Design Strategies, Method, dApp Development Process, Reference Architecture for Blockchain Applications (REF-ArcBC)

It is important to understand dApps for Blockchain Application Development. Decentralized applications, commonly known as dApps, have revolutionized the world of blockchain technology and application development. In this brief introduction, we will define what a dApp is, clarify what it is not, and outline a systematic process for dApp development. A dApp, short for decentralized application, is a software application that operates on a blockchain network. Unlike traditional centralized applications that rely on a single central server, dApps leverage the principles of blockchain technology to run on a decentralized network of computers, known as nodes. Key characteristics of dApps include:

* Decentralization: dApps operate on a peer-to-peer network of nodes, eliminating the need for a central authority or intermediary.
* Transparency: Transactions and data within dApps are recorded on a public ledger (blockchain), making them transparent and immutable.
* Security: The cryptographic nature of blockchain technology ensures the security and integrity of dApps, reducing the risk of fraud or data tampering.
* Open Source: Many dApps are open-source projects, encouraging community involvement and contributions.

Developing a dApp involves several key steps and Ethereum development resource provides more detailed platform and learning resources (https://ethereum.org/en/developers/):

* Idea and Conceptualization: Start by defining the problem your dApp will solve or the unique value it will provide to users. Consider the blockchain platform (e.g., Ethereum, Binance Smart Chain) that aligns with your goals.
* Design and Architecture: Plan the user interface (UI), user experience (UX), and overall architecture of your dApp. Choose the appropriate blockchain and smart contract platform.
* Smart Contract Development: Write and test the smart contracts that will power your dApp's functionality. Ensure security and efficiency in your code.
* Front-End Development: Develop the front-end interface of your dApp, which interacts with the blockchain through web3 libraries or APIs.
* Testing: Thoroughly test your dApp for functionality, security, and performance. Use testnets to simulate blockchain environments without real assets.
* Deployment: Deploy your smart contracts to the chosen blockchain network and make your dApp accessible to users.
* User Onboarding: Provide clear instructions for users to interact with your dApp, including wallet setup and transaction processes.
* Community Engagement: Foster a community around your dApp to gather feedback, address issues, and encourage adoption.
* Maintenance and Updates: Continuously monitor and maintain your dApp, addressing bugs, optimizing performance, and implementing updates.
* Scaling: Explore options for scaling your dApp as user demand grows, considering solutions like layer 2 scaling or sidechains.

It is important to choose a blockchain dApp development process, design method, and data structure based on strategic business applications. Blockchain (dApps) design consists of several steps such as elaborating and mapping requirements, design rationale, and carefully choosing blockchain algorithms for more energy efficiency, computational efficiency, scalability, etc. as shown in Figure 9 provides a development process for dApps which consists of several stages:

* Elaborating on dApp Requirements with BC-SQUARE involves gathering blockchain application requirements using the domain analysis method discussed in 5.2 and understanding stakeholder needs and business viability analysis.
* Creating Proof of Concept (POC) involves creating proof of concept by modeling and simulating business requirements using Business Process Modelling and simulation tools such as BonitaSoft, Bizaghi, etc.
* Selecting your dApp platform involves Designing a rationale for selecting a decentralized application (dApp) development platform is a crucial step in the development process. Your choice of platform will significantly impact the functionality, scalability, security, and overall success of your dApp. For example, blockchain platforms in the healthcare domain include Ethereum which is one of the most popular blockchain platforms for dApp development, and it has been used for various healthcare applications such as patient records management, drug traceability, and telemedicine. Other platforms include Hyperledger Fabric, Corda, and MediBloc.
* Choosing a Blockchain Data Structure involves choosing the right blockchain data structures is a critical aspect of designing and building a blockchain system. The choice of data structures can significantly impact the blockchain's efficiency, security, and functionality.
* Design & Implement Blockchain and Mapping onto Reference Architecture involves Designing a blockchain-as-a-service (BaaS) platform involves creating a set of components and services that make it easier for developers and organizations to implement and use blockchain technology. Adopt a design strategy for creating a BaaS platform, implementing blockchain technology, and mapping it onto a reference architecture.
* Developing smart contracts involves Developing smart contracts systematically involves a structured approach to ensure the reliability, security, and efficiency of your blockchain-based applications. Adopt a systematic process for developing smart contracts such as defining the user requirements with use cases and BPMN as presented earlier, specifying smart contracts, choosing implementation platforms, and validating & verifying smart contracts.
* Choosing a front-end framework involves selecting the right front-end framework for blockchain applications is a crucial decision that impacts user experience, development efficiency, and the overall success of your project. By following a specific design rationale, you should make an informed decision when selecting a front-end framework for your blockchain application, ensuring that it aligns with your project's requirements, development capabilities, and user expectations.
* Starting the testing cycle involves testing blockchain decentralized applications (dApps) requires a systematic approach and specialized test techniques due to the unique characteristics of blockchain technology. Adopt a systematic process and test techniques to start the testing cycles for blockchain dApps such as defining test objectives, creating test plans, and test environments, and creating test cases, test techniques including unit testing, functional testing, integration testing, and energy-efficiency testing.

Figure 9 Design & Development Process for dApps

Data structures play a critical role in various computer science and distributed systems, and when comparing and evaluating them, it's essential to consider their design, purpose, and performance characteristics. In this response, I'll compare and critically evaluate several data structures, including Blockchain, GHOST (Greedy Heaviest Observed Subtree), BlockDAG (Directed Acyclic Graph), and Segregated Witness, in terms of their design, use cases, and notable features. Table 1 on Blockchain Data Structures provides a list of data structures such as Blockchain, GHOST, BlockDAG, and Segregated witness.

|  |  |  |  |
| --- | --- | --- | --- |
| Blockchain Design Methods and Its Data Structures | Design Rationale | Use Cases | Notable Features |
| Blockchain | Blockchain is a distributed, append-only ledger that relies on a linear chain of blocks, where each block contains a list of transactions. Blocks are linked through cryptographic hashes, ensuring the integrity and immutability of the ledger. | Blockchain is popular in applications like cryptocurrencies (e.g., Bitcoin), supply chain management, and smart contracts (e.g., Ethereum) | Blockchain provides strong security through Proof of Work (PoW) or Proof of Stake (PoS) consensus mechanisms, but it can suffer from scalability and latency issues due to its linear structure. |
| GHOST (Greedy Heaviest Observed Subtree) | GHOST is a data structure used to resolve forks and reach consensus in Ethereum's blockchain. It considers not just the longest chain but also includes orphaned blocks, giving a more comprehensive view of the network. | GHOST is primarily used in blockchain networks that adopt Ethereum's protocol for consensus. | GHOST enhances security by considering more blocks in the consensus process but can lead to increased complexity in certain cases. |
| BlockDAG (Directed Acyclic Graph) | BlockDAG is a data structure that allows multiple blocks to reference one another in a directed acyclic graph rather than a linear chain. This structure eliminates the need for a single, global consensus point. | BlockDAG is used in cryptocurrencies like IOTA and Nano to address scalability and throughput issues associated with traditional blockchains. | BlockDAGs offer improved scalability and reduced confirmation times compared to traditional blockchains. However, they require different consensus algorithms, such as Tangle or DAG-based PoW. |
| Segregated Witness (SegWit) | SegWit is a data structure upgrade for Bitcoin. It separates transaction data and witness data, allowing for more efficient use of block space and fixing transaction malleability issues. | SegWit is specific to the Bitcoin network and aims to improve its scalability and security. | SegWit reduces the size of transactions, enabling more transactions to fit in a block. This helps reduce transaction fees and enhance the overall efficiency of the Bitcoin network. |

Table 1 Blockchain Data Structures

## 6.1 Critical Evaluation of Blockchain Data Structures

Critical evaluation of the various blockchain data structures should be based on non-functional requirements identified for your business strategy such as security, scalability, use cases, complexity, and adoption as follows:

* Scalability: BlockDAGs and SegWit address scalability challenges more effectively than traditional blockchains like Bitcoin. They enable higher throughput and lower transaction costs.
* Security: Blockchain and GHOST provide strong security through their consensus mechanisms. SegWit enhances security by reducing the risk of transaction malleability attacks.
* Complexity: GHOST introduces complexity by considering orphaned blocks. BlockDAGs are more complex to implement due to the lack of a linear chain.
* Use Cases: The choice between these data structures depends on the specific use case. BlockDAGs are suitable for high-throughput applications, while SegWit is tailored for Bitcoin's needs.
* Adoption: Blockchain and SegWit have seen widespread adoption, while GHOST and BlockDAGs are adopted in specific blockchain networks.

Marchesi, M. et al. (28) propose several Agile practices based on user stories and UML modeling. They also argue Agile methods are well suited for dApps development as they offer for self-organised teams and requirements are not well understood initially. They also proposed other Agile techniques such as Continuous Testing, Test Driven Design, Refactoring, Continuous Integration, Collective code ownership, Information Radiators (Cards, Boards, Burndown charts), Coding Standards, and Pair Programming (in some cases).

In conclusion, the choice of a data structure depends on the requirements and goals of a given blockchain or distributed ledger system. Each of these data structures has its advantages and disadvantages, and selecting the right one involves considering factors like scalability, security, complexity, and the intended use case. Furthermore, the landscape of blockchain technology is continually evolving, with innovations and data structures emerging regularly.

## 6.2 Component-Based Software Design for Blockchain

Component-Based Software Design for Blockchain is important as it provides a natural extension of service by providing required and provider services to other blocks in the blockchain. Component-based software design is crucial for blockchain development as it offers a systematic approach to building complex blockchain systems. This approach provides a natural extension of services, enabling required services to be provided to other blocks in the blockchain. Here are several reasons why component-based software design is essential for blockchain:

1. Modularity and Reusability: Component-based design promotes modularity by breaking down a blockchain system into smaller, self-contained components or smart contracts. These components can be reused in various parts of the blockchain or even in different blockchain applications. This reusability saves time and resources in development.
2. Service Composition of Blockchain: Blockchain systems can leverage a wide range of services, including oracles, data feeds, identity management, and more. Component-based design enables the composition of these services, allowing for the creation of complex, feature-rich dApps (decentralized applications). An example of a blockchain service component is shown in Figure 10 which is a service component model for Data Analytics consisting of several provider services (shown as Lolli pop symbol) such as Interface on Data Pre-Processing (IDataPreProcessing), etc.

A diagram of a component

Description automatically generated

Figure 10 Blockchain Service Component Model

### 6.2.1 Blockchain Security

Blockchain has arisen as a potent security solution for various applications, primarily due to one of its fundamental attributes: immutability. Regrettably, Destefanis et al. (11) highlighted vulnerabilities found in smart contract libraries and the insecure programming of smart contracts. Consequently, there is a growing need for the advancement of Blockchain Software Engineering to address these issues. Therefore, this paper proposes, as shown in Figure 11, a concept of security service smart contract component and architecture which is a reusable smart contract and can be plugged into across applications. This improves the security, scalability, and immutability of smart contracts.

A screen shot of a computer security service

Description automatically generatedFigure 11 Blockchain Service Component for Security Smart Contract

The blockchain service components model for security smart contracts as shown in Figure 11 provides several required interfaces such as ISignature for identity management, IEncryption for encryption for cryptographic algorithms, etc. The following section is devoted to presenting the reference architecture for dApp development for standardization and sustainability.

## 6.3 Reference Architecture for Blockchain (REF-ArcBC)

The integration of blockchain, AI, and IoT technologies can enable the development of powerful and innovative applications that can transform various industries. To standardize the development of AI-enabled blockchain applications and blockchain-driven AI applications requires a reference architecture. A reference architecture for blockchain applications is a standardized blueprint or framework that provides a structured and well-defined approach to designing and building blockchain-based solutions. It serves as a foundational guide for developers, architects, and organizations looking to leverage blockchain technology effectively. Here's a brief introduction to the importance and meaning of a reference architecture for blockchain applications:

* Standardization and Consistency: A reference architecture establishes a common set of design principles, best practices, and components, ensuring consistency across different blockchain applications. This standardization streamlines development and maintenance processes.
* Efficiency: It helps developers avoid reinventing the wheel. By following a reference architecture, they can leverage pre-established patterns and components, reducing development time and costs.
* Interoperability: Reference architectures often consider interoperability with existing systems and other blockchain networks. This is vital for ensuring that blockchain applications can seamlessly work with other technologies.
* Security: Security is a paramount concern in blockchain applications. A reference architecture typically incorporates security best practices and guidelines, which helps in reducing vulnerabilities and risks.
* Scalability: As blockchain applications grow, scalability becomes a critical factor. A reference architecture guides how to design systems that can easily scale to accommodate increased loads.
* Adoption and Collaboration: A reference architecture facilitates collaboration and adoption by providing a common framework that different organizations and developers can use. This leads to faster adoption of blockchain technology.
* Regulatory Compliance: Compliance with regulatory requirements is a significant challenge in the blockchain space. A reference architecture may include guidelines on how to design systems that adhere to relevant regulations.
* Flexibility: While offering a structured approach, a reference architecture is often flexible enough to accommodate variations based on specific use cases or industry requirements.

In summary, a reference architecture for blockchain applications is essential for establishing a standardized, efficient, and secure foundation for developing and implementing blockchain solutions. It ensures that blockchain technology is used consistently and effectively, promoting interoperability, security, and scalability, while also aiding in regulatory compliance and encouraging wider adoption.

The reference architecture for blockchain is illustrated in Figure 12 as Reference Architecture for Blockchain (REF-ArcBC). REF-ArcBC consists of four layers namely: BC AI and IoT Application & Prediction Layer, Application Layer, Blockchain Layer, and Infrastructure Layer. The application layer for such applications typically consists of the following components as shown in Figure 12.

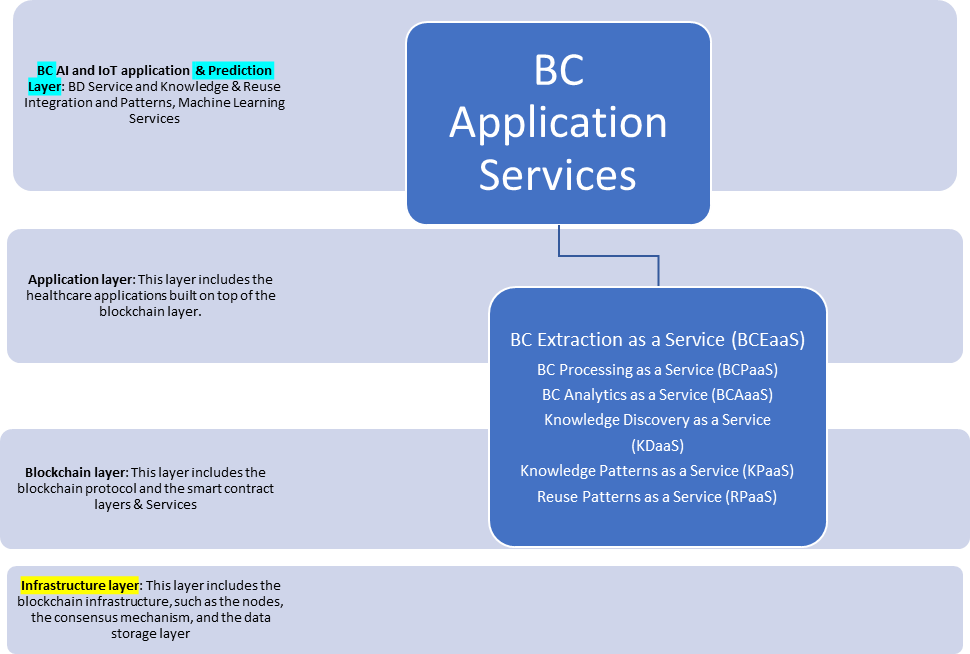
* Blockchain Layer: This layer includes the blockchain network and infrastructure that stores and validates data. It ensures the immutability, security, and transparency of the data by using cryptographic algorithms, consensus mechanisms, and smart contracts.
* AI Layer: This layer includes machine learning algorithms, neural networks, and other AI models that enable the analysis and processing of large amounts of data. It can be used for various purposes, such as predictive analytics, anomaly detection, and natural language processing.
* IoT Layer: This layer includes the physical devices, sensors, and gateways that collect and transmit data to the blockchain and AI layers. It can enable real-time monitoring and control of various systems and processes, such as smart homes, smart cities, and industrial automation.
* Integration Layer: This layer includes the middleware and APIs that enable the integration and interoperability of the blockchain, AI, and IoT layers. It can ensure the seamless exchange of data and transactions between the different layers, enabling the development of complex and decentralized applications.
* Application Layer: This layer includes the user interfaces, dashboards, and other applications that enable users to interact with the blockchain, AI, and IoT layers. It can provide various features and functionalities, such as data visualization, decision-making support, and automation.

These layers can be combined and customized based on the specific requirements and use cases of the applications. The integration of blockchain, AI, and IoT technologies can enable the development of innovative solutions that can address various challenges and opportunities in different industries. These are just a few examples of the services provided by the blockchain application layer. The actual services and functionalities depend on the specific use cases and requirements of the applications. The blockchain layer in blockchain technology refers to the underlying network and infrastructure that stores and validates data using cryptographic algorithms, consensus mechanisms, and smart contracts. Some of the services that happen at the blockchain layer include decentralized data storage, data validation and consensus, smart contract execution, cryptographic security, tokenization, digital asset management, interoperability, and integration.

These are some of the services that happen at the blockchain layer in blockchain technology. The actual services and functionalities provided by the blockchain layer depend on the specific use cases and requirements of the applications. The infrastructure layer in a blockchain reference architecture provides the underlying technical infrastructure that supports the blockchain network and enables the execution of smart contracts and other decentralized applications. The services provided by the infrastructure layer include:

* Node Management: The infrastructure layer provides services for the management of the nodes that participate in the blockchain network. These services include node registration, node discovery, node synchronization, and node communication.
* Network Consensus: The infrastructure layer provides services for the consensus mechanism that enables multiple nodes in the network to validate and agree on the data and transactions stored on the blockchain. These services include consensus algorithm implementation, block creation, block validation, and block propagation.
* Data Storage and Retrieval: The infrastructure layer provides services for the storage and retrieval of data on the blockchain network. These services include block storage, transaction storage, and data retrieval through APIs.
* Smart Contract Execution: The infrastructure layer provides services for the execution of smart contracts and other decentralized applications on the blockchain network. These services include smart contract development, deployment, and execution.
* Security and Privacy: The infrastructure layer provides services for the security and privacy of the blockchain network and its participants. These services include cryptographic mechanisms, access control mechanisms, and identity management services.
* Interoperability and Integration: The infrastructure layer provides services for the interoperability and integration of the blockchain network with other networks and applications. These services include inter-chain communication, cross-chain transaction support, and application programming interfaces (APIs).
* Scalability and Performance: The infrastructure layer provides services for the scalability and performance of the blockchain network. These services include sharding, sidechains, and other techniques that enable the network to handle large volumes of data and transactions.

These are some of the services provided by the infrastructure layer in a blockchain reference architecture. The actual services and functionalities depend on the specific use cases and requirements of the blockchain application.



Blockchain Service Bus

Figure 12 Reference Architecture for Blockchain (REF-ArcBC)

It's crucial to recognize that assessing intricate reference architectures like REF-ArcBC demands a multi-year effort to comprehensively gauge their performance in real-world scenarios and assess their adaptability for making global smart contract revisions. Consequently, the subsequent sections delineate a method for evaluating REF-ArcBC, which centers on a real-world case study concerning a chatbot, an application of conversational AI. This evaluation approach encompasses the utilization of Business Process Modeling Notation (BPMN) tools, specifically leveraging Bizaghi for modeling and simulation.

BD Enterprise Service Bus (BD-ESB)

Reference

1. Ramachandran, M (2023) S3EF-HBCAs: Secure and Sustainable Software Engineering Framework for Healthcare Blockchain Applications, International Journal of Blockchain in Healthcare Today (BHTY), Full Open Access Journal, September 2023, <https://blockchainhealthcaretoday.com/index.php/journal>, https://doi.org/10.30953/bhty.v6.286