Nimble - The Ecosystem First Web3 Intent Infrastructure

Build Web3 Intent Economy for Satisfaction

Preprint

September 26, 2023

Abstract

Today's blockchain systems are obscure. For users, DApps with an unfamiliar user flow makes it prohibitively difficult to onboard the next billion Web3 users. For developers, launching a DApp is not easy for a lot of Web3 domain knowledge and system complexities. For blockchains, developer communities are slow to grow. For instance, DApp operations like bridging are time consuming, annoying and insecure. To launch a DApp, developers need to understand different ecosystem particulars in details. Blockchains are competing for the same set of Web3 developers inherently since the Web3 development is so different from Web3 development.

Nimble is building an intent layer to solve this notorious pain-point and further boost Web3 composability. User interactions with DApps are modeled as queries. The protocol understands queries with intent modeling, route the queries to specialized solvers and then execute the queries on-chain. Queries can be GUI inputs and natural language queries. Example queries can be as simple as *sell ETH with minimum price of 1000 USDC*, or as complicated as *how to build a lending protocol from scratch*. Meanwhile, critical Web3 components like price searching, crypto market making, wallet contracts and so on become solvers in Nimble protocol. DApp builders can thus get focused on the user experience and product building. Solvers compete for intents, minimizing harmful MEVs as well. A decentralized network is formed for Byzantine fault-tolerance.

The protocol development adopts a community first and ecosystem first driven approach. The protocol is designed to be open, decentralized, and easily extensible.

1 Introduction

Figure 1: Nimble intent protocol hides the blockchain low level details like operating system for computers.

Today's blockchain user experience is like using computers in the early days. To use a computer at that time, users had to understand hardware configurations, install drivers and interact with applications through ugly user interfaces or even command lines. It was difficult to both users and developers. That is similarly true in today's blockchain world. Users have to understand the differences among blockchains, wallets, and Dapps. Developers cannot build platform agnostic applications like Web2.

The Nimble vision is to deliver a Web3 intent infrastructure for Web3 user satisfaction. As a Web3 portal protocol, Nimble sits on top of blockchains, L2s, bridges and protocols. It hides all the underlying complexities like computer operating systems. Users and developers interact with Nimble directly and do not need to worry about the Web3 system complications. DApps can be built much more easily, and users interact with DApps naturally. The development philosophies are *decentralization*, *open* and *community first*. Decentralization is achieved with an overlay network governing the messaging, payload parsing and then intent processing. The protocol is open to use with publicly published SDKs and APIs. Community members can contribute to the code development and product direction discussions.

The Nimble network is a set of operators forming an overlay network and running intent processing and reaching consensus for intent operations. There are three protocol modules: intent recognizer, intent dispatcher and intent validator. Each operator is a physical machine on top of which multiple conceptual modules can be run. The overlay network adopts gossip protocols and distributed hash tables. Due to the numerous literature on that, in this paper, the discussions are focused on the conceptual modules instead, which are the key building blocks and innovations of this work:

- **Intent Recognizer.** Intent recognizer accepts intent inputs and translates them into concrete intent operations. Intents are mapped into tree based taxonomy from ontology. Intent operations are leaves on the trees. LLMs are adopted for natural language intent understanding and rule engines for structural intent processing (i.e., map user intents into tree leaves).

- **Intent Dispatcher.** Intent dispatcher hosts an intent pool of intent operations and runs intent auctions. Intent pools are represented as buffer maps. Each row in the buffer map is an intent

operation from intent recognizer. Domain specific languages (DSLs) detail intent operation specifics. Dispatchers exchange buffer maps to expand intent pool visibility of user queries.

- **Intent Validator.** Proof of Stake consensus is the key logic implemented by the intent validators. Validators reach consensus about intent operations instead of blockchain transaction themselves.

Section 2 details the three key intent protocol modules: recognizer, dispatcher and validator. Section discusses other critical system components such as intent contracts and SDKs. This is followed by related work in Section 3 and conclusions (i.e., Section 4.

2 Intent Architecture

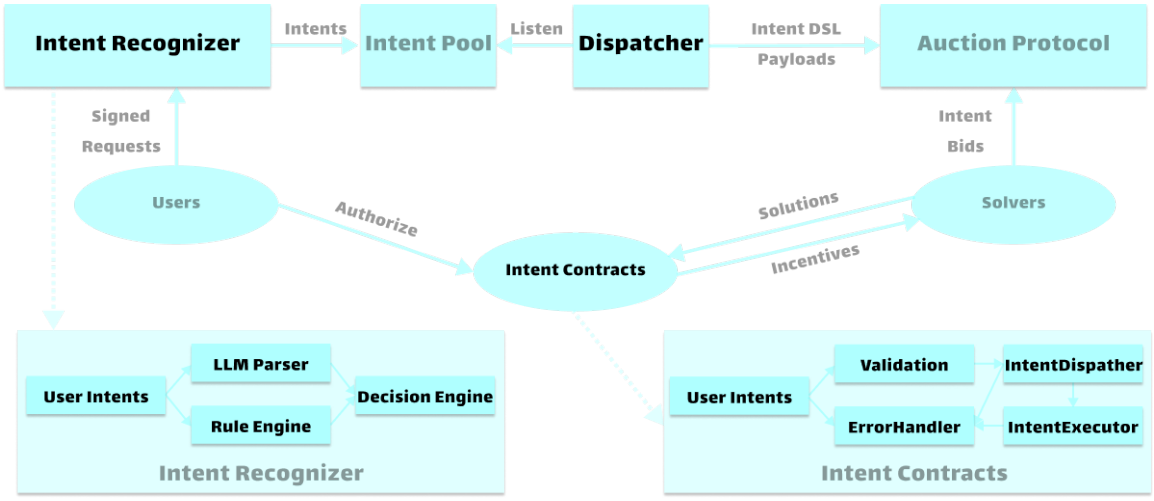

Figure 2: Nimble intent protocol consists of three essential modules that work together seamlessly to facilitate decentralized user interactions with blockchain technologies. Intent recognizer understands user intents as machine-readable intent operations. Intent operations are published to intent pools a critical piece of intent dispatcher. Intent dispatcher runs second price auction for truthful solver bids. Solvers interact with auction protocol within the dispatcher for intent competitions. Intent contracts provide permissionless onchain executions.

The intent user flow is depicted in Fig. 2. The key design components are discussed below.

2.1 Intent Recognizer

The Intent recognizer serves as the gateway to Nimble protocol, acting as the interpreter of user intentions. Its primary function is to bridge the gap between the user's natural language expressions and the protocol's machine-readable understanding. Leveraging Natural Language Processing (NLP) techniques for entity recognition, this layer deciphers user input, extracting the essence of their blockchain-related requests together with rule based approaches. It translates these intentions into a structured format, such as a list of user operations, often using an intuitive Intent Query-Based DSL. This layer is essential in ensuring that user interactions are comprehensible to the subsequent layers, creating a seamless and user-friendly entry point to the world of Web3.

- This foundational layer serves as the entry point for user interactions with the Web3 protocol.

- It utilizes Natural Language Processing (NLP) algorithms and smart contract interfaces to decipher and extract user intent from textual or voice-based inputs.
- Within the Intent Recognizer, a key component is the utilization of an Intent Query-Based DSL, which serves as the bridge between users' natural language intents and machine-readable operations.
- This DSL allows users to express their blockchain-related intentions in a structured and declarative manner, resembling SQL or YAML for ease of use.
- Users can formulate complex queries or declarative statements that specify the desired blockchain actions, parameters, and conditions.
- It abstracts the complexities of blockchain interactions, including smart contract interactions, token transfers, staking, and more, into intuitive and human-readable syntax.
- It employs advanced parsing and pattern recognition techniques to translate DSL statements into a list of user operations comprehensible to the protocol.

2.1.1 Intent Representation

Intents are outcome driven messages signed by users. Intents are desirable user specified state rather than instructions like transactions. Intents focus on user preferences, the product journey and the desired protocol outcomes. For extensibility, intents are standardized in both the definitions and understanding. Example intents are:

- Stake ETH for best yields,
- Trade USDC for ETH at the optimal price, and
- Swap USDC for ETH with minimum slippage of 0.1%.

To have unambiguous definitions, tree based ontology is adopted. Ontology, as a branch of philosophy, is the science of what is, of the kinds and structures of objects. In simple terms, ontology seeks the classification and explanation of entities.

Nimble taxonomy is a forest formed by a set of trees. In computer science, a tree is a widely used abstract data type that represents a hierarchical tree structure with a set of connected nodes. Each node in the tree can be connected to many children (depending on the type of tree), but must be connected to exactly one parent, except for the root node, which has no parent (i.e., the root node as the top-most node in the tree hierarchy).

Each tree defines an independent intent category. The root is the largest concept for that category. The leaf is a particular operation defined by DSL. The forest is extensible by adding new trees for new operation categories with the evolution of the Web3 concepts. Trees can be extended by adding new operations. In the example above, new derivative types can be added constantly as nodes and leaves.

The forest and leaf operations can be defined as a set of configuration files or specific code languages. The specific implementation does not matter too much only if it is accurate. For example, operations can be defined as EVM, Move or Rust smart contract functions, while the forest is defined as configurations. An alternative can define both as contract programs. As a result, our definition of intent is extensible, accurate and flexible.

Figure 3: Nimble intent protocol consists of three essential modules that work together seamlessly to facilitate decentralized user interactions with blockchain technologies. Intent recognizer understands user intents as machine-readable intent operations. Intent operations are published to intent pools a critical piece of intent dispatcher. Intent dispatcher runs second price auction for truthful solver bids. Solvers interact with auction protocol within the dispatcher for intent competitions. Intent contracts provide permissionless onchain executions.

2.2 Intent Understanding

The outcome of intent recognizers are intent operations. Each intent operation is a leaf on a taxonomy tree. Before deep diving intent understanding, the specification of particular intent operations is discussed. A Domain Specific Language is a programming language with a higher level of abstraction optimized for a specific class of problems. A DSL uses the concepts and rules from the field or domain. In Web3, different intents are defined in a structured manner as a configuration or a piece of code. Below is a simple example intent *swap 10 BTC for USDT at maximum* 0.5% *slippage instantly*.

```
intentType: swap
  from: BTC
  to: USDT
  slippageThreshold: 0.5
  amount: 10
  delay: 0
```

Despite the extensible, accurate and flexible DSL definitions, it is impossible for users to learn it before using the products. Thus, the user intents can be in arbitrary format. It can be natural languages or GUI inputs from DApps. Due to the structured nature of intent definitions, tgoal of intent understanding is to parse the arbitrary user inputs into specific leaf nodes in our intent taxonomy.

The intent recognizer module is a combination of LLMs and rule engine as illustrated in Fig. 2. There is a wide literature on explaining them. Basically, rule engines define specific patterns to parse user input queries while LLMs are a large model of neural network used for pattern identification.

- **Rule Engine.** It defines a set of patterns to process the intents and map intents to leaf nodes in the taxonomy forest (i.e., specific intent operations).
- **LLM Parser.** LLMs are used for intent natural language understanding with named entity tagging by identifying salient entities, followed by specific operation classifications.

- **Decision Engine.** Decision engine is to combine the operation classification outputs with another AI model for intent operation prediction precision boosting.

2.3 Intent Dispatcher

The Intent Dispatching Layer serves as a crucial intermediary between the intent recognizers and solvers. It plays a multifaceted role in optimizing and safeguarding user operations as they move through the protocol.

Its primary responsibility is to efficiently manage and organize these operations as they flow in from the Intent Recognition Layer. Using event-driven architecture and real-time data batching, this layer ensures that user operations are processed in an orderly fashion. It employs a bundling mechanism to group related operations together, reducing blockchain network congestion and optimizing gas costs. By sequencing and prioritizing intent operations, it maintains the consistency and integrity of the blockchain network. In essence, this layer acts as the traffic controller, directing intent operations toward solvers for validation and execution, ensuring a streamlined and efficient process.

Additionally, this layer verifies the authenticity and integrity of each operation by checking the associated digital signatures, which must be validated to ensure that the operations indeed originate from authorized users and have not been tampered with during transit. Furthermore, this layer is responsible for implementing business logic rules and access control policies. It enforces constraints and conditions on user operations to prevent unauthorized or malicious actions, ensuring that only valid and secure operations proceed to the solvers.

- Positioned as the middleware layer between intent recognition and execution, the intent dispatching layer optimizes and organizes intent operations.
- Consists of bundling operations with specialized optimization algorithms.
- Optimizes and bundles related intent operations together for more efficient execution to reducing blockchain network congestion and optimizing gas costs.
- Ensures proper sequencing and prioritization of operations to maintain consistency and integrity in the blockchain network.

2.4 Intent Executions

The Intent Execution Layer serves as the engine that transforms user intentions into tangible blockchain actions, and it introduces an innovative element known as the Solver Network to enhance its capabilities further. Within this layer, the primary function is to leverage the Solver Network-a decentralized group of nodes specializing in different dApps and blockchain functionalities-to simulate and compete in solving user intents.

These specialized nodes form a decentralized marketplace of computational resources, offering their expertise and computational power to compete and find the most optimal solutions for user intents. For instance, some nodes might specialize in decentralized finance (DeFi) operations, while others focus on non-fungible tokens (NFTs), smart contract interactions, or decentralized storage solutions.

The Solver Network ensures that users receive efficient, cost-effective, and timely results for their intents, harnessing the collective intelligence of a decentralized community of experts. This layer's role is to coordinate and manage the interactions with the Solver Network, ensuring fair competition, transparent decision-making, and efficient resource allocation.

- Cooperates a solver network that leverages optimization algorithms, smart contract scripting, and decentralized finance (DeFi) protocols to competitively find the best solution to execute user operations.
- Anyone can join the solver network, deploy their own algorithms, and compete for the best solution to satisfy user intents.

2.5 Intent Contracts

User intent classes, execution, dispatching, validation and error handling are key components of the intent contract. We focus our discussions on the intent class, execution and dispatching, since validation and error handling come naturally. The UserIntents class defines intents through the taxonomy. It also consists of important intent utility functions. The IntentDispatcher is the unified interface for solvers to interact with and it routes intents to specialized IntentExecutor. IntentExecutor class handles particular intent operation executions. Intents are validated before being sent to the IntentDispatcher. ErrorHandler is called by different components whenever errors are detected. The flow is illustrated in Fig. 2.

2.6 Intent Validator

To participate in the network consensus validation, validators must have a minimum required amount of staked Nimble tokens. The staked amounts proportionately affect the $2f + 1$ stake weighted *PoAv* during intent dissemination as well as vote weights and leader selection during intent recognition and ordering. Validators decide on the split of rewards between themselves and their respective stakers. Stakers can select any number of validators in which to stake their tokens for a pre-agreed reward split. At the end of every epoch, validators and their respective stakers will receive their rewards via the relevant on-chain modules. Any validator operator with sufficient stake can freely join the Nimble consensus. All parameters, including the minimum stake required, can be set by the network enablement processes.

3 Related Work

This section discusses the evolution of decentralization technologies and Web2 data intelligence.

3.1 Bitcoin

In the evolution of blockchain, Bitcoin was the first viable solution for internet money and decentralized network. After that, numerous coins and blockchains were built. Among them, Ethereum improved decentralization and pioneered smart contract platform to build the Web3 infrastructure. In recent few years, combined with Proof of Stake, initially proposed by PeerCoin, building scalable Web3 smart contract platform has been the focus.

Bitcoin is a decentralized digital currency that can be transferred on the decentralized bitcoin network. The peer-to-peer network of nodes running bitcoin software maintain the bitcoin blockchain which is a public records of bitcoin transactions. The network nodes independently store a copy of the blockchain and validate transactions and append them to their own copies of the blockchain and transmit the additions to other nodes. In every epoch, a new block that contains a new batch of valid transaction records is added to the block chain and issued to all nodes.

The record keeping of the blockchain is done by the miner nodes by using their computer processing power. The easy to verify but hard to solve proof-of-work (PoW) is required in each new block to be accepted by the rest of the network. More specifically, Merkel Trees are adopted so that the non-miners can use the block header to verify a new blockchain without checking all past transactions. Once a new block is accepted by the rest of the network, the miner who successfully solved the problem is rewarded with newly created bitcoins and transaction fees.

3.2 Ethereum

Ethereum inherits Bitcoin ecosystem's consensus, decentralization, and cryptography. While Bitcoin is limited as a simple ledger, Ethereum blockchain has a built-in Turing-complete programming language that allows anyone to write smart contracts and develop decentralized applications (dApps) on top of it. Ether (ETH) is the native cryptocurrency of Ethereum blockchain that is paid to the miners as transaction fees. Due to their autonomous nature under their own logic, Ethereum contracts are often called smart contracts. The codes in the contracts are written in Ethereum virtual machine (EVM) code, a low-level, stack-based bytecode language. The EVM is the runtime environment of Ethereum that includes a stack, memory, gas balance, program counter, and the persistent storage for all accounts. When a transaction calls a contract's function, the EVM translates and executes the contract's code into stack operations. A transaction sender must pay the miner, who is running the EVM and adding the transaction to the blockchain, a certain amount of gas fee in ETH. The gas fee system can reduce the amount spam transactions.

3.3 Proof-of-Stake

During "the Merge", Ethereum switched its consensus mechanism from Proof-of-Work to Proof-of-Stake. PoW has a scalablitiy issue because it requires more computer processing power as networks grow and wastes a lot of energy. PoW's security in trustless networks comes from the miners who are competing to be the first one to solve the puzzle; therefore as more miners join the network, its security increases. On the other hand, PoS is more scalable and suitable for blockchains like Ethereum that not only serves as a ledger but also as an infrastructure for dApps, and need to not only process incoming and outgoing ETH transaction but also has to process transactions from decentralized finance (DeFi), non-fungible token (NFT) minting and sales, and more. In PoS, a blockchain employs a network of validators who stake their own tokens into a pool to get a chance to validate new transactions, update the blockchain, and get rewarded. A majority consensus has to be reached in order for a block to be accepted and added to the blockchain. Weighted voting can be adopted to improve the robustness of the consensus, where the weight is determined by the node's profile such as the amount of stake and reputation.

3.4 Intent Understanding

User intent understanding is well studied in Web2 for personalized search, recommendation and user growth. The mainstream approach is to adopt large scale AI models and knowledge graph for named entity tagging and similarity search. For example, search queries are tagged as entities like companies, human names, and so on. This understands the query structures and user purposes by mapping query keywords to concepts. Such concepts (i.e., entities) are mapped into a graph to represent their relationships. This lays the foundation to have a structural understanding of human intents. Another popular approach is to represent user queries as embeddings (i.e., a vector of bits). The intents of the queries and items (i.e., products in in e-commerce, articles in

search and posts in recommendations) are measured by cosine similarity among them. That is, queries with similar intents and items on similar topics are closer in the mapped space.

4 Conclusion

In conclusion, the Intent Protocol we are diligently crafting represents a pivotal leap forward in the realm of Web3 and blockchain technology. This innovative framework bridges the communication chasm between users and decentralized networks, offering a seamless and intuitive means to articulate complex intentions within this intricate digital landscape. By encapsulating the essence of human intent within a structured query language, we empower individuals to navigate the Web3 ecosystem with unparalleled ease and precision.

Our primary objective is to significantly lower the bar for users to enter the Web3 world. Through the judicious fusion of Natural Language Processing, machine learning, and blockchain expertise, our intent protocol paves the way for a harmonious collaboration between human cognition and the decentralized world. It promises a future where blockchain interactions are as effortless as conversing in plain language, all while upholding the security and integrity of blockchain transactions.

With the Intent Protocol, we venture into uncharted territory, shaping the future of Web3 by democratizing access to the blockchain. This groundbreaking initiative stands as a testament to our commitment to enhancing user experiences, catalyzing innovation, and ultimately transforming the way we interact with the digital frontier. As we continue to refine and expand upon this vision, we invite you to join us on this remarkable journey into the future of Web3. Together, we shall redefine the possibilities of human-machine collaboration in the blockchain era, making the Web3 world more accessible and inclusive than ever before.

