.* Answers: How are transactions voted on? Protocol changes? (Short and long time scales, respectively).
2. *Governance of Ocean Foundation.* Building community, managing funds, etc.

This document focuses on network governance. We explore explore short time scale governance first, followed by longer time scales.

The short time scale is about voting on transactions, on choosing & incentivizing keepers.

**Goals.** We ask:

* Do we address: How are active keepers chosen? Removed? They validate txs, but what else do they decide?
* Do we address: How do we incentivize keepers?
* Do we address: How to avoid token whales & sale whales dominating?

Recall that Ocean is delegated proof of stake (DPOS) running “keeper” nodes that are either active keepers (do validation, etc) or read-only keepers. DPOS limits to e.g. 100 total active keepers total, according to some criteria. For example, Cosmos lets the top 100 stakers be (active) keepers, and Bitshares uses one token one vote to elect the top 100 vote-winners as active keepers.

So that there *is* a network to talk to, we want to incentivize people / organization to run those nodes. Related goals:

* **Key goal: Are people / organizations incentivized to run keeper nodes?**
* Is there incentive to improve network performance? For example reduce latency, improve capacity, improve uptime

**Tools**

We don’t need to incentivize the read-only keepers, because we are incentivizing exchanges via referrals, and they are incentivized to have high volume. The read-only keepers are incentivized to turn into active keepers to get mining rewards.

We could reward based on keeper\_score, which can be a function of the keeper’s stake\_score, identity\_score, reputation\_score, and performance\_score. Performance\_score can be a function of latency, capacity, uptime, etc. We should also reward based on handling all transactions. Finally, we should take action to ensure keeper votes in line with other keepers (i.e. staying honest); one is to incentivize score, the other is simply to take their stake if they are out of line.

Alternatively, we could simply give equal payout to each keeper, but they must cross a threshold of stake, reputation etc in order to be a an active keeper in the first place.

We want to heavily penalize any downtime (1 - uptime). The formula *uptime\_score = 1 - exp(-115\*downtime)* works well. With 2% downtime it penalizes by 90%. The table below gives more values.

|  |  |
| --- | --- |
| **x: Downtime** | **y: Uptime\_score = exp(-115\* x)** |
| 0.01% (0.0001) | --> 1.0 |
| 1% (0.01) | 0.31 |
| 2% (0.02) | 0.10 |
| 3% (0.03) | 0.03 |
| 10% (0.1) | --> 0.0 |

## 17.5. Medium Time Scale Governance (for Bugs)

**Questions.** We ask:

* How is the protocol updated for bugs?
* How do we choose Ocean core devs?

**Tools.** Options include: chosen by other core devs; foundation signoff; both)

## 17.6. Long Time Scale Governance (for Protocol Updates)

**Questions.** We ask:

* How is the protocol updated for changes?
* How to avoid token whales & sale whales dominating protocol votes?
* How to balance speed of development with desire for decentralized control? (E.g. how to slowly hand control from Ocean core devs or foundation, to broader public?)

**Tools**

Tool: have a lightweight process to quickly let bugs handled, while have a longer process (e.g. 1-3 months) for controversial changes. For bugs, someone (Ocean Foundation, Ocean core dev team, other) can issue a patch, and make a secure recommendation to keepers to update. For longer- process, leverage one of the schemes below.

Tool: have simple majority (50%) for some decisions and super majority (66% or 75%) for bigger ones.

Tool: Give veto to some parties, at least for an initial time period. E.g. to foundation core devs or to foundation. Ideally there is a way to hand off control gradually.

Tool: liquid democracy. In any scheme, add in where a token holder can give control to another token holder to vote on behalf of his tokens. That control can be revoked at any time. Like Aeternity.

Tool: Grandfather in tokenized governance. Initially, core dev team has control but has a weekly feedback session with token holders. Replaced by tokenized governance within <=2 years. Like Maker. Issue: how is the core dev team governed? Issue: a technical bias to the interests of the core dev team (even if it is just temporary).

Tool: like above, but use the Foundation not the core dev team. The foundation has its own governance mechanism that can account for all stakeholders’ opinions, not just the technical considerations of the core dev team.

Scheme: One staked token one vote, to elect 100 keepers. Combines reputation and stake: people with reputation and no tokens can still get elected. Like Bitshares. Issue: token whales can control it.

Scheme:100 biggest stakers get to be miners or validators. Like cosmos. Issue: token whales control it.

Scheme:100 highest bidders get to be keepers for e.g. one year. Could be rolling elections, e.g. once every 3.65 days. Like VISA board of directors. Issue: token whales can control it.

Scheme: Pre-select an initial set of 20-100 keepers. Then, keepers are elected by other keepers, and also run nodes. Like IPDB. Therefore only keepers get keeper block rewards. Issue: too many rewards to a select “in” group. (This is not an issue in IPDB because it does not currently have mining rewards.)

Scheme: like above, but don’t give mining rewards. Instead keepers they just get reimbursed for their expenses, like IPDB. Issue: since no block reward, people are not incentivized enough to do things. Con: not as open to participate in co-running the network.

Scheme: Like Decred. With a “ticket” I can validate a block, and vote on protocol changes. I get a ticket by staking tokens *and* having enough hash power (combined PoS & PoW). The price of a ticket goes up if >400 tickets, and down if <400, such that there’s a roughly fixed supply of tickets. The ticket matures according to a distribution, but with minimum time 28 days. Protocol changes take 28 days to go through. Issue: super complicated.

Scheme: Use “tickets” for voting on protocol changs. To start, foundation owns ⅓ of tickets.Another ⅓ are obtained via tokens (decred style), another ⅓ by reputation (approach TBD).

Scheme: One identity one vote. Any participant can vote if id is vetted, stake > threshold, reputation > threshold.

Scheme: For each identity, a weighted vote. Weight = (id score, highest if e-res) \* (stake score, diminishing returns for more stake) \* (reputation score, diminishing returns for more reputation).

# 18. Appendix: Candidate Token Designs

This section contains a few candidate designs that we considered. The first focuses on data that is sold for a price. The second and third focus on commons data, with two riffs on what “free” means.

## 18.1. Candidate Design 1: Incentivize on Sale Price

**High Level**

The token is used as a currency for buying and selling. Data is priced in currency of vendor’s choice (eg USD or EUR) then converted just-in-time to a token price, according to crypto exchange rates (like Golem). The Ocean API includes support for each pricing scheme, such as fixed price (post for sale, buy at price) and auction price (post according to auction parameters, bid).

The block mining rewards incentivize keepers to run the network, and an ever-growing supply of data. *T* tokens are given out in a given time period. 20% of mining rewards goes to keepers (*TK = 0.2\*T*), 40% goes to sellers (*TS = 0.4\*T*), and 40% to referrers including exchanges (*TR = 0.4\*T*).

We split sellers from referrers so that there isn’t a zero-sum competition between sellers and referrers for block rewards. We also split apart keepers for similar reasons. Sellers are incentivized to make lots of good data available. Referrers are incentivized to refer good data of good data suppliers, for example having a beautiful exchange interface.

Mining rewards are dispensed daily, to average out time-zone effects of when markets are open.

Block mining rewards follow exponential decay with a half-life of 10 years. By the time relative mining rewards are small, the incentive to run keepers will likely be transaction fees, especially by exchanges.

Double-staking would allow Sybil attacks of over-leveraging the stake, adding extra risk to the system. Therefore, staked tokens can only be used in one place at a time, e.g. for selling a particular dataset.

**Sellers**

Let there be a given sale instance *i* of a dataset; it could be sold more than once. There is seller *j.* Here is how we calculate total tokens awarded to the seller in that period:

* tokens to seller *j*: *TS,j = ∑i (seller\_scoreij / ∑i seller\_scoreij)*

Where

* *seller\_scoreij = dataset\_sale\_price \* stake\_score \* seller\_identity\_score \* seller\_reputation\_score \* data\_reputation\_score*
* *stake\_score = log10(seller’s stake in system)*
* *seller\_identity\_score = 1.0 if e-res, 0.9 if govt id (eg authenteq), 0.75 if social (eg civic), 0.0 otherwise*
* *seller\_reputation\_score = 1.0 if in “seller” whitelist registry (adchain style), 0.0 otherwise*
* *data\_reputation\_score = 1.0 if in “data” whitelist registry (adchain style), 0.0 otherwise*

**Referrers**

If there is a referrer *k*, then:

* tokens to referrer *k*: *TS,k = ∑i (referrer\_scoreik / ∑i referrer\_scoreik)*

Where:

* *referrer\_scoreik = {0.0* if no referral; else *dataset\_sale\_price \* stake\_score \* referrer\_identity\_score \* referrer\_reputation\_score \* seller\_reputation\_score \* data\_reputation\_score.* This includes *seller\_reputation\_score* to incentivize referrers to use high-reputation sellers.
* *stake\_score = log10(referrer’s stake in system)*
* *referrer\_identity\_score = 1.0 if e-res, 0.9 if govt id (eg authenteq), 0.75 if social (eg civic), 0.0 otherwise*
* *referrer\_reputation\_score = 1.0 if in “referrer” whitelist registry (adchain style), 0.0 otherwise*
* *data\_reputation\_score = 1.0 if in “data” whitelist registry (adchain style), 0.0 otherwise*

The referrer only needs to make one overall stake; that stake is used to calculate the *stake\_score*, and for staking in the “referrer” and “data” whitelist registries.

**Short Time Scale Governance (Keeper Decisions)**

**Choosing keepers.** Anyone can download and start running a keeper node. To start with, it is read-only (passive). To become an active keeper, it must get voted active:

* Votes choose which keepers gets to be active keepers.
* Votes can be delegated to other identities; delegation can be removed anytime (liquid democracy).
* Who can vote: any participant who has an identity and a stake within the system. This can be keepers but also simple users of the system.
* Vote score by participant *i* = *vote\_scorei = identity\_score \* voting\_reputation\_score \* stake\_score* where
  + *identity\_score = {1.0 if e-res, 0.85 if govt, .., 0.0 if none}*
  + *stake\_score = log10(stake).* Therefore higher stake helps a bit, but not much. Basically “one identity one vote” with a bias to identity and more stake.
  + *voting\_reputation\_score = 1.0 if in “voting” whitelist registry, 0.0 otherwise*
* Vote tally for keeper *j* = *vote\_tallyi  = ∑participant\_i (vote\_scorei  if i voted for j, 0.0 otherwise)*
* Vote tallies are taken at the same time interval as block rewards are calculated and divvied up.
* Active keepers are chosen as the keepers with the 100 highest vote\_tally values

As soon as any of a keeper’s votes on transactions are out of alignment with the majority of votes, then the keeper loses their stake, and reward goes to zero. Another keeper automatically replaces them.

**Rewarding keepers.** We reward (active) keepers as follows:

* tokens to keeper *i*: *TK,i = keeper\_scorei / ∑j keeper\_scorej*

Where

* *keeper\_scorei = tx\_count\_score \* uptime\_score*
* *tx\_count\_score = log10(tx\_count* in time period*)*. Incentivizes for progress made in the system.
* *uptime\_score = 1 - exp(-115\*downtime\_percent)*. Therefore 90% penalty for 2% downtime.

**Medium & Long Time Scale Governance (Bugs, Protocol Updates)**

This is like the chosen (3rd) token design.

## 18.2. Candidate Design 2: Curation Market Pricing on Commons Data

**Actors Registry**

We start with an [adChain](https://medium.com/@AdChain/introducing-the-adchain-registry-cc5b8b831a7e)-style registry of vendors and referrers, which are “accredited as non-fraudulent by Ocean token holders.” (Like.) There are no further mechanisms for actor identity or actor reputation.

A new actor can stake a proposal to join; there is a vetting period in which challengers can come forward (with stake). If there are no challengers, the new actor is in. If there is a challenger and the majority vote “ok”, then the challenger loses some tokens and the new actor is in. If the majority votes “not ok”, the new actor loses their staked tokens.

**Free-as-in-Beer Pricing**

Different data pricing schemes sit on top, to handle different types of data: fungible, non-fungible, and free. Data reputation occurs implicitly within data pricing schemes.

For free data, we use a [Curation Market](https://medium.com/@simondlr/introducing-curation-markets-trade-popularity-of-memes-information-with-code-70bf6fed9881). The submitter of the data stakes 1 Ocean token. To use data, one must stake, as follows. The first person wanting to use the data the must stake 2 tokens to signal skin-in-the-game. The next person must stake 3 tokens, and so on. Each new staking acts as an upvote; the cost to stake is the reputation of the data. At any time, a prior staker can exit and get back T/N tokens, where T is the total tokens staked by all participants, and N is the number of stakers. These mechanics incentivize participants to submit quality data, and gives them an opportunity to make money too.

This means that overall, each actor has “holdings” in terms of stake (belief) of the relative value of different datasets. If an actor is early to understand the value of a dataset, they will get high relative rewards.

One might ask how pricing for this is free. The answer is: it’s free *as in beer*, where if you are participating in the community by submitting data or by upvoting high-quality datasets, then you are earning tokens that you can then use for other data. It’s a tit-for-tat on data, in a vein similar to BitTorrent for data blobs or iota for voting.

This mechanism also has a non-free path for actors that are not yet contributing members of the community. They simply buy Ocean tokens on an exchange, and stake to use the data. And by staking, now *they too* have joined the community.

It’s worth noting that Curation Markets do not need any block rewards mechanics; everything comes from staking. They also implicitly incentivize referrals.

Someone can only post data if they are the rights holder, or if the data is public domain / CC licensed. Anyone can challenge the poster’s claim at any point (with stake). There is a vote. If the majority votes “data ok”, the challenger loses staked tokens. If the majority votes “data not ok” then the poster loses their staked tokens (on this dataset), and gets removed from the ocean registry. They do not lose *all* their staked tokens; that would be too harsh.

But Curation Markets do not cover all cases. They’re less appropriate for truly fungible data. And, they may also scare off would-be sellers who are looking for more traditional pricing mechanics like fixed pricing, auction pricing, or royalties. For all these cases, we have a second mechanism where the token incentivizes data selling price.

**Non-Free Pricing**

This is like the chosen (3rd) token design.

**Short, Medium, & Long Time Scale Governance (Keepers, Bugs, Protocol)**

This is like the chosen (3rd) token design.

## 18.3. Candidate Design 3: Free Pricing on Commons Data

This candidate design is described in the main section of this paper.

## 18.4. Comparison of Candidate Designs

The table below compares the three candidate designs using the main questions as criteria. Let’s highlight the concerns with each design:

* The first design has strong incentives for non-free data, but takes no consideration for free data. Multiple registries hurt onboarding. Rewards are more complex, therefore prone to gaming.
* The second strongly incentivizes free data, but has higher barrier for new users getting popular free data. Also, charging to access commons data (even if can be viewed as tit-for-tat) will cause offline sharing.
* The third runs the risk of popular “free” (but stolen) data overtaking the system, i.e. the same fate as BitTorrent. The key to avoid this is to see the registry extremely well, so that it has a good trajectory.

In summary, the third candidate is our current preferred option.

|  |  |  |  |
| --- | --- | --- | --- |
| **Key Question** | **1** | **2** | **3** |
| **For non-free data: incentive for supplying more? Referring?** | ✔ | ≈ | ≈ |
| **For non-free data: good spam prevention?** | ✔ | ✔ | ✔ |
| **For free data: incentive for supplying more? Referring?** | ✖ | ✔ | ✔ |
| **For free data: good spam prevention?** | ≈ | ✔ | ≈ |
| **Does token give higher marginal value to users of the network, vs external investors? Eg Does return on capital increase as stake increases?** | ✔ | ✔ | ✔ |
| **Are people incentivized to run keepers?** | ✔ | ✔ | ✔ |
| **It simple? Is onboarding low-friction? Where possible, do we use incentives/crypto rather than legal recourse?** | ≈ | ≈ | ✔ |

# 19. Appendix: Token Design Full Checklist

We use the following as a thorough checklist against design candidates, especially as they mature.

## 19.1. Main goals

These are distilled from the goals below, and shortened / refined further.

* **For non-free data: incentive for supplying more? Referring?**
* **For non-free data: good spam prevention?**
* **For free data: incentive for supplying more? Referring?**
* **For free data: good spam prevention?**
* **Does token give higher marginal value to users of the network, vs external investors? Eg Does return on capital increase as stake increases?**
* **Are people incentivized to run keepers?**
* **It simple? Is onboarding low-friction? Where possible, do we use incentives/crypt (and maybe AI) rather than legal recourse?**

## **19.2. Buy/Sell/Incent Data**

* **Key goal: Is there an incentive for supplying more data?** Data of high quality? Sooner rather than later?
* **Key goal: Is there an incentive for referring others’ data?** Eg return on capital is proportional to stake.
* **Key goal: Is the token design sufficiently general to support our target pricing schemes, and other developer-defined pricing schemes?**
* **Key goal: Do we incentivize for supplying and referring free data (“data commons”)?**
* **Key goal: For free data, do we prevent spam?**
* For fixed-price pricing: Is there a means to post a fixed price? Agree to buy at that price?
* For auction pricing: Is there a means to post for auction? A means to bid in that auction?
* For exchange pricing: Is there a means to make an ask? Make a bid? Eg 0x protocol.
* For free pricing: who pays for compute resources? (Storage, write bandwidth, read bandwidth)

## **19.3. Token Economics**

* **Key goal: Do we address: How might the token increase in value?**
* **Key goal: Do we address: Which of the following rights does a token holder get? {Payment, access, profit or fee, contribution, block creation, governance}**

## 19.4. Towards Good Acting via Staking, Id, Reputation

Good acting general

* **Key goal: Is there a means to get high-quality metadata? Eg How do we prevent non-owners of the data from submitting that data?** (Fraud).
* **Key goal: Are we incentivizing skin-in-the-game? E.g. Does return on capital increase as stake increases?**
* Do big providers of data need to stake a lot? Consumers?
* If I have high stake but low reputation, can I make $? If I have low stake but high reputation, can I make $? If I have high stake and high reputation, can I make $$$?
* Do keepers (at least keepers with a higher level of reward or privilege) need to stake a lot?
* Is there a good threshold of individual / org identity - are they are who they say they are? At the very least, to prevent Sybil attacks. But potentially more, to adhere to data privacy regulations.
* Is there a good measure of individual / org reputation - are they a good actor in the ecosystem? (In buying, selling, keeping, etc?)
* Is there a good threshold of data identity - is the data what they say it is?
* Is there a good measure of data reputation - is the data useful?
* Does remuneration favor data freshness?

Good acting via staking

* **Key question: Does token give higher marginal value to users of the network, compared to external investors**
* Do we avoid double-staking?
* Is stake locked for a period of time?

## 19.5. Short Time Scale Governance

* Attack: keepers keep a block rewards for themselves. Addressed? Related: how do you prove that you are creating coins in a valid way?
* Do we address: How are active keepers chosen? Removed? They validate txs, but what else do they decide?
* Do we address: How do we incentivize keepers?
* Do we address: How to avoid token whales & sale whales dominating?
* **Key goal: Are people / organizations incentivized to run keeper nodes?**
* Is there incentive to improve network performance? For example reduce latency, improve capacity, improve uptime

## 19.6. Medium Time Scale Governance

* Do we address: How is the protocol updated for bugs?
* Do we address: How do we choose Ocean core devs?

## 19.7. Long Time Scale Governance

* Do we address: How is the protocol updated for changes?
* Do we address: How to avoid token whales & sale whales dominating protocol votes?
* Do we address: How to balance speed of development with desire for decentralized control? (E.g. how to slowly hand control from Ocean core devs or foundation, to broader public?)

## 19.8. Software / Complexity

* Is this the simplest thing that can possibly work?
  + Eg what if we got rid of anything fancy wrt identity, reputation, data rep, etc.
  + For each building block, what if we removed it, would the system still function?
* How is data storage paid for? Compute?
* Does this design work with BigchainDB?
* Can an MVP be implemented in reasonable time frame, with logical evolutions?
* Does it rely on 3rd party components that aren’t ready yet, and might not be ready in time?

## 19.9. Usability for Buyer / Seller / Referrer

* Have we walked through the sequence of onboarding steps for a buyer? Is it low friction to get going, with appropriate incentives to upgrade account? (E.g. simple -> complex identity)
* Have we walked through the sequence of onboarding steps for a seller? Is it low friction to get going, with appropriate incentives to upgrade account? (E.g. simple -> complex identity)
* Have we walked through the sequence of steps for a lone individual referrer? Is it low friction to get going, with appropriate incentives to upgrade account? (E.g. simple -> complex identity)
* Have we walked through the sequence of steps for an exchange-style referrer? Is it low friction for them to get set up? (E.g. copy-and-paste DEX software and licenses)

## 19.10. Attacks

* General: How do we ensure data-quality though validation?
* Attack: Self referral. Data supplier creates a second identity where they act as the referrer, and get the referrer bonus for themselves. Addressed?
* Attack: Referral buddies. Two data suppliers “refer” each other to get the referrer bonuses. Addressed?
* Attack: Data fraud. Someone sells someone else’s data, and getting paid. Addressed?
* Attack: Data spam. Someone posts or refers junk data and gets non-negligible block rewards for it, as bonus tokens or otherwise. Addressed?
* Attack: Sybil data Spam. Someone posts or refers lotsa junk data, and getting paid lotsa bonus tokens for submitting data. Addressed?
* Attack: Reputation overriding spam. Someone with high reputation posts some junk data, and gets non-negligible block rewards for it, as bonus tokens or otherwise. Addressed?
* Attack: Sybil sell tweaked data. Someone takes a dataset (owned by them or others), modifies it slightly, and re-sells as a unique piece. Repeats 100 times. Addressed?
* Attack: Sybil sell generative data. Someone takes a dataset (owned by them or others), builds a model from it, then draws >>1 samples from the model, and sells 100 new sets. This is problematic as no new real value. Addressed?
* Attack: Sybil signup bonus. Someone creates 100 new accounts and gets bonus signup tokens with each new account. Addressed?
* Attack: Sybil validation votes. Someone creates 100 new accounts and gets 100x more votes for validating txs. Addressed?
* Attack: Sybil governance votes. someone creates 100 new accounts and gets 100x more votes for governance decision. Addressed?
* Attack: Flash stakers. Someone briefly stakes a huge amount on a dataset, refers that data, gets huge block rewards, immediately un-stakes, and then repeats 100 times on other datasets.
* Attack: Rating buddies. Two data suppliers rate each other highly everywhere to increase returns. Addressed?
* Attack: Rating spam. Someone rates lots of data even though they haven’t used the data. Addressed?
* Attack: Byzantine ratings. Someone rates his competitor’s data badly so that his competitor gets less reward. Addressed?
* Attack: Sybil Byzantine ratings. Someone makes 100 accounts and rates his competitor’s data badly 100 times. Addressed?
* Attack: Fraudulent referrers. Someone claims to have referred data with “proof” but they’ve gamed the “proof”.
* Attack: bid sniping. In auction pricing mode, someone jumps in at last second. Addressed?
* Attack: people put illegal data (eg Equifax hacked data) onto Ocean. Addressed?

## 19.11. External Checklist

This is William Mougayar’s checklist, from [here](https://medium.com/@wmougayar/tokenomics-a-business-guide-to-token-usage-utility-and-value-b19242053416).

* Is the token tied to a product usage, i.e. does it give the user exclusive access to it, or provide interaction rights to the product?
* Does the token grant a governance action, like voting on a consensus related or other decision-making factor?
* Does the token enable the user to contribute to a value-adding action for the network or market that is being built?
* Does the token grant an ownership of sorts, whether it is real or a proxy to a value?
* Does the token result in a monetizable reward based on an action by the user (active work)?
* Does the token grant the user a value based on sharing or disclosing some data about them (passive work)?
* Is buying something part of the business model?
* Is selling something part of the business model?
* Can users create a new product or service?
* Is the token required to run a smart contract or to fund an oracle? (an oracle is a source of information or data that other a smart contract can use)
* Is the token required as a security deposit to secure some aspect of the blockchain’s operation?
* Is the token (or a derivative of it, like a stable coin or gas unit) used to pay for some usage?
* Is the token required to join a network or other related entity?
* Does the token enable a real connection between users?
* Is the token given away or offered at a discount, as an incentive to encourage product trial or usage?
* Is the token your principal payment unit, essentially functioning as an internal currency?
* Is the token (or derivative of it) the principal accounting unit for all internal transactions?
* Does your blockchain autonomously distribute profits to token holders?
* Does your blockchain autonomously distribute other benefits to token holders?
* Is there a related benefit to your users, resulting from built-in currency inflation?

1. oceanprotocol.com [↑](#footnote-ref-0)
2. bigchaindb.com [↑](#footnote-ref-1)
3. dex.sg [↑](#footnote-ref-2)