Software Architecture for OpenWorm Simulation and Blockchain Integration

Gianni Pojani

17/11/2023

1 Overview

The system is designed to execute the OpenWorm simulations, manage the simulation data, upload it to the InterPlanetary File System (IPFS), and record the data's location on the Ethereum blockchain.

2 Components

2.1 Simulation Execution

The OpenWorm simulation is run, generating data to be stored and shared.

2.2 Data Output Handling

The output from the simulation is collected, organized, and prepared for IPFS upload.

2.3 IPFS Integration

The simulation data is uploaded to IPFS, returning a content identifier (CID) for accessing the data.

2.4 Blockchain Interaction

The CID is stored in an Ethereum smart contract using ethers.js, allowing for decentralized access and verification of the data.

3 Workflow

The workflow initiates with simulation execution, followed by data output handling. The IPFS integration uploads the data and obtains a CID. Finally, the blockchain interaction records the CID in the smart contract.

4 Comparison with Traditional Cloud Storage

The proposed system departs from traditional cloud storage architecture in several key aspects:

- **Decentralization:** Utilizes the InterPlanetary File System (IPFS) for distributed file storage, unlike cloud storage which is centralized.
- **Data Integrity:** Employs Ethereum blockchain to ensure immutable and traceable data history, a feature not inherent in standard cloud services.
- **Gamification:** Incorporates a smart contract with gamification to motivate user engagement, absent in typical cloud storage solutions, haha.
- **Accessibility:** Demands more technical know-how for interaction compared to user-friendly interfaces offered by cloud services.
- **Performance:** Offers variable retrieval times that can be influenced by the distribution of nodes, unlike the optimized delivery of cloud services.
- **Costs:** Potentially lowers costs due to the peer-to-peer nature of IPFS, though blockchain interactions incur gas fees.
- **Control:** Provides users with greater data control, avoiding the restrictive terms of service of centralized providers.

These distinctions highlight the trade-offs between the two systems, with an inclination towards increased decentralization and user control at the potential cost of performance and ease of use.

Still, the distance between cloud storage and computing is not far away, if there will be necessity

5 Bridging the Gap Between Decentralization and Cloud Computing

While the system at hand is inherently decentralized, leveraging both IPFS and Ethereum blockchain, the conceptual bridge to cloud computing is not distant. Smart contracts can potentially trigger computational tasks, akin to cloud functions, providing a decentralized alternative to cloud-based triggers.

5.1 Decentralized Triggers in a Cloud Paradigm

Smart contracts can serve as event-driven triggers, similar to cloud functions in a serverless architecture, initiating script execution upon fulfilling certain conditions on the blockchain. This creates a decentralized event-driven architecture, offering a novel approach to automated task execution without centralized control.

6 Smart Contract Functionality

This section delves into the functions of the `DocumentVerification` smart contract, which provides a robust framework for managing document integrity and access within the OpenWorm simulation ecosystem.

6.1 Document Management

The contract allows for adding, updating, and securely deleting documents. Each document is associated with a unique identifier, and actions are immutably recorded on the Ethereum blockchain.

6.2 Version Control

To facilitate document traceability and revisions, the contract employs a versioning system that records every change made to a document, enabling users to access historical versions.

6.3 User Reputation System

The contract introduces a reputation system to incentivize quality contributions by rewarding users with reputation points for their interactions with the system, enhancing communal trust.

6.4 Document Rating

Within the ecosystem, documents can be rated. These ratings contribute to the overall reputation of the document's contributor, fostering a quality-first approach.

7 Enhancing Performance with Layer 2 Solutions

Considering the high demand for Ethereum network resources, Layer 2 scaling solutions offer a promising avenue to enhance transaction throughput and efficiency.

7.1 State Channels

State channels enable off-chain transaction processing, significantly reducing the load on the main Ethereum chain and allowing for rapid interactions within the OpenWorm ecosystem.

7.2 Plasma Chains

Plasma chains are autonomous blockchains that run in parallel to the Ethereum mainnet. They report back to the main chain, ensuring security while providing a high-throughput environment for complex simulations.

7.3 Rollups

Rollups execute transactions outside the main Ethereum chain but post transaction data on-chain. They come in two varieties: Optimistic Rollups and ZK-Rollups, each with distinct mechanisms for handling transaction verification.

7.4 Sharding

Sharding is a multi-phase upgrade to the Ethereum blockchain that will spread the network's load across multiple chains (shards), dramatically increasing the network's capacity to process transactions and store data.

8 Integration with Existing Systems

This architecture is designed not in isolation but with considerations for integration into existing infrastructures, bridging the gap between decentralized and traditional systems.

8.1 Interoperability with Cloud Services

The OpenWorm simulation architecture can interoperate with cloud services, allowing for the utilization of cloud computing power and storage when necessary, and providing a hybrid model that leverages both centralized and decentralized advantages.

8.2 APIs and Interfaces

The system is equipped with API endpoints to facilitate interaction with the smart contract, making it accessible to external services and creating a seamless bridge between the users and the underlying blockchain technology.

9 User Experience and Interface

A user-centric approach is vital for the adoption of any new technology. The OpenWorm simulation platform is designed with a focus on user experience.

9.1 Front-end Integration

A user-friendly front-end integrates with the blockchain backend, providing an intuitive interface for document management and interaction with the simulation data.

9.2 User Onboarding

The platform includes a comprehensive onboarding process to guide new users through the nuances of blockchain interactions, ensuring a smooth transition to this novel technology.

9.3 Feasibility and Efficiency Considerations

Despite the innovative integration of blockchain to replicate cloud-like functionality, efficiency concerns remain. The latency in transaction confirmation on blockchain and the dependence on network node distribution in IPFS can introduce variability in performance. However, the trade-off comes with enhanced user sovereignty over data and the inherent security and transparency of blockchain operations.

9.4 Cost Implications

While decentralized systems may reduce costs associated with data storage via IPFS, blockchain interactions, particularly on Ethereum, can be costly due to gas fees. This factor is crucial when considering large-scale adoption and continuous operation of decentralized applications.

9.5 Future Prospects

As the technology matures, optimizations and advancements in blockchain and decentralized storage solutions are likely to bridge the current performance gap. The convergence of these technologies with cloud principles could lead to a new hybrid model, offering the best of both worlds: decentralization with cloud computing efficiency.

10 Conclusion

This architecture ensures a seamless pipeline from simulation to blockchain storage, facilitating data integrity and accessibility.