Towards Autonomous Organizations

[DRAFT]

Jubin Jose
hello@josejub.in
linkedin.com/in/jubin-jose-dev

Introduction

Autonomous Organizations are one of the near future promises of current technology and is a buzz word in the field of Decentralized networking and Blockchain oriented Smart Contracts. But when we look closer and analyze the details of it, we see that the technology requirement for engineering Autonomous Organizations is not only limited to aforementioned fields of technology but also many other fields including Artificial Swarm Intelligence, Internet of Things etc. We also see that, the terminology itself is not limited to closed organizations that are controlled by shareholders but also an open network of tiny organizations that are governed by simple rules written by individuals contributing to self identifying, communicating and organizing swarm of collective intelligence. We believe that Web 3.0 and advancements in Artificial Intelligence will be the greatest influencers of this. We observe that the progress towards Autonomous Organizations has already been started and will improve slowly and will win by adapting to and improving iteratively the existing technology, especially the Web. This approach will also ensure the gradual switching of end users without even knowing the technical complexities during the progress.

One of the main challenges that we face today is the lack of proper standards to achieve this goal of building the infrastructure. Currently being at the bleeding edge of all potentially contributing technologies, it is not possible to define a final standard for everything. So in this paper we are going to look into proposed, the high level building blocks of the infrastructure. And we will eventually break them down into smaller components by consulting current technologies and will suggest improvements to be made. We hope that each of them can eventually be standardised with the help of community and keeping open to the community.

Related Works

One of the oldest remarkable works in Multi-Agent Systems (MAS) is done by the Foundation for Intelligent Physical Agents (FIPA). Which formed in 1996 as an independent Swiss based organization later merged into IEEE CS standards organization. FIPA specifications were originally written for the Multi-Agent communications on the Web. Due to the eventual inactivity of the organization, the specifications are currently inactive and mostly outdated. However the specifications (which are made public and still bound to undisclosed patent restrictions) at its core provide enough inspiration to start building MAS related systems.

FIPA specifications cover three low level modules - Agent Communication, Agent Management, Agent Message Transport upon which a MAS can be built. Agent communication details the data structure of both the message being sent between two agents in a network and the shared Ontology between two agents in communication. A message being exchanged is the fundamental element of MAS which facilitates a Transaction between two agents. Ontology brings in context and background knowledge which enables self identification between two agents. Ontology plays an important role to bring meaning and mutual understanding between agents to accomplish a goal. Agent Management covers agent management services, agent management ontology and agent platform message transport. This specifies a model for the creation, registration, location, communication, migration and retirement of agents, which can be condensed down into an agent lifecycle supported by agent location service (Directory Facilitator). Agent Message Transport describes on-wire protocol for message exchange. We are not interested in Agent Message Transport now because it is outdated and there are better methods to replace it.

Recent advancements in efforts to build Knowledge Graphs over the semi / fully structured Web and Connected Devices led to two popular open standards - Schema.org (RDF is another popular standard) and W3C WoT specification. With which any agent (human or machine) can identify, interpret the ontology and contents of a web oriented site / app or a connected device and interact with it directly without a third party translation.

Schema.org solves the problem of ontology to a large extent for MAS. Adapting the Web to it will eventually create a planet wide, dynamic Knowledge Graph (it is like a database) that any agent can access at any time. It is up to the agent to decide which part of it to be accessed at a time (localize). Today, it is mostly available as embedded information within rich documents. The scope of Schema representations is not limited to documents but to the APIs as well, to make avail ontology description by an agent at a location in the Web.

W3C WoT on the other hand solves the problem of self identification (primitive ontology) and message exchange (Transactions) within a limited scope - electronic devices connected to the Web. It allows the agent to first retrieve the device information - to understand the capabilities of that device (ontology) and then read / update the device parameters or run actions on behalf. Mozilla's adaptation of this specification includes message exchange details as well.

Current self proclaimed Web 3.0 standardization efforts are clustered around blockchain technology. Two such notable projects are the DAOStack and Aragon. There are more projects out there exploring the same domain but we ignore them for now because, they either follow similar patterns or are in primitive stage. Both DAOStack and Aragon put effort to specify in detail - the internal workings of a DAO, but ignored the networking details by explicitly mentioning that there possibly coexist a mesh network of DAOs and agents. This suggests that DAO's can be built as independent nodes / agents in a network exposing a common interface for communication. This opens up a window of opportunity offering full freedom to choose multiple network communication protocols as well as the seamless integration with other nodes in the Web 3.0. This ensures the interoperability with diverse app ecosystem and upgradability from Web 2.0 to 3.0 without the end user even noticing it.

DAOStack is a framework (they call it an operating system for DAOs - which is true in a sense when Ethereum is considered as a Virtual Turing Machine) to build DAOs on Ethereum blockchain. It is built on Arc - which is an abstraction over a set of Smart Contracts (open library of governance modules and templates) implemented on the Ethereum blockchain. This abstraction underlies the governance systems decomposed into actions, schemes and global constraints that every agency can be built of. External agents are limited at the DAO boundary by only allowed to send inputs through Subscribed Schemes to influence an action through voting. This includes self modification of the DAO governance itself. Aragon is vary similar to DAOStack in it's architecture, where the governance is overseen by Aragon Court decisions.

Web by design decouples logic from data. Any logic can access any data as long as they are directly / indirectly reachable and have the right permission. InterPlanetary File System (IPFS) is a promising content addressing protocol that best match the decentralized nature of the Web. IPFS by design prevents data redundancy, permanent availability of data (as long as at least one node is serving it in a network), and offline first data distribution (enabling seamless communication within temporarily fragmented network - eventual consistency) through this concept of content addressing.

IPFS protocol is made itself transport agnostic with the help of libp2p module - which offers a drop in solution for Web 2.0 - 3.0 compatibility. libp2p brings a single network transport layer for the agents within Web to communicate over diverse transports through protocol negotiations.

IPLD is another notable module emerged out of IPFS, which tries to make avail all hash-linked data structures under a common data model. One us ecase of IPLD allows the agents in a network to explore and analyze public blockchains through a unified interface. IPFS already has received wide acceptance from the developer community for decentralized data storage and one such notable application is distributed key-value databases that further extended to accept SQL Queries.

Decentralized Identity (DID) management is another important dependency to build trust between Autonomous Organizations. A self explainable code is more trustworthy than any authority that issue IDs. Digital Identities will become an inevitable part of any organization, agent (human / bot) or device. A DID when created will associate zero credibility with it (which is different from real world IDs) but will eventually add up credibility and increased trust in the system. Associated credibility with a DID will exert varying influence over governed decisions made by Autonomous Organization from ID to ID - which at the collective level is the driving force to maintain the democracy within the Web to keep sustainability with it.

One notable effort in standardising the specifications (which is in the final stage to become official standard) for DIDs is taken by W3C Community Group. This specification highly relied on distributed ledger technology (blockchain). The DID registry is a distributed ledger to build a Distributed Public key Infrastructure. A DID is a text string composed of the URL scheme identifier, the identifier for the DID Method and the DID Method-specific identifier. This DID resolves to a DID document contains the context of that document, cryptographic authentication information for that DID and services that can be used to interact with the entity. DID support create, read/verify, update and deactivate operations on DID document.

Migration to Web 3.0 will be slow and steady. Currently, we have parts of technology that combined will give birth to the first generation of Autonomous Organizations in the near future. Web applications we see today will eventually become autonomous agents / services with right enough intelligence (even though it is not necessary). At the same time, innovations in Artificial Intelligence today is growing fast, that might add more organic behaviour to this collective ecosystem of Autonomous Organizations. What we should be doing right now is to establish a standard engineering path to properly combine what’s available and invent what else is needed to achieve the final goal.

High Level building blocks of Autonomous Organizations [edit in progress]

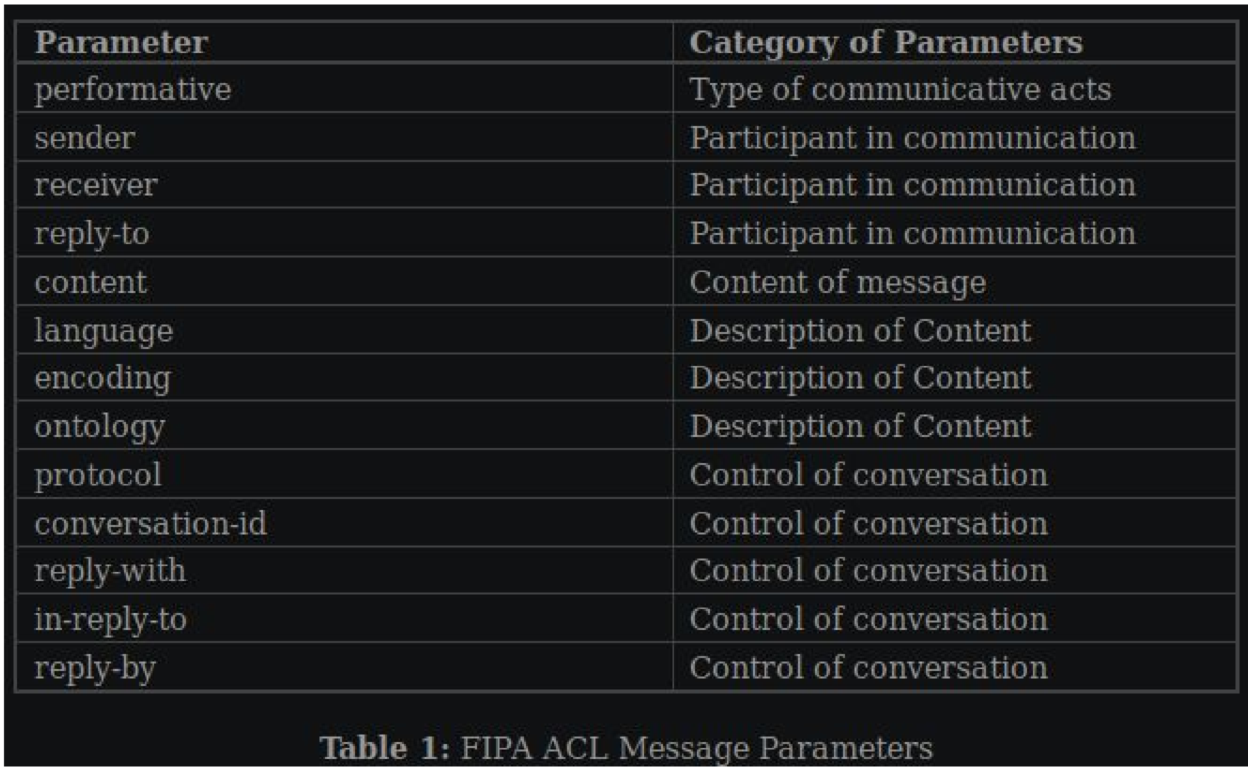
Autonomous organization of virtual agents:

- FIPA (Multi-agent communication standards)
    - Generalization: http://www.fipa.org/subgroups/ROFS-SG-docs/2007-TAAS-specifying-MAS.pdf
    - By spec. status: http://www.fipa.org/repository/standardspecs.html
    - By spec. subject: http://www.fipa.org/repository/bysubject.html
- Web of Things (agents identify others and communicate - API designs)
    - W3C standards: https://www.w3.org/standards/
    - Mozilla's adaptation API spec: https://iot.mozilla.org/wot/
- Categorizing the web (semantic representations (metadata) of objects and services for machine readability and understanding):
    - Google's schema.org: https://schema.org/
    - More: https://en.wikipedia.org/wiki/Metadata_standard
- DAOStack for Architecturing an agent (agency / service) & Aragon
- Libp2p & IPFS protocols for p2p communications, inter network communications and content storage
- Trust, Identity, Transactions (blockchains)
    - W3C DID: https://w3c-ccg.github.io/did-spec/
    - Hyperledger for permissioned blockchains: https://www.hyperledger.org/
    - Etherium for smart contracts: https://www.ethereum.org/
- Self modify code
- Decentralization, network latency and clustering
    - Universal protocol stack for decentralized communication: https://ipfs.io/
- Modern engineering (how to engineer, refine above standards for modern web iteratively)
    - Query one API endpoint (very important): https://graphql.org/
    - Couch replication protocol - eventual consistency: http://docs.couchdb.org/en/stable/replication/protocol.html
    - Conflict resolution by design - CRDTs: https://en.wikipedia.org/wiki/Conflict-free_replicated_data_type

Building blocks in detail [edit in progress]

FIPA Agent communication specs

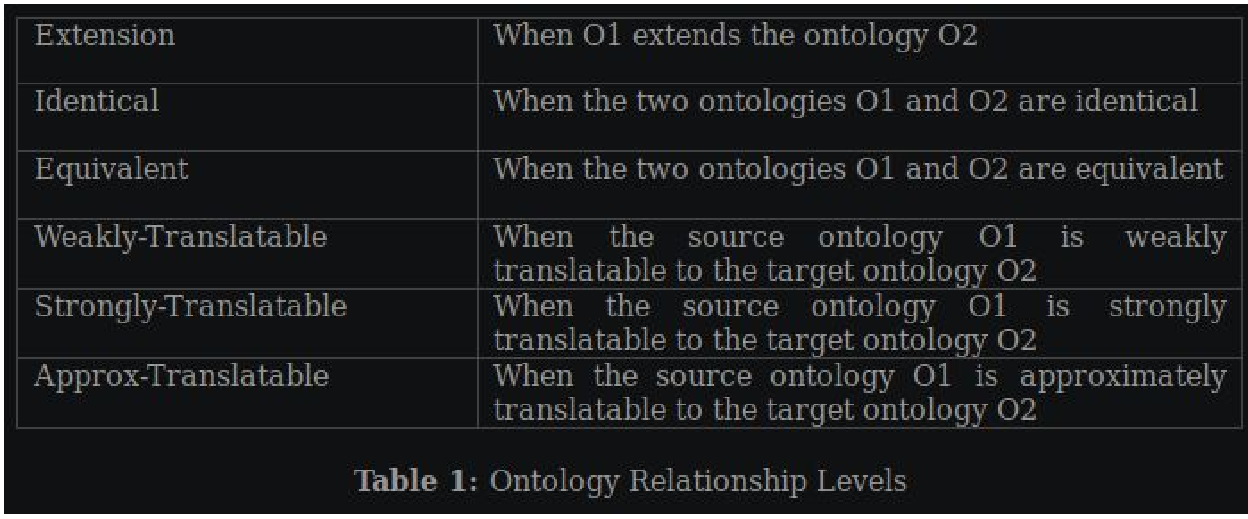
- Message structure spec

- 

Parameter	Category of Parameters
performative	Type of communicative acts
sender	Participant in communication
receiver	Participant in communication
reply-to	Participant in communication
content	Content of message
language	Description of Content
encoding	Description of Content
ontology	Description of Content
protocol	Control of conversation
conversation-id	Control of conversation
reply-with	Control of conversation
in-reply-to	Control of conversation
reply-by	Control of conversation

Table 1: FIPA ACL Message Parameters

- Ontology service spec (refers to scheme.org or WoT description)
    - Agents that communicate together should be aware of the Ontology - capabilities of each of them
    - Ontologies of agents can be related in multiple ways

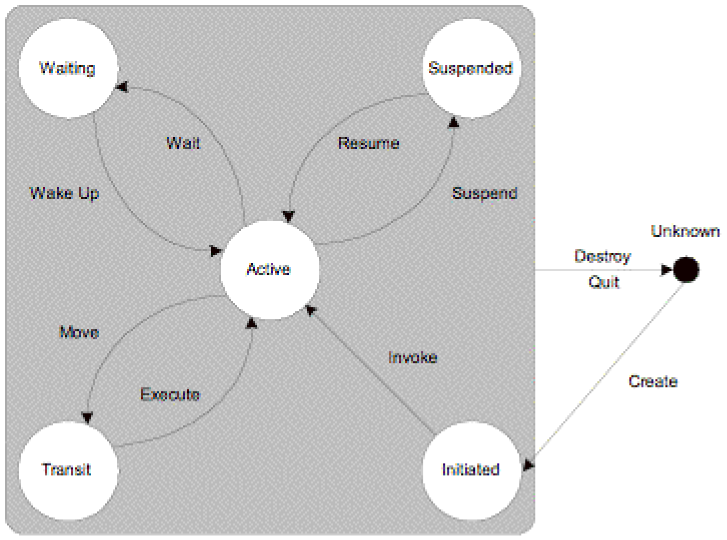


Extension	When O1 extends the ontology O2
Identical	When the two ontologies O1 and O2 are identical
Equivalent	When the two ontologies O1 and O2 are equivalent
Weakly-Translatable	When the source ontology O1 is weakly translatable to the target ontology O2
Strongly-Translatable	When the source ontology O1 is strongly translatable to the target ontology O2
Approx-Translatable	When the source ontology O1 is approximately translatable to the target ontology O2

Table 1: Ontology Relationship Levels

    - It is common and good engineering practice to build a new ontology by extending or combining existing ones.
    - Translation between Ontologies is necessary in this case. More: http://www.fipa.org/specs/fipa00086/XC00086D.html

FIPA agent management spec

- Covers: agent management services, agent management ontology and agent platform message transport
- It establishes the logical reference model for the creation, registration, location, communication, migration and retirement of agents.
- Directory Facilitator - a directory listing for agents with UID and nickname
    - API: register, deregister, modify, search, get-description agents
    - Federated Directory Facilitators: network of DFs
    - 

agent lifecycle

FIPA Architecture

- http://www.fipa.org/specs/fipa00001/SC00001L.html
- Agents & services - core components. A service can be an agent or an RPC function
- Agent & service directory service (indexing) Data structure: [name, type, locator]
- Agent message structure & encryption

FIPA Applications & Examples

- http://www.fipa.org/repository/applicationspecs.php3
- Ontology is defined in action level. Do we need an Ontology defined this way? Or in a very basic form like Schema.org?

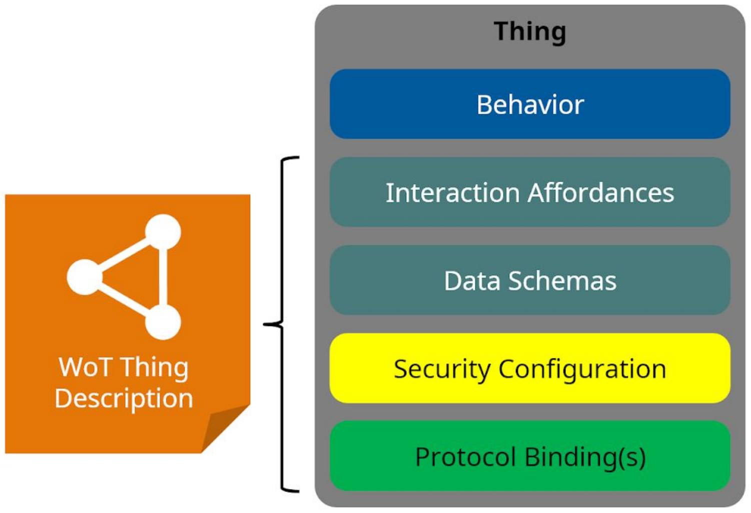
W3C web thing description

- Property, Action, Event - Interaction affordances
- Each Interaction affordance is located with a URI
- Description of a thing: https://www.w3.org/TR/wot-thing-description/#thing

W3C web thing architecture

- A web thing identifies itself with thing description

- A thing can be standalone or linked with other things. The thing description will contain links to others if there's any. There can be terminating or intermediate things as well. (https://www.w3.org/TR/wot-architecture/#sec-architecture-overview)



-
- The Interaction Model of W3C WoT introduces an intermediate abstraction that formalizes the mapping from application intent to concrete protocol operations and also narrows the possibilities how Interaction Affordances can be modeled.
- Each interaction model in detail: https://www.w3.org/TR/wot-architecture/#sec-interaction-model
- Communication patterns between things: https://www.w3.org/TR/wot-architecture/#sec-wot-servient-architecture-high-level

Mozilla WoT

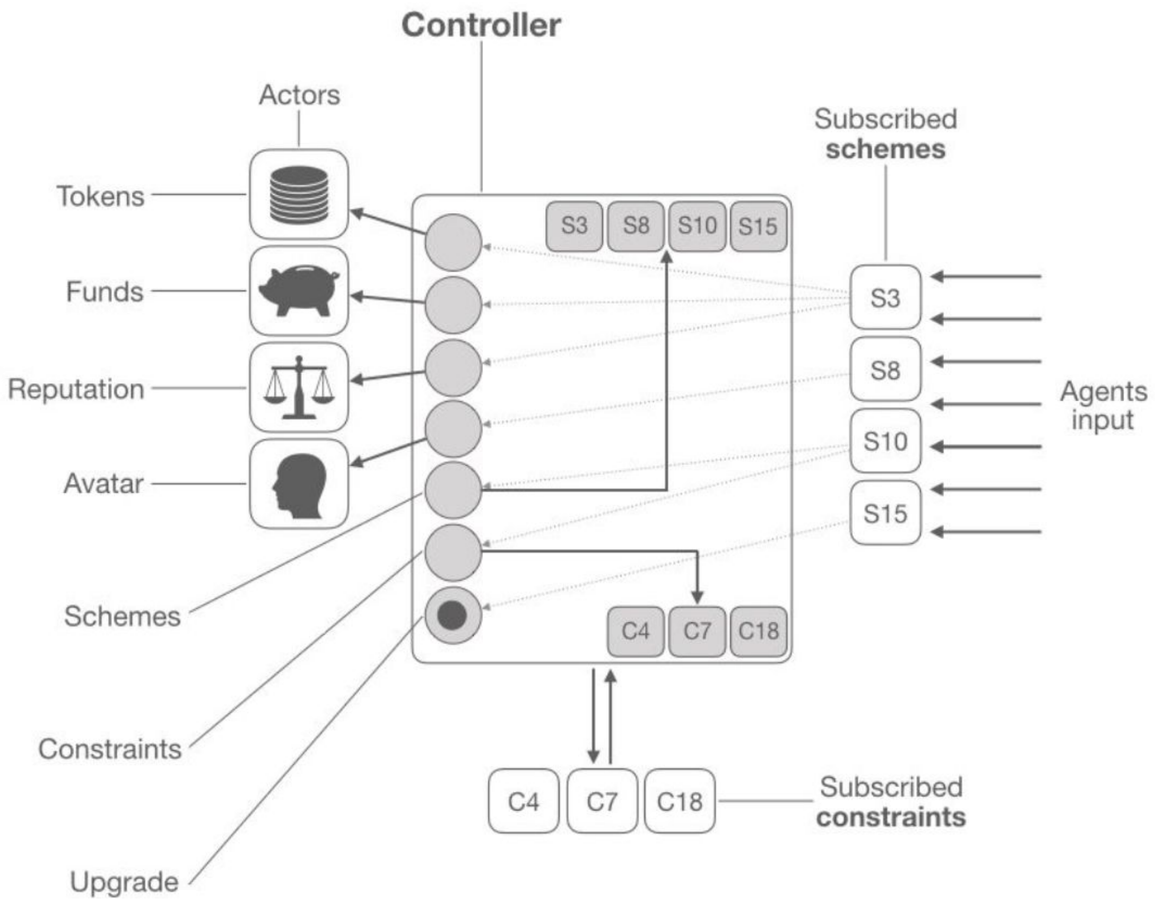
- Contexts: http://iotschema.org/, https://iot.mozilla.org/schemas

DAO Stack

- reduction of coordinational cost
- A DAO is a new form of scalable, self-organizing cooperation, that is operated by smart contracts on the blockchain
- The building blocks of DAOs are smart companies, or agencies
- smart agency is an atomic governance unit that is managed and operated with smart contracts on the blockchain
- It has its own token (related to benefits of the company's resources), its own reputation systems (related to credibility and influence in company matters), and its own governance system (its "bylaws" encoded in smart contracts).
- A DAO is a centerless mesh network of agencies, which is also an agency in itself
- DAO Topology

    - assembly mode: a large number of agents are interacting in decision making within a single agency via its smart contract, assuming that reputation, and thus decision-making power, is fairly distributed
    - fractal federal governance mode: In the extreme fractal federal-governance mode of a DAO, the DAO is an agency with a a few agents, each of which is itself an agency with a few agents, each of which is itself an agency, and so on and so forth.
    - complex mesh-network mode: a nested mesh network of agencies, interwoven with other DAOs through shared agencies
- Blockchain Governance
    - Agency is the basic unit of governance over the blockchain. A blockchain governance system is a state-transition function, that collects inputs from blockchain addresses and under certain rules spells out an output in terms of the blockchain global state transition.
- Agency functionalities
    - Token distribution: distribute its own native tokens (utility tokens, share tokens) to contributors of value, as valued by the organization. The issuance of native tokens enables the organization the creation of its own separate economy
    - Funds allocation: An organization can earn, or collect via its own-token sale, external tokens such as ETH, GEN or other DAO's tokens. It can keep them in reserve, and distribute to third parties in exchange of a particular effort or contribution.
    - Reputation assignment: Each agency can assign reputation scores to its members. Reputation is not transferable. It is awarded to or earned by specific members, according to their merits and contributions made to the organization.
    - Collective data curation: An organization can manage its own collective databases of objects, and maintain their curation.
    - External activity: An agency can act within another agency as a single entity. For example, an agency can submit a proposal inside another agency (or DAO), and vote on others' proposals.
    - Governance upgrade: Each organization can configure and update its own governance system. By approving or removing certain elements.
- Schemes: Schemes are logical functions made of a series of instructions that take a particular set of inputs, and process them to generate a particular set of outputs.
- Global Constraints: Global constraints are specific conditions that can be attached to a particular agency or DAO and that will limit its functionalities.
- The DAOstack ecosystem is made of a multitude of distinct but interoperable DAOs, interacting with one another in order to maximize the potential benefit of open and distributed collaboration.

- all DAOs are made of a series of smart contracts, deployed through Arc : a Solidity framework of governance DAOstack allowing to create, configure, deploy and operate DAOs onto the Ethereum blockchain possibly relying on IPFS as an overlay network for data storage and retrieval.
- People can interact with these DAOs either directly, through the execution of blockchain transactions, or indirectly, by relying on a particular front-end to the underlying blockchain ecosystem.
- Alchemy is the collaborative DApp developed internally by DAOstack, enabling anyone to create a new agency or DAO and start collaborating with others in the DAOstack ecosystem.
- Arc is a general governance framework for an interacting internet of blockchain agencies, the basic operating system for DAOs. It is an open-source, modular and general-purpose framework by design, and it comes with an open library of template governance modules, or elements, that will evolve by the needs of its users.
- Arc elegantly implements in smart contracts the basic decomposition of governance systems with: actions, schemes and global constraints that every agency can be built of.
- The only interface for agents to interact with the agency is via its subscribed schemes. Each scheme comes with its specific "knobs" (functions) which can be operated by external agents (blockchain addresses) via transactions that call and operate these functions.
- The controller: The controller contract is the main engine of the agency. It is "owned" by and gets commands from the subscribed schemes alone, which operate its functions. Via its functions its sends commands to the actors contracts: the token and reputation printers, the funds wallet and the avatar, as well as the upgrading functions of the governance system and the technical architecture itself.
- The avatar [identity management] - is the "face" of the agency (and its address), which acts externally and is capable of doing — via the help of particular schemes — anything that can be done on the blockchain. In particular, it can participate as an agent in other agencies — e.g. submit and vote on proposals — on behalf of its agency. It is also the identity to which reputation in other agencies is assigned.
- Self modify:
    - Protocol upgrade:
    - Technical upgrade:



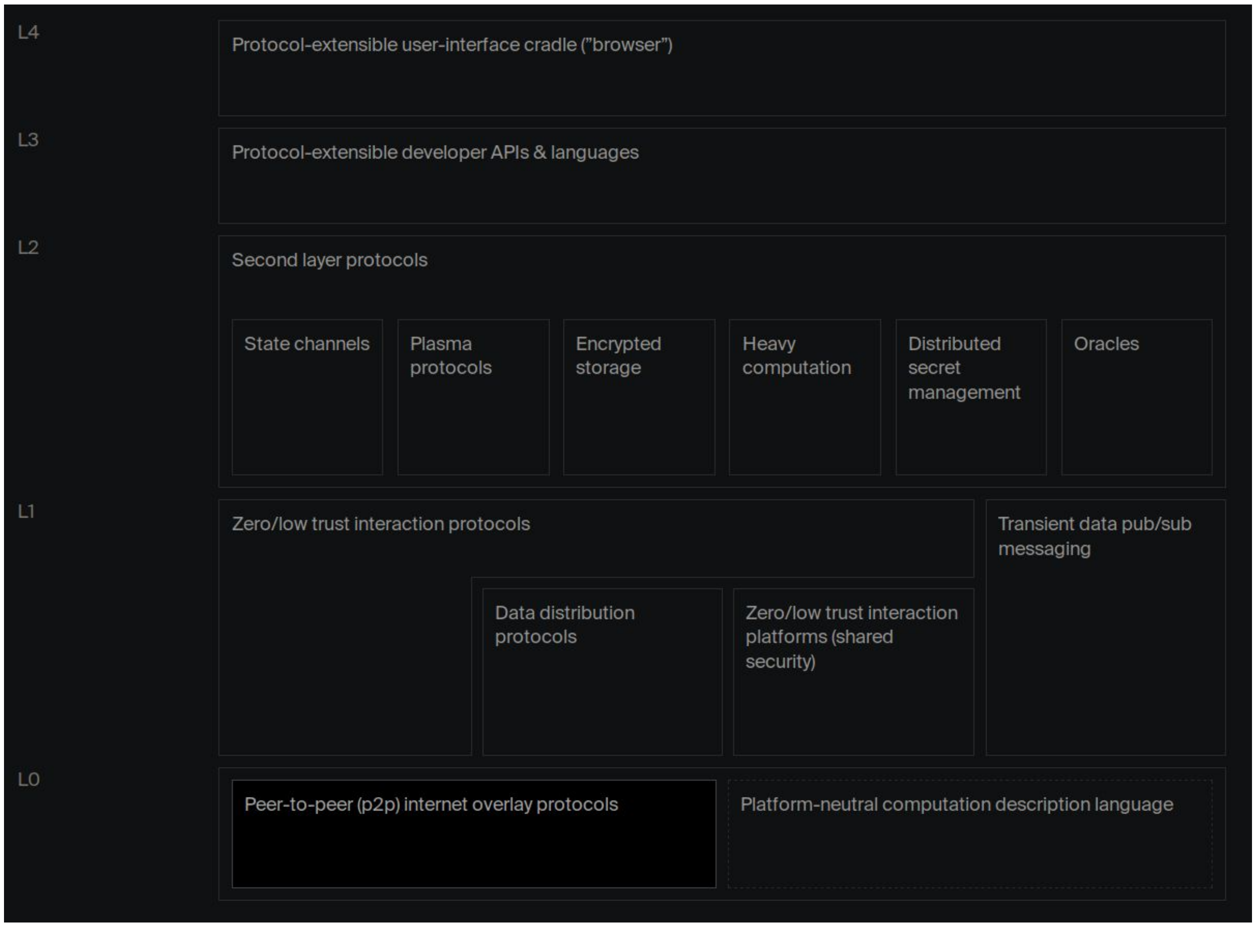
-

Aragon

- The Aragon Network is an Aragon organization that provides infrastructure and services to users of the Aragon platform
- governed by ANT holders
- (similar to DAOStack) Each Aragon organization exists as a set of smart contracts that define the organization's stakeholders and their associated rights and privileges
- some rights and privileges require subjective constraints that cannot be encoded in a smart contract directly - Aragon Court is a decentralized oracle protocol developed and maintained by the Aragon Network - to resolve subjective disputes with binary outcomes. It enables an organization to create Proposal Agreements that define subjective constraints on an organization's operation and can be enforced by minority stakeholders.

Web 3 Foundation

- Web3 stack:


L4
Protocol-extensible user-interface cradle ("browser")
L3
Protocol-extensible developer APIs & languages
L2
Second layer protocols
State channels
Plasma protocols
Encrypted storage
Heavy computation
Distributed secret management
Oracles
L1
Zero/low trust interaction protocols
Data distribution protocols
Zero/low trust interaction platforms (shared security)
Transient data pub/sub messaging
L0
Peer-to-peer (p2p) internet overlay protocols
Platform-neutral computation description language

Matrix Message Protocol:

- Matrix defines a set of open APIs for decentralised communication, suitable for securely publishing, persisting and subscribing to data over a global open federation of servers with no single point of control.
- Uses include Instant Messaging (IM), Voice over IP (VoIP) signalling, Internet of Things (IoT) communication, and bridging together existing communication silos.
- provide an open decentralised pubsub layer for the internet for securely persisting and publishing/subscribing JSON objects.
- Matrix defines APIs for synchronising extensible JSON objects known as "events" between compatible clients, servers and services.
- It's focused on eventual consistency - synchronization of events (JSON objects)
-

User experience and Value generation ecosystem [edit in progress]

- User experience
  - A virtual assistant per person
  - No more knowledge aggregation in one place - like Alexa, Google Assistant, Siri but a swappable assistants, like you can use any web browser to get the same experience
  - Any capability of Assistant is plug-in - which includes AI (cognitive) services
- Privacy
  - Data stays on device
  - Assistants see data on device and behave dynamically (intelligence downloaded to device)
- Value and ecosystem
  - Addons generate value
  - DAO Search service, app store, AI / ML services