OpenDSU BluePrint Architecture

Author: Sînică Alboaie, PhD, Axiologic Research
Purpose: OpenDSU BluePrint Architecture
Visibility: Public
Date: May 2024
Version: 0.1

1. Architecture Blueprint

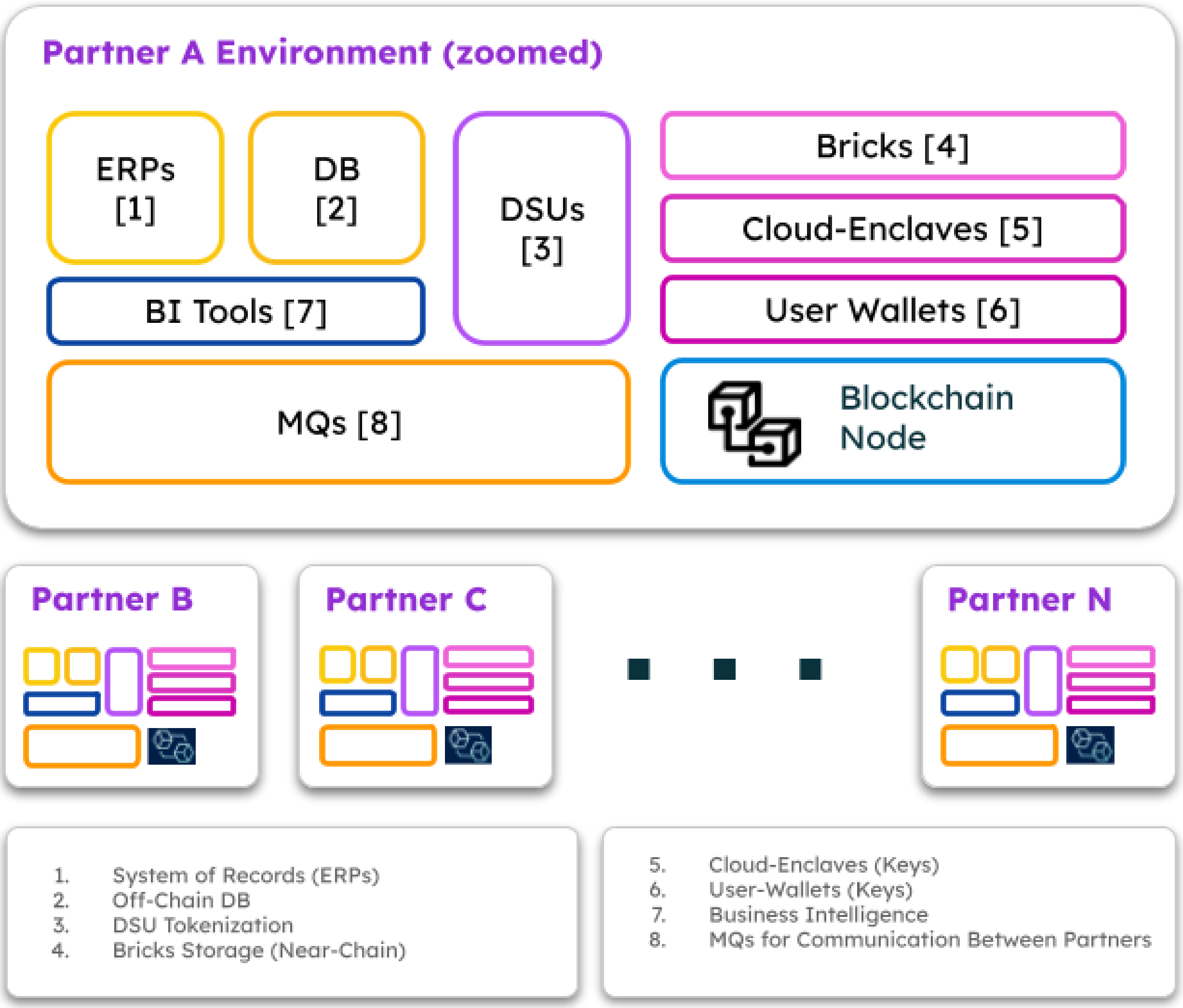

Diagram 1: Architecture Blueprint

The above diagram presents a generic architecture for an OpenDSU application, illustrating the intricate interplay of various components integral to its functionality, each labelled from 1 to 8 for clarity.

Component 1, labelled 'System of Records,' represents the core application servers, including complex Enterprise Resource Planning systems like SAP. These primary organisational data repositories are crucial for maintaining business operations and records. In some cases, the DSU components are the 'master data,' but in others, the enterprise data source remains in legacy systems.

Moving to component 2, we have the 'Off-Chain Database.' This component is pivotal in storing data not recorded on the blockchain, offering a complementary data management solution that works with the blockchain infrastructure.

Component 3, 'DSU Tokenization,' involves importing data into Data Sharing Units (DSUs). This step is critical for converting organisational data into a format suitable for blockchain integration, ensuring seamless and secure data tokenisation. This component can take different forms, from APIs to enterprise wallets.

Next, component 4, 'Bricks Storage or Near-Chain,' refers to the OpenDSU storage solution close to the blockchain. It gives the anchored data blockchain-like properties and is designed for efficient storage and retrieval, bridging the gap between on-chain and off-chain data environments.

Component 5, 'Cloud-Enclaves for Enterprise Key Management,' highlights the system's security-enforcing mechanism around cryptography and storage of sensitive data. It involves managing cryptographic keys in a secure cloud environment, ensuring that data access and transactions are protected and controlled.

For component 6, we have 'User or Enterprise Wallets.' This component utilises cloud enclaves for managing digital wallets, essential for transactions and interactions within the blockchain network. It could also take the form of APIs or OpenDSU digital wallets.

Component 7 is the 'Business Intelligence Component,' an optional yet valuable component that could provide analytical capabilities. It enables the extraction of insights and intelligence from blockchain data, enhancing decision-making processes. The

Finally, component 8 emphasises the 'MQs for Communication Between Partners.' This component is essential for facilitating secure and efficient communication between network participants. It recommends using Decentralized Identifiers (DIDs), which are secured with verifiable credentials, ensuring a high level of security and trust in communications.

These components illustrate a blockchain application network's complex and multifaceted nature. Each plays a specific role in ensuring the system's overall functionality and efficiency.

2. Architectural Pattern: Pull pull-encrypted bricks from partners

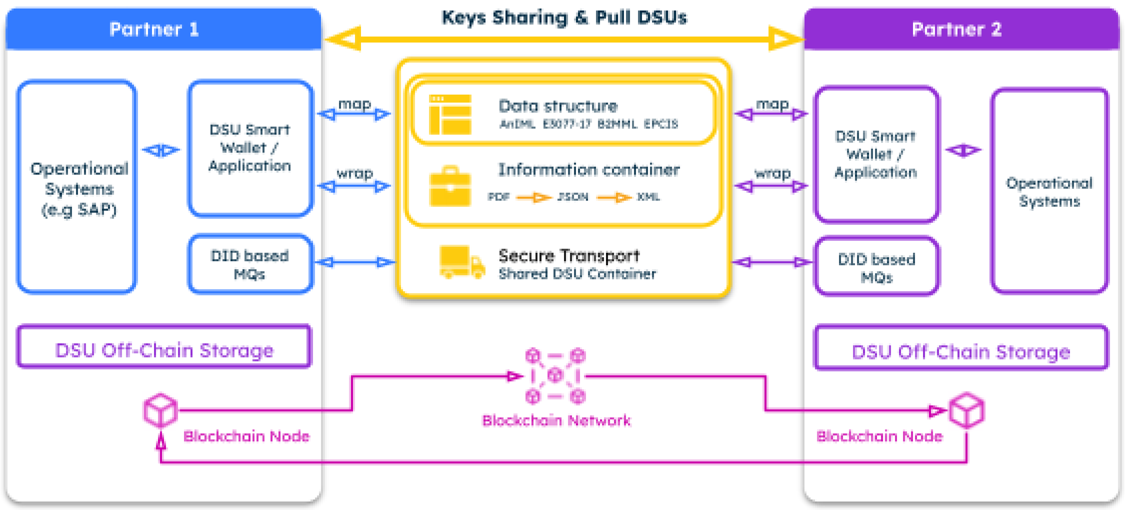

Diagram 2: Implicit Communication through DSU states

In the above diagram, we present the high-level approach of OpenDSU-based communication, which is crucial for sharing data about the states of business processes between two partners. This approach can be generalised to any number of partners. The image exemplifies a generic architecture for how two entities, each with their operational systems, explain that DSUs are stored in the network and shared based on cryptographic keys. The main architectural pattern we want to explain is that each organisation will pull the bricks and reconstruct the DSU locally.

Initially, Partner 1 and Partner 2, utilising systems such as SAP, begin their collaborative or "choreography" (a decentralised process or flow). Their respective DSU Smart Wallets/Applications act as intermediaries in this process, skillfully handling and preparing the data for transfer. This data is then processed and transformed into formats, for example, EPCIS events, and wrapped in universally accepted formats like XML, PDF, and JSON. This conversion is critical to ensure seamless compatibility and data integrity across different systems.

At the heart of this exchange is MQ based on Decentralized Identifiers (DIDs), establishing a secure and verifiable link between the partners. This secure connection fosters reliability and maintains the confidentiality and integrity of the shared data. The procedure further encompasses the exchange of keys and the transfer of Data Sharing Units (DSUs) between the partners, an essential step for maintaining robust access control and data security.

To guarantee the safe transit of this data, DID-based Message Queues (MQs) are utilised, ensuring data protection throughout its journey. Upon reaching its destination, the data can be used or stored in databases, while the original data from the DSU (the off-chain data) remains with the original creator. This arrangement is connected to a Blockchain Network, which provides an immutable and traceable record of all transactions and interactions involved in the data sharing.

The Blockchain Network, complemented by its nodes, strengthens the system, offering a decentralised and secure environment. This framework not only bolsters the security of the data exchange but also enhances the trust and reliability of the entire process.

With this chapter, we tried to offer a clear overview of how OpenDSU enables effective data sharing and management of business process states between partners, underscoring the efficiency, security, and dependability of this “pull-encrypted data when-needed” method of sharing & communication.

2. Internal Data Flows

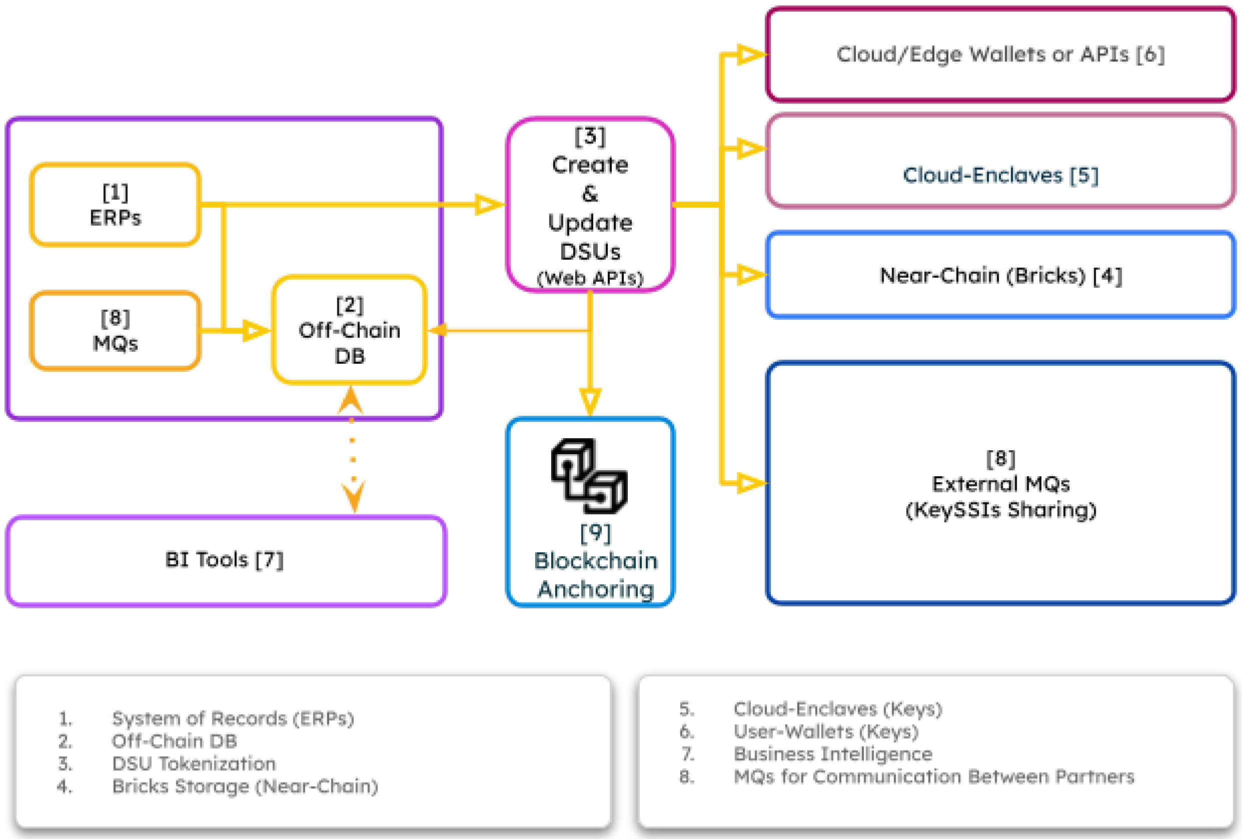

Diagram 3: Typical Internal Data Flow

The typical flow for anchoring new data in DSUs inside of a company is illustrated in the diagram above, highlighting that new entries in component 2 (Off-Chain DB) are initiated by component 1 (System of Records) or component 8 (MQ message queue received from a partner). This process triggers the creation and update of DSUs in component 3 (DSU Tokenization). This results in the creation of new bricks in component 4 (Bricks Storage), functioning as Near-Chain in OpenDSU terminology, and also leads to anchoring in the blockchain. Additionally, component 3 could add new keys in enclaves, which can be either type 5 (Cloud-Enclaves) or type 6 (User-Wallets). These activities may also prompt further updates and data requests to other partners since component 3 typically serves as an API that can handle any necessary business processes.

The architecture assumes the use of MQs based on W3C DID for communication between partners, supported by BDNS for routing to the appropriate partner. However, this is not mandatory, as simple endpoint calls to partners can be easily implemented and can, in turn, use BDNS configurations to determine the partner or partners that need to be notified.

3. Hosting resources

The following table estimates the hosting resources required for an OpenDSU-based solution. Each component is evaluated based on its specifications and expected load. This resource estimation helps plan the deployment and ensure the system's performance.

Component	**Resource Estimation**
1 - System of Records (ERPs)	It is not part of our specification
2 - Off-Chain DB	By default, we use a very light, embedded database (called LightDB) that saves data on files. However, a proper database is recommended for complex use cases.
3 - DSU Tokenization	These are custom APIs, and they must be planned according to the expected volume.
4 - Bricks Storage (Near-Chain)	It is a small set of APIs that depends on the expected load, the primary factor being the size of the upload bricks in the specific use cases. Each brick has a limit of 15 megabytes, but if many such bricks are downloaded or uploaded, it will pressure server memory.
5 - Cloud-Enclaves	The default one that can be used in APIs is very light, uses component 2 in a lazy mode, and has no special requirements.
6 - User-Wallets	They are executed in browsers or mobile applications
7 - Business Intelligence	It is not part of our specification
8 - MQs	By default, the MQ is a small set of APIs that uses component 2 to store messages. However, in complex use cases, it could be recommended that a more sophisticated message queue be replaced.
9 - Blockchain	The blockchain itself is not part of our specification. However, depending on the blockchain technology, a particular server called a “Blockchain Adapter " must be deployed for each blockchain type. Typically, it is just a set of APIs around the blockchain SDK that adapt the generic anchoring APIs expected by the OpenDSU SDK to the actual smart contract.

Table 1: Hosting Resource Considerations

On the server side, OpenDSU can be executed on systems with minimal resources, depending primarily on implementing custom APIs and the expected load on these APIs or components like the Off-Chain DB, Bricks Storage, and anchoring. Proper planning and resource allocation for these components will ensure efficient and scalable deployment of OpenDSU-based solutions.

For example, our previous testing in the PREQA and QA environments was successfully executed using the Quorum blockchain on an AWS EC2 t3.large instance with the following specifications: vCPUs: 2, Memory: 8 GiB, Network Performance: Up to 5 Gigabit. This demonstrates that even moderate cloud instances can effectively handle the requirements of an OpenDSU-based solution. We recommend more resources for production systems, but they must be tested for each cloud, infrastructure and use case.