Blockchain's Dual Role: Protocol Agreement and Collective Recommendation Engine

Ara: a Framework and Implementation That Realizes Web3.

https://www.ara.foundation

Table of Contents

Executive Summary

by Medet Ahmetson medet@ara.foundation, 2026

A decade of blockchain development , yet the promised Web3 never materialized.

Medet Ahmetson has been developing this framework since the early 2010s. As former CTO of Blocklords,[1] he observed firsthand how blockchain's structural limitations prevent it from becoming a platform for users and developers.

This whitepaper is the result of the accumulated analysis. It identifies five reasons why blockchain has not become Web3. Then, provides a framework that redefines **Web3 as a creative workspace built on collective wisdom**.

The five reasons are grouped into three categories. The first category is a problem endemic to computer science: treating computers as information technology rather than software platforms. The second category identifies two architectural gaps: blockchain's isolation from the existing internet, and an incentive structure that serves miners and holders rather than builders. The last category draws on lessons from the rise of tech giants that blockchain development has ignored.

The solution does not dismantle existing internet infrastructure or existing blockchain ecosystems. It adds the missing components, linking blockchain and the existing internet into a coherent decentralized network.

The proposed framework is consists of the three components: Arada, a blockchain acting as a semantic protocol layer; Aramak, a decentralized recommendation engine built on collective user behavior; and Maydan, a desktop shell that gives users direct ownership of their software environment.

The first implementation targets the project management and open-source software sustainability using blockchain as the semantic linking layer.

[1] Building crypto games since 2018.

Part I: Problem Statement

Decentralization as a solution has been a foundational instinct among hackers since the early days of computer science. There wasn't a viable solution until the emergence of Bitcoin in 2008. A peer-to-peer network using cryptocurrencies to support the network without central authority reignited interest in decentralization.

In 2013, Vitalik Buterin then further proposed that blockchain, a technology underlying Bitcoin, could serve purposes beyond programmable money. Ethereum sparked the second wave of the blockchains, as world computers where anyone could deploy and use decentralized applications. Gavin Wood, another early contributor, coined the term Web3 to describe this vision of the internet's future.

Since the launch of Ethereum in 2015, a decade has passed, but we don't see the rise of the promised decentralized future that could be solved by the blockchain.

There are five reasons why blockchain has not yet become Web3. The reasons could be grouped into three categories.

The first reason is a problem endemic to computer science as a field: Internet and Computers are treated as Information Technology rather than as software platforms.

The second category is the fundamental error in the blockchain architecture. This category is composed of the second and third reasons. Blockchain is missing the core of the web, the links in its design. The third reason is that the network incentivizes the miners and token holders, treats self-sufficiency as the goal, ignoring its target audience: the dapp makers.

The third category is the failure to learn from the rise of the tech giants since 2008. It includes reasons four and five. The user experience of the apps is equally important as Data, and Software. Lastly, blockchain misses a discovery mechanism that serves user intent rather than capturing user attention.

To make Web3, we need to take the lesson from the tech giants and create competitive solutions. These solutions argued below do not dismantle the existing blockchain infrastructure or the Internet. They add a layer that binds both into a coherent decentralized web.

First reason: Data point-of-view versus software point-of-view

Computer Science has institutionalized itself under the label Information Technology (IT). Alan Kay stands as a rare exception, arguing that researchers fixate on static data rather than on what

software can do. Blockchain developers inherited the same bias, calling it a decentralized public ledger[2].

Computers are not about storing or exchanging data. Their core is the execution of instructions. Thinking in these terms reframes the computer as a problem-solving instrument, pointing naturally toward AI, and richer human interaction. The history of the Web demonstrates this paradigm in practice: most proposed directions, identified problems, and offered solutions never found a real audience because they approached the web through the data lens.

Historical lesson of the Web evolution

Tim Berners-Lee invented the Web in 1989 to link research papers at CERN. With the organization's permission, he released it into the public domain.

University students saw the potential beyond knowledge sharing, building online versions of the existing offline services. These online services made the web rapidly popular.

Seeing the unexpected rise of the Web, the thinkers attempted to map its trajectory through a data lens, popularising the Web 2.0 as the era of user-generated content within social networks. According to them, if the original Web was for read-only, Web 2.0 is read-write era.

Tim Burners-Lee rejected this framing noting the Web was designed for read-write from the start and was already planning its next stage. He argued that data should be semantically structured to be readable by both humans and machines. The thinkers embraced it, labelling it Web 3.0.

The semantic web, despite years of effort, never achieved mainstream adoption. Web ontology remains the domain of a few research groups, and the idea of Web3 as a semantic layer has been quietly shelved.

Further development

Tim Burners-Lee, dissatisfied with the Web's centralization, began working on a remedy. His next project, Solid, framed the solution as read-write-own, still through the data lens. Around the same time, Gavin Wood coined Web3, addressing the same ownership problem from the blockchain side.

The Web's actual trajectory becomes clearer when viewed through the software lens: the rise of SaaS, IaaS, and PaaS and ultimately cloud computing which the data lens never anticipated or explained.

[2] Cosmos defines blockchain in their documentation as a deterministic state machine, a more precise definition than the public ledger framing.

We must approach computers and the internet through the software lens. This shifts helps us to identify root causes rather than treat symptoms.

Second Reason: Missing the core component of Web

For security reasons, blockchain operates as an isolated network. It excludes the web's core architectural element: hyperlinks.

Dapps implemented as smart contracts, exist entirely within the blockchain platform. Workarounds such as oracles add complexity and cost, making the technology impractical for dapps. Another trajectory popular among blockchain is rewriting the entire web stack around a single network as the base layer. In both cases, a dominance of a single currency and network hands disproportionate power to miners and large holders, undermining the decentralization blockchain claims to deliver.

—

Viewed through the software lens, we propose a new blockchain that from onward we will name it as blockchain A. It's a complementary alongside existing blockchain with programmable money, from now on called blockchain PM. Our blockchain acts as a protocol agreement for a specific domain.

In doing so, it connects blockchain PM to the rest of the Internet without the web stack to be rewritten. Secondly, it positions the blockchain network as a semantic layer connecting end users to each other.

Third Reason: Value Capture goes to holders rather than creators

When blockchain is used as an application platform, demand for its native coin rises, redirecting incentives away from builders toward holders. It becomes a self-sufficient network that rewards holders and node operators, leaving no room for builders. After all, popularization becomes the task of builders, while a self-sufficient network makes it hard to achieve.

To address this, we propose a new architecture. This architecture separates the developers' incentives[3] from the network incentives.

Rather than a single coin, each node issues its own cryptocurrency representing its reputation within the network.

[3] For payments between developers, Ara uses the blockchain PM

The node based cryptocurrency makes the network topology discrete, with each node carrying its own incentive structure centered on fiat onramp. To have their coin to be accepted among other nodes, each node must bring users and value to the network.

Blockchain PM plays an additional role here: it supplies collateral to prevent network spam. A new node could provide as a collateral either well-reputed node's coins, ARA or a fungible token within blockchain PM. It's then used if the node was adversary.

Following certain rules, the network's alchemy process converts highly reputable node coins into ARA tokens. These alchemized tokens are supported at the protocol level, allowing to use tokens without additional collateral or if node loses its reputation along with its custom coin.

We expect ARA coins to be difficult to produce, and anticipate nodes dividing along geographic, political or ideological lines[4].

Fourth Reason: No incentives for UX/UI

Blockchain development has consistently neglected UX and UI. Apple's history is proof of why it's vital for the future of the Web.

In the early days of personal computers, manufacturers built their own hardware, software and operating systems. Among all early manufacturers, Apple is the only one that survived and grew. Because it treated UI and UX as a core design commitment, not an afterthought. Current blockchains ignore this entirely, and provide no incentive structure for UX and UI development.

We propose two things. First, a client environment[5]. This environment, which we call Client E, runs programs and fetches semantics, giving users a flexible automation and customization. Secondly, it supports UX and UI closely aligned with the semantics stored on blockchain A.

Fifth Reason: No discovery

As blockchain becomes a platform, it needs a proper distribution and discovery mechanism for its users. The rise of the tech giants over the last two decades demonstrates this.

Steve Jobs, having watched Microsoft win the OS wars through developer ecosystems, designed the iPhone for developers as much as for the end users, giving them tools to build and distribute apps. The App Store created a radically new industry of mobile development. Android succeeded

[4] For example, some crypto maximalists will accept and advocate for collaterals using wrapped BTC or ETH. Others will operate within legal frameworks.
[5] Ethereum's original vision of a dedicated browser, realised briefed in the abandoned Mist project, pointed in the right direction

where Windows Phone and Nokia failed because it launched with the developer marketplace from the start.

Recommendation systems are the other reason platforms became tech giants, driving the rise of YouTube, Netflix, Amazon, Instagram, Spotify and TikTok. Data became valuable primarily as fuel for recommendation systems.

—

For blockchain to fulfill the Web3 vision, it also requires a decentralized recommendation system, one that serves user intent rather than business-driven algorithmic incentives. We propose calling this component blockchain B, a discovery layer operating alongside blockchain A and blockchain PM. Built on user's collective behavioral data and open to anyone to extend, such a system remains impossible to replicate regardless of how much capital or AI corporates or centralized institutions deploy.

Summary of why blockchain is not Web3

Drawing from the history of computing and the rise of tech giants, we identified five structural gaps in the current blockchain architecture.

Blockchain A acts as a protocol agreement between users and developers. Their connection on the network is shaped by the semantic schema defined by the nodes.

This semantic base and moving payments outside the network addresses the second reason by linking users to the broader internet alongside the crypto network.

The discrete topology of the network, with nodes acting as the gatekeepers ensures that they are incentivized toward wide adoption addressing the third reason.

Client E combined with a semantic layer, addresses the fourth reason by making UX and UI a first-class component of the protocol.

Blockchain B, the decentralized recommendation system in tandem with the previously described components addresses the fifth reason.

Throughout, we apply the software lens rather than the data lens. In this design, we move computation to users' computers, while the network acts as the communication layer, accumulating collective knowledge for creativity.

Part II: Solution

Data centric point of view has already produced its convenience products and achieved its goals. Since 2015, it has been hitting its limits. It's now possible to create any form of data, be it music, art or a game, it's possible to share as well. There is only a tradeoff between privacy, speed and convenience, rather than ownership or data centralization[6].

To illustrate, the whitepaper itself was written in Google Docs. The convenience outweighs the privacy tradeoff.

The real problem is not where data is stored, but who controls the execution environment, or simply lack of: software ownership.

By owning the software we mean two abilities for users:

- the ability to extend or compose app functionality beyond what the developers intended,
- the ability to customize what appears in the UI[7].

Both are technically achievable through plugins offered via marketplaces or scripting languages provided by the large applications. Yet these still constrain within the boundaries of the application API.

The user has no control over Google Docs to personalize it, and software ownership would change that. To illustrate: in an alternative document processing application registered on the Ara network, the app could integrate an AI tool, automating paragraph-level fact checks. Making the writing more personalized yet faster and smoother.

The combination of a blockchain A, blockchain B, Client E, and Semantics addresses this challenge directly. It makes the global Internet at one application domain operate as a coherent living organism.

The framework that connects software semantically for software ownership has two long-term effects. The first is refactoring over existing software to decouple it into MVC pattern: data sits on the client side, the view is separated from the logic. The second is turning a software into a craft rather than an ecosystem. Each application does one job well, while users pipe applications together at the computer screen level by semantics, that over time due to AI, and recommendation engine gets more and more sophisticated.

---

[6] People use the messengers and popular social media for daily chats. But for sensitive data, use the alternatives such as Secret Chats in Telegram or Signal.
[7] The ability to control the UI means being able to modify the interface by adding, hiding or rearranging elements according to the users own needs

Web3 redefined

The software ownership with the semantic layer, recommendation engine, and decomposing software into MVC modules leads to the powerful decentralized internet. **Web3 is a creative workspace built on collective wisdom**. The creativity at a global scale opens the door to applications no single team could produce alone. The killer dapp is not a blockchain but the applications users composed for themselves.

The appendices demonstrate how this framework could be implemented to various domains. Below is the first implementation.

Ara: Project management & Open-source software sustainability

Ara, as the first implementation of the framework[8], targets the intersection of open source software and project management. Technical details are scoped to personal computers.

All three (personal computers, project management and OSS) are foundational to daily computing and among the most frustrating problems to address. Yet combined through Web3, they resolve each other.

Currently, project management satisfies no one. Apps either try to be all-in-one, or remain completely siloed from each other. Every attempt to solve coherent project management with minimal user input adds yet another tool to an already crowded stack.

Open source software is modifiable by nature, which suits project management directly. It's easier to assemble a stack of open source softwares on Client E personalized well for specific work style in an organization.

On the other side, despite building an entire modern tech foundation, open source projects are underfunded, and their maintainers burn out. Sponsorship implies responsibility most maintainers don't want, and carries a psychological resistance: it feels like commercializing something that was never meant to be a business. Maintainers who need funding must also develop business and marketing skills, which pulls them away from the work itself.

In Ara, a node access for users is based on the subscription model. The subscription fee is split partially as a node charge, and part distributed to the open source projects used by that user. This node's fee is to curate the network: remove harmful or low-quality OSS, and replace them with the better alternatives, and continuously onboard open source alternatives to proprietary apps.

[8] Ara framework: dual blockchain with execution environment, and semantics as a shared data.

Registering open source projects brings two benefits for developers. Being semantically registered makes their work discoverable and usable automatically. Second, due to semantics, seeing how people use parts of app features, it drives reduction of unnecessary code, making their projects more modular and refined.

In Ara, the execution environment is represented as a Desktop Shell. It includes a built-in browser. The web apps registered on the network, Maydan opens as a windowless group of data. Window controls[9] are replaced by meta tools built into the shell itself.
Users can open multiple URIs, and bind them together by the custom semantics and custom personalization.

Some groups of components are better represented as spatial 3D data as it reduces data intensity. For example a spreadsheet with a country list could be replaced by a spatial 3D globe. Or a project's architectural diagram in UML could be represented as an isometric model of data flow.

Spatial data along with the software ownership, makes Ara share the spirit of Hypercard or Macromedia Flash. Project management becomes a creative workspace, reduces the burnout, and time spent on setting it up.

Ara's Components

Client E: Maydan

Stakeholders: Project managers, Node operators, Developers

Technology: Based on Vulkan/Wayland protocols, with the built in Chromium browser and Godot for spatial data.

Compatibility requirements: Similar to standard desktop shells, all standalone applications open as normal.

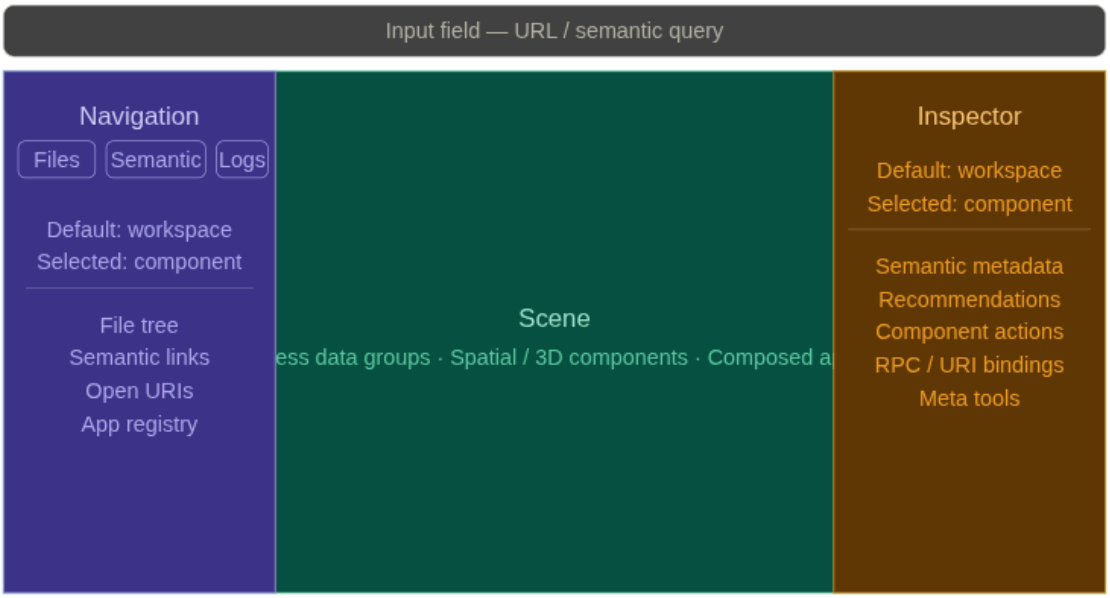
Additional requirements: Input Field. The Input Field is a search bar for entering web URLs for the apps registering their semantics on Arada.

Web apps registered on the Arada open within Maydan's built-in browser. DOM[10] is directly available for matching against the semantic layer.

[9] Close, minimize or fullscreen
[10] Document Object Model, see more on Wikipedia

Maydan ships with semantic data for project management, including a built-in list of apps stored in the file system or remote web applications.

Layout



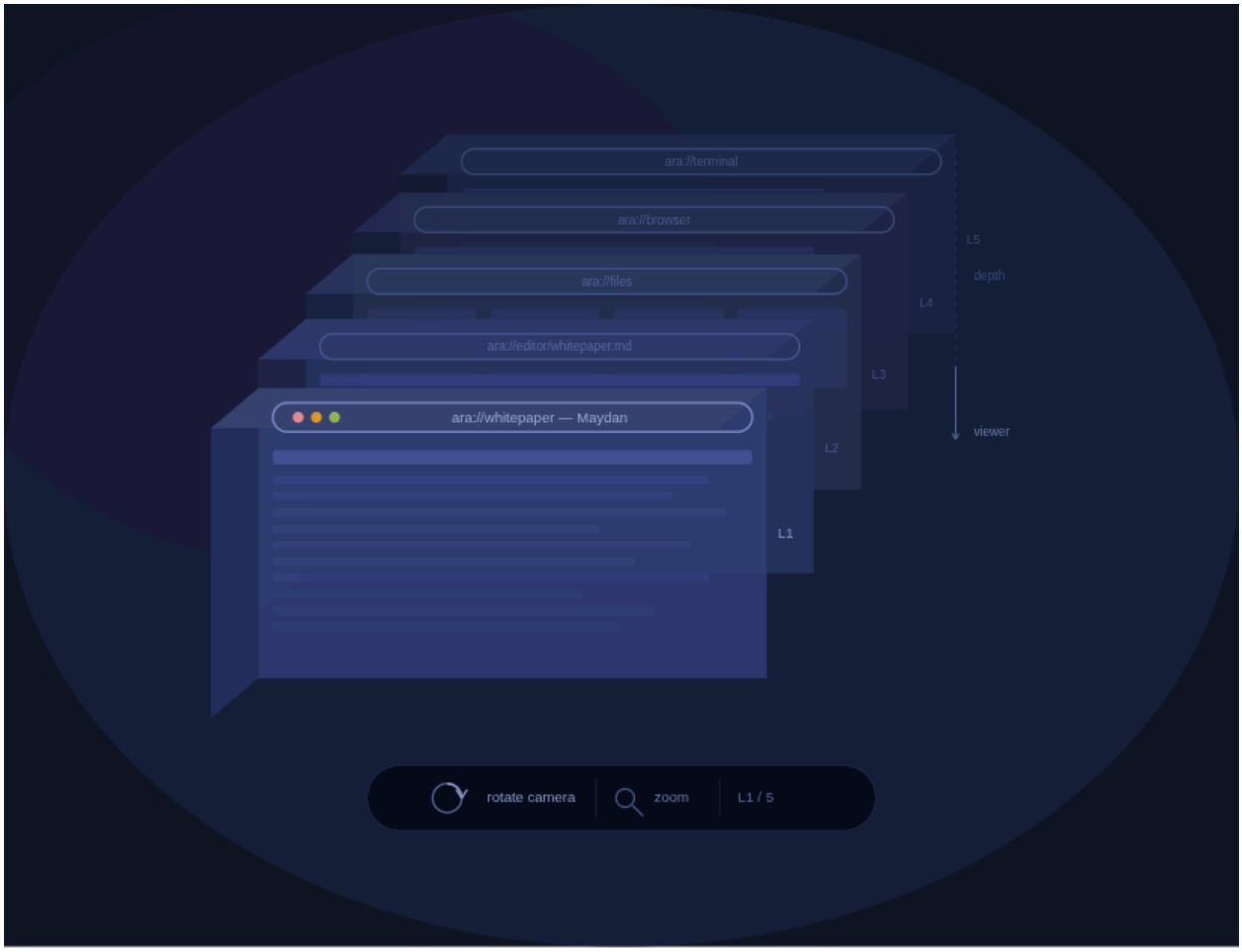
The screen shows the scene of an application, while navigation and inspectors side panels show application parameters.

The navigation has three tabs, to navigate by three ways: file structure, semantics, and logs.

If the user opened an app, or selected the group of data, the navigation and inspector addresses that. If the user did click nothing, then the navigation and inspector is addressing the workspace itself.

Scene

In Ara, the scene is an infinite virtual canvas with stack-based layers.



The scenes can be rotated and seen from a different camera angle. This allows users to select and reorganize data across layers.

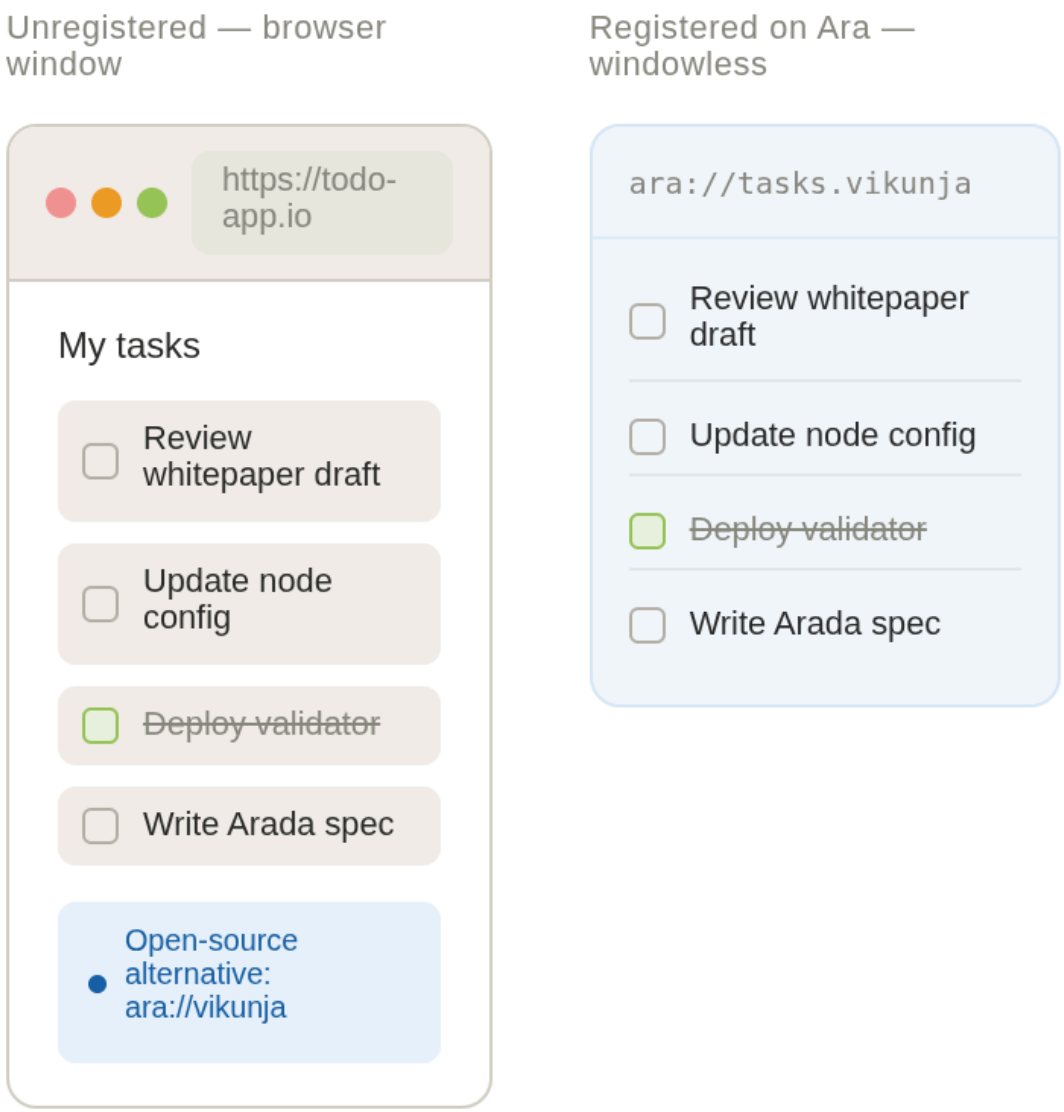
Users familiar with the scene and camera controls in Godot, Unity, or 3D modelling tools will find the logic familiar.

By default, the front layer is layer 1. Each popup, context menu, or overlaid app appears on layer 1+n, up to a maximum of 1024layers.

Windowless

The web apps typed on Maydan's InputField, are shown as a windowless application, if its semantics are registered on the Ara network. If not, then the web page is shown with the border,

however the recommendation engine shows open-source alternatives if it exists.



Users may group data from multiple websites, together into a custom website. This custom windowless data group is a local website with the file:// protocol URI.

Spatial 3D objects

Scene stacks are rendered by the Scene Compositor and run by the Spatial Compositor, which operates as a transparent Godot Scene overlaid on the desktop.

The Spatial Compositor allows any semantic component in a windowless data group to be replaced by a 3D asset. But in order to make them interactive and more based on semantics, it supports scripted 3D objects, allowing users to share spatial data as Godot's Pack file directory.

Holographic recommendations

Data retrieved from the recommendation engine is displayed holographically. This uses the holographic shader in Godot.

Holographic data displays a title, description and properties panel to its right.

file://work-inbox

ara://mail.proton

- Arada spec review 09:14
- Node validator update 08:52
- Grant application sent Yesterday

ara://cal.nextcloud

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

file://work-inbox — holographic

ara://mail.proton

- Arada spec review 09:14
- Node validator update 08:52
- Grant application sent Yesterday

ara://cal.nextcloud

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

ara://docs.cryptpad/regions.doc

Country	Users	Revenue
Turkmenistan	1,204	$4,810
Vietnam	3,870	$9,340
Germany	7,120	$22,600
Brazil	5,430	$11,200
Japan	6,890	$31,450
Nigeria	2,310	$3,920

holographic asset — globe.pack

The country data table can be replaced by a spatial 3D globe, mapping the same data geographically.

The 3d asset sharing follows a marketplace model for the game assets. The full scope of this sharing model remains an open question.

Meta Tools

Meta tools are: *highlight*, *group*, *reset*, *zoom* and *camera*.

Highlight is to select the group of data. Selection works either by holding CTRL and clicking individual components, or by dragging a mouse selection over a screen area.

Zoom in, *zoom out*, and the camera changes the view across all scene layers.

When a component is selected, reset clears the selection. When nothing is selected, it closes all open apps.

Once a component is highlighted, Maydan shows a contextual menu. These are: *close*, *minimize*, *resize*, *recommend*, *stack*, *move* and *search*.

The first three commands are equivalent to standard window operations. In Maydan they can be applied across layers or to specific components within a page.

The recommendation is showing the possible data or alternative for the selected group. It is shown as the holographic equivalent and with short descriptions.
Stack repositions a component between layers. Move repositions it within the virtual canvas

Search opens a menu to find specific data, additional tools or invoke AI.

Navigation

The navigation side panel allows the user to navigate the data or in general the whole computer in three ways.

First is the file representation, usually the local file data that the desktop shell maintains.

The second is the semantics of the app. The semantic schema is described in the upcoming Arada section. It maps the source code to what the user sees on screen, making it easier to store and retrieve components by meaning rather than file path. It is where users may manage, or create new semantic groupings.

The last tab is the logs. It tracks the user's activity within the app since the current session began.

Settings

The input field accepts a special keyword "ara" that opens the node page in the settings.

The desktop shell has its own domain, at the user level and managed through the blockchains. Validators are available using the .ara domain name.

Other domains can be added, for example a wallet address using .eth which opens up the corresponding app for that domain.

Maydan has four additional system settings of the desktop shell: account, gateway, settings and assistant.

The first is the account settings, managing the user's private key.

The second setting is to manage nodes. It's where users are picking the nodes of their choice. It contains four URL lists:

- The request to Arada
- The request to Aramak
- Subscribers for Arada
- Subscribers for Aramak.

Additionally, it has a side panel describing the payment.
Again, due to windowless nature, it could either show a payment as a single place, or one for each type, or individually. It doesn't lock users to a single configuration. If a user wants to use one single payment, then it will filter out validators that support the payment method of the first validator. If not, then for each validator user could customize their own validator.

The third is settings, which covers the desktop shell's options and customizations. It is where the user also sees the marketplace, and it can update.

The last is the assistant, which provides the API and configuration for AI access.

Relationship with Arada

The Maydan connects with Arada for readonly data only. Upon opening the web app, it gets the information from the network about its semantics. If a semantic entry exists, it loads the semantics in parallel in the navigation side panel. Register web apps in the recommendation engine tracking. The meta tools also become aware of the opened windowless apps.

Relationship with Aramak

Every 10 seconds, user interaction is submitted to Aramak using semantics and zero-knowledge proofs.

Each batch sent to Aramak contains the desktop shell state, list of active windowless groups, virtual screen coordinates, zero-knowledge information about data in use and pointer's coordinates on the virtual screen.

If the user clicks on a recommended meta tool, or pauses for five or more seconds, Maydan automatically checks holographic data from Aramak.

Aramak returns up to three possible actions, either as an Arada URL or a web URL.

Semantic tracking with ZK proofs

ZK proof is provided by the module that runs in the background. It can read the file system, and the desktop shell itself, but network connection goes within the network. It has the second verifier attached to the gateways to verify the data.

Security

A potential attack vector is rewriting the desktop shell's hash or injecting a false application hash to inflate app reputation or manipulate the recommendation engine. This is prevented by users downloading Maydan or other Web3 desktop shells only from trusted sources.

As long as Maydan itself is not compromised, it can not alter application hashes. Additionally, the zero-knowledge signature generator can verify Maydan's hash directly in memory.

Blockchains

Although each blockchain in the architecture is independent, we assume one node runs both. This way, it is possible to update them separately.

Why blockchain instead of federated data, or relays that Nostr uses? The purpose is to track collective wisdom. First, log tracking maps naturally to blockchain logic, making it the right structure for observing how collective data accumulates. Second is the validator's reputation as a cryptocurrency. This incentivizes nodes to maintain the highest quality data globally.

Consensus mechanism and Araladyk (ARAK) token

The Maydan work is logged into the zero-knowledge proofs, sent through validators to the blockchain. Since validators are node operators, they set their own fees, denominated in blockchain PM tokens.

Each blockchain epoch, the network randomly selects N nodes out of the S nodes pool. If a node coin is confirmed by all N nodes it turns into the ARAK coin at the end of epoch.

Node coin uses the UTXO model, making the alchemy process traceable. They are equivalent to USD in daily transactions, while the nodes as foreign currency holders. The ARAK acts as the reserve, the equivalent of gold.

Even if three to five nodes dominate the network, their coins would function as the dominant currency. This does not prevent smaller, niche nodes from participating, supporting only specific recommendations.

Several open questions remain on the game theory and security properties of this mechanism. These include Nash equilibrium conditions for validator behavior, the economic cost of Sybil attacks, collusion prevention, alchemy manipulation resistance, front-running prevention, and the ratio of attack to potential gain. These are known considerations that will be addressed in a dedicated security specifications after choosing the blockchain SDK.

Blockchain A: Arada

Arada is to hold the semantics of the applications. The application semantics map files to their view on screen: which part of the UI maps to the code in the file system. Semantics enable behavior tracking and feed the recommendation engine, helping identify more refined components with less redundant code. The semantics are added by the software developers.

The protocol consensus for Arada is: *“a software registered on Arada is user-centric, not business-centric nor hobby-centric”*. Any developer registering on Arada agrees with this.

Ara requires app code to be hosted on a public git server and git commits must be signed with the developer’s private key for identity verification. To generate semantics, Ara will come with the Nodejs module that supports major web frameworks such as React, Astro, Svelte, Vue and Angular.

A developer first generates the semantics and SBOM and along with the authors receiving wallet addresses put it in the `ara.json` file. Then, push it to the git. Then passes the git commit to the Arada as a new transaction which will register the application upon completion.

Semantics

The semantics are stored in JSON-AD[11] format. The format is designed for decentralized databases that link by reference rather than duplication, unlike JSON-LD and other semantic formats designed primarily for the machine readability. The structure of the semantics has two necessary and one optional part.

- First, components with example data and their links to source files.
- Second, third-party dependencies described using an SPDX-based SBOM[12].
- *Optionally,* a flow description, aimed at developers and AI tools. Typically expected to align with the test files. If it’s done, then it must include a reference to the first part.

One open question of this design is how to handle apps composed of multiple parts across multiple languages. This remains a known limitation of the current design.

Spatial 3D data sharing

The spatial data is hosted separately from application semantics. 3D designers register on Arada by linking their asset to the relevant component’s semantic entry and pushing it as the view. Reputation and security for spatial data registration remain open questions.

Payment channel

The app’s payment is also described within the semantics. Payment flow is described using the hyperpayment protocol[13]. Arada applies the open-source hyperpayment specification by default, available with the semantic generators. For example, if an app charges for cloud hosting or runs on a subscription basis, it initiates a payment channel through the node gateway, which

[11] JSON format for web ontology by Atomic Data. See official documentation on developer portal.
[12] A format describing the software artifacts. See more on the Wikipedia article “software supply chain”.
[13] Hyperpayment protocol is described on hyperpayment.org

distributes 19% to the open-source projects, 0.19% to Ara nodes as the desktop shell fee, and 0.01% as the node fee. The remainder goes to the app's maintainers.

Developers can adjust this for their own needs. The semantic structure maps directly to whichever hyperpayment method the user applies.

There is an additional reason for targeting open-source projects. There is no legal pressure for the hyperpayment split to be perfectly precise, since no formula can fully capture what constitutes a fair split.

Transactions

Transactions by the software owners

- **Register** registers a new application by its git URL.
- **Check** checks for a commit and from that the `ara.json` is refetched.
- **Withdraw** withdraws the tokens in the maintainer's balance on blockchain PM; node can charge a fee.

Transactions by the users

- **Read** returns the semantic data.
- **Hyperpay** initiates a payment on Arada's certain semantic part; node can charge a fee.
- **RegisterSemantic** registers new semantic schemas made by users.
- **DeleteSemantic** deletes the user-made semantic schema.
- **ToggleSemantic** adds another user's semantic schema made by other users.

Transactions by the artists

- **RegisterAsset** registers a new 3D package linking to the semantic data.
- **Withdraw** withdraws the tokens in the artist's balance on blockchain PM.

Transactions by the nodes

- **RegisterEnvironment** registers a new desktop shell.
- **ToggleEnvironment** toggles the support of the desktop shell by the node.
- **OptionalCollateral** changes the collaterals supported by the node.
- **WithdrawCollateral** withdraws collateral accumulated by the node prior to current epoch.
- **TransferToken** transfers token ownership to a new account with no existing mints.
- **RegisterToken** registers a new token issued by the node.
- **ToggleToken** changes the tokens supported by the node.
- **MintToken** mints a new token associated with the address.
- **Node** returns a list of supported collaterals, environments, and token information.

- **RegisterHyperpayment** registers a new hyperpayment protocol by the node.
- **ToggleHyperpayment** changes the hyperpayment specifications supported by the node.

Transactions for all accounts

- **Transfer** transfers the node tokens, or ARA between accounts.
- **Reserves** returns the list of all tokens associated with the account.

Blockchain B: Aramak

Aramak holds the user behavior related to the software semantics. The zero-knowledge proof is to show that a user's behavior occurs on real data within their computer, without revealing what is actually being processed. The only requirement is that the gateway (verifier) and the client (prover) share the same protocol. The suggested zero-knowledge protocol is the zkSNARKS[14] as it's the fastest proof generation.

Every 10 seconds, user interaction data is submitted to Aramak. Batches are not stored on the blockchain but on the client-side to represent the chronology of the app use. Instead, Aramak tracks only the submitter's last batch. And the file changes — modifications to the app's virtual file structure. This file structure changes, when the user creates custom semantics, when the app requests the network data or the user changes data in the view. These are submitted as ZK-proven events mapped to their semantic component identifiers and virtual screen coordinates, without sending the DOM snapshot or raw content.

From each processed batch, only the derived behavioral increments are written to the chain. Aramak persists interaction counters, and file changes related to the semantic link: how many times users interacted with a given component, in what context, alongside which other components. This recommendation engine operates from this behavioral graph directly. Data retention policy and storage costs remain open questions to be resolved.

Aramak uses collaborative filtering as its recommendation algorithm. Users with similar app combinations, the same workspace stack, arranged and customized in a similar way, surface relevant apps, spatial assets, and workspace layouts to each other. Over time, as the behavioral graph accumulates, Aramak compares virtual screen arrangements across users with matching stacks. A user whose layout and interaction pattern converges toward the high-performing configuration used by others with the same apps will receive that configuration as a recommendation holographically, as an overlay the user can adopt, customize or dismiss. New professions emerge from this: workspace curators, visualization specialists, and workflow

[14] zkSNARKS generates a proof in 10-100ms, and verifies in 5-10ms. zkStarks generates a proof in 50-500ms, and verifies in 20-50ms. The bulletproofs generate a proof in 100ms-1s, and verifies in 50-100ms.

automation authors who publish optimized configurations and receive hyperpayment attribution when others adopt them.

Recommendations are triggered in two cases: when the user is idle for five seconds, or when the user activity log contains a repetitive pattern. Aramak returns up to three possible actions as an Arada URL. The algorithm specification is open. Nodes may implement alternative models provided the input and output interfaces remain standardized.

Transactions

Transactions by the users

- **Track** submits a new batch of behavior records.
- **Recommend** returns a recommendation for a given semantic entry or behavior.
- **Info** returns user’s schemas and the last batch.

Transactions by the nodes

- **RegisterModel** registers a new recommendation model, associated with a semantic or behavior schema.
- **ToggleModel** supports the model registered by other nodes.
- **Node** returns the list of supported models and schemas.

Transactions for all accounts:

- **RegisterSchema** registers a new behavior schema.
- **ToggleSchema** adds another account’s schema to the caller's active list.
- **DeleteSchema** removes a schema. For nodes, removed schemas are excluded from all future registrations and responses.

Conclusion

Blockchain showed the potential of decentralization, but the promised Web3 never materialized.

There are five reasons. The first reason is generally applicable to the whole technology space: focus too much on data, rather than software. As a result, software development went its own way, and the users lost the software ownership.

The second and third reasons are the architecture of the blockchains. Blockchains miss the core component of the web, hyperlinks. Blockchains are designed around miner and native token holder incentives, treating self-sufficiency as the goal rather than serving builders.

The fourth and fifth reasons are the lessons from the rise of tech giants to apply for a global platform. First is the inclusion of UI/UX in the incentives. Secondly, to include the recommendation engine to discover apps.

We propose the *Ara framework* and its first implementation for project management and open-source software sustainability.

Ara framework proposes two new blockchains, blockchain A for a protocol agreement over a shared semantics. The blockchain B is the recommendation engine built on the first blockchain's data. Ethereum's Dapp browser, Mist, identified the right instinct. Ara extends it as a full desktop desktop shell where users run and compose software on their own terms.

The network is decoupled from the node incentives. The dual blockchain acts as the coordination layer while computation moves to the user's computers. This is **Web3: a creative workspace built on collective wisdom.**

This Web3 also prepares the stack for AI to further personalize user's creativity. Together they create a metaverse, not a VR social environment, but rather a modern version of the Lisp or Smalltalk environment returned to the user's computers.

—

For questions, discussion, and feedback, visit ara.foundation/aratalk.

Glossary

- Ara - a concept and protocol of the web 3, as a dual blockchain along with a software environment that brings software ownership to the users.
- ARAK - Token ticket of the alchemized token of Ara network. Short for Araladyk.
- Araladyk - A token name once completed in the alchemy process within Ara network.
- Maydan - a name of the perfect shell. In Turkmen language it means a public square or arena.
- Arada - a name of blockchain A. It's where hyperpayment protocol and software semantics are recorded. The word "Arada" means *something in between*, or *state in between* in Turkmen language.
- Aramak - a name of blockchain B. It means to seek, to look for something in Turkmen language.
- Aralyk - any semantic node written in the Arada. It means the quality of being in between or state of intermediacy.

Appendices

Future of computers

AI

The redefined Web3 opens up a potential for the new use case for AI.

The current hype began with OpenAI at the end of 2022. For a decade, corporations had deployed chatbots trained on their own databases. OpenAI kept the same interface but replaced the corporate database with the entire web. The result was ChatGPT.

Every major name followed. With FOMO as the driver, they promised to capture the same momentum but delivered variations of the same chatbot. They confused the success of the interface with the success of the idea, and currently AI integration means integrating the AI chatbot into the software without reimagining what it could do.

The AI interface has not changed. What is missing is a total reimagining of the UI/UX: a modular environment built for creativity.

While researching this whitepaper, we discovered that the authors of Siri were one of the pioneers of web ontology, which suggested the field was moving in the right direction, but the semantic layer wasn't there. Web3, by semantically aligning software rather than data, provides what Siri lacked.

With AI, the interface and composition opens up more personalized, future-rich applications not yet possible. Current desktop automation tools work by capturing screenshots or staying within API boundaries. In Web3, they operate through a semantic layer of the applications.

Social networks illustrate the AI's new possibilities within Web3 directly, particularly dating apps. Every major social network and dating app shares the same UI, differentiated only by content type. In this example, the blockchain holds a user profile semantically, and the recommendation engine drives the matchmaking.

AI, especially LLMs, showed that they can understand text messages properly. Social networking in real life operates by unwritten rules: different for romance, friendship, or professional hiring. Dating apps, particularly, could apply real-life social rules to both the UI and application logic, organizing interaction into levels of disclosure, going progressively deeper into the personal information. The software ownership means these tools are available to anyone building on semantic data or matchmaking logic.

AI is on the client side, along with the data, accessing the software through the meta UI. Web3 gives AI access to decentralized data as a form of collective wisdom, while client- side interaction personalizes the environment. These use cases, built by people and distributed through the recommendation engine demonstrate what AI is actually for. It resolves the fear around AI, as it becomes a creative tool that serves the user's intention, not as a replacement for the user.

Metaverse redefined

Web3 gives the creative environment, AI makes it personal and amplifies it. Together they produce the metaverse, the ultimate form of creativity.

This metaverse is not a game world for fun and social play, but the entire computer itself with spatial data, piping software together. It's like Lisp or SmallTalk where language, UI, and runtime are all the same thing, and let users modify and share it. The metaverse is a malleable computing environment with the 3D spatial data, something that simply didn't exist when Lisp and Smalltalk were designed in the 50s-60s. The data and personal UI, bound to the private key, live outside of any specific hardware.

It drives the further decoupling of the OS from the execution environment data environment. The OS becomes an API and the kernel provides the API. The Meta UI becomes the execution environment, and AI along with semantics becomes the user's interface to its metaverse.

When a user logs in another computer a user has two options: fully restore their own AI, data and active session, or simply let AI adjust the new hardware's UI to their preferences.

This metaverse will emerge naturally from Web3's success, as the next iteration of computing, not a designed destination. It's not Ara's goal, but an inevitable future. Spatial data, AI, semantics, and decentralized infrastructure — abstracted from hardware altogether — make it possible. Ara is the first step that focuses on project management and open source sustainability.

After Metaverse

The metaverse is not the final form. Beyond it lies a further shift, AI as the interface itself, mediating between the user and all connected devices. The question that follows is: what replaces the keyboard, the touchscreen, and the mouse?

Two trajectories exist. The first is direct neural input in a form of brain chip that reads intention before becoming motion. That path leads toward cyborg augmentation, controlled by the few corporations with the deep medical infrastructure to make it viable. If it appears, it will be rigid, proprietary, and available only to those who accept its terms.

The second is body tracking. Camera tracking of gestures and movement is the obvious candidate, but cameras are unreliable across angles, lighting conditions, and distances.

Jewelry is the resolution. People wear jewelry by choice. Each piece carries its own hardware and acts as the user's deliberate interface to the computing environment. Each piece holds one or a few roles: a battery, a private key, a wallet, a microphone. Together they compose a personal input system organized around the user's own body and movement.

Jewelry is craft. It is made by hand, customized, inherited, and traded. A craftsman who understands both electronics and form becomes a new kind of professional. The result is an input ecosystem that grows more diverse and more personal over time, not more uniform.

In the future jewelry will become something more precise: a marker of sovereignty over one's own computing environment. The person wearing a composed set of functions is not displaying wealth, rather they are displaying independence.

The most prominent single piece will be a ring on the pointing finger. A ring on that finger, aimed at a fridge, or a screen, connects the user to AI running in the metaverse, which reads the gesture through semantics, understands the intent, and projects the response holographically. The user does not touch the screen. This shall be called Solomon's Ring.

Pieces work together to compose entirely new input systems organized around the user's body and movement.

When the user removes their accessories at the end of the day, they step out of the computing environment entirely. The offline self is restored on the user's own terms.

Holographic output becomes the natural complement. As jewelry and AI combine through semantic understanding in the metaverse, the screen itself becomes optional. The display ceases to be a fixed object and becomes a responsive layer of the environment.

The era of the keyboard, the touchscreen, and the mouse era will close, not because they failed — but because something more personal, more composable and more sovereign replaced them.

Ara Framework for domains

Social Network on digital identity

The blockchain A holds semantic user profile, verified, extensible and optionally ZK-obscured. Blockchain B's recommendation engine drives matchmaking against that profile. The second blockchain allows the matchmaking algorithm for the users that relies on the semantic user data.

It allows not only to create various social networks such as friend based (which finds people who worked in the same group, but for its sake, only people who studied there, rather than public for anyone). It can allow users to extend the semantics by creating the groups. But since it's extendable, let people create dating apps that run by transparent matchmaking. People could run the matchmaking by various algorithms for example, perhaps based on astrology, enneagrams or DNA to find those who have similar roots. Since not only user data, and software itself is important but also view is important, others may find the way to make the level based chatting, by defined set of rules, then for example in the first 20 minutes talk about the general information, and if it filters out, the second level talks about personal job, hobbies. And the ultimate one is the confidential information as well.

Another way it can work is, rather than finding your own tribe by the common groups, which is partially correct, it can help people find those who are in the similar three or more groups.

People, who use the dating app, will find their own endless ways on how to connect to others. Seeing how people connect, how its view is represented, lets them create their own variations that are shared across the world.

Its client side could be acting as the holder of the user's personal data, that is connected to the government websites or other tools such as work places, DNA, or psychological clinics, where people could put the necessary data and keep them in the ZK proofs. It can also work as portraits, and various photos and information to permit or not permit.

This for the data analytics could allow verifiable studies, across various groups, for the sociologists and psychologists.

It can also be the best advertisement targeting. But here, since blockchain is the intermediary for protocol, while money goes between users and those who create the software, the money for ads could go directly to the users. Perhaps it will find a radically new mechanism, when a person who receives the job offer, will redirect in his networking circles, by getting the fees.

Games, NFTs and metaverses

Ara's framework applies directly to games.

The blockchain A will allow game developers to register their games, and what can be changed, and what can be added there: skin, character, addon or modification. The changes are linked to NFTs. The blockchain B's recommendation system will allow users to match new NFTs, and integrate them into the games and integrate them across games.

The client side will allow users to list their NFTs, find similar ones. It also acts as the Steam, or Xbox marketplace alternative as well. Because here NFTs are not bound to the game, but independent and bound to the players. Rather games are the environment where they can find new contexts.

Users may add addons, and instead of trying to make them part of the game's world, it is part of the users authorship, while semantic blockchain allows them to split the revenue with the original authors.

Client side is not sufficient alone, game engines also need to support semantic and mod-making capabilities natively.

Within the game itself, and choosing their NFTs, or wishes could add new characters. For example, a fighting game, built in with the Ara network for games could have an additional button that allows users to add their NFTs, and request the freelance platforms to create the game characters. While thanks to the game semantics, freelancers get the examples (from the game itself), the technical requirements for the 3D model, animations. And once it's released, it will be bound to the NFT. NFT is not just a card, but also binds multiple variations across the games on how this NFT could be.

The game authors hold back their modifications out of fear of losing clients to new games. The global blockchain that treats their authorship will allow them safely to release their games. The NFTs not only establish authorship but enable derivative works, with royalties automatically redirected to the original authors.

With the rise of large language models, NFTs gain a new dimension of value. Beyond skins and game models, a player's mechanics can be encoded and released as an NFT. A pro player who trains a model on their own replays, tweaks and tests it, then releases it through the Ara's network creates something genuinely scarce playstyle. Blockchain B surfaces these to players with compatible preferences, while blockchain A tracks authorship and ensures royalties flow back to the original players.

Metaverse

The same framework extends to metaverse-scale world building. Arada registers not just games but entire functional universes: their lore, rules, characters and semantic relationships. A world like Murim, rooted in the ancient Chinese martial arts world, or the Witcher's universe spanning Western fairy tales lore, becomes a shared semantic foundation that any developer or creator can build on. New games, characters, and scenarios created within that universe are registered on blockchain A as derivative works, with authorship tracked and royalties flowing automatically to the original world builders. Even large IP holders such as Disney or Microsoft could open their

universes of Marvel, Star Wars, Halo allowing anyone to create variations while retaining royalty control through the protocol. .