RE4DY

MANUFACTURING DATA NETWORKS

RE4DY TOOLKIT

Name of the Tool	Data Provenance and Traceability application
Tool Owner	Industry Commons Foundation
Version	1.0
Date	Nov 2025
Version	V1.0

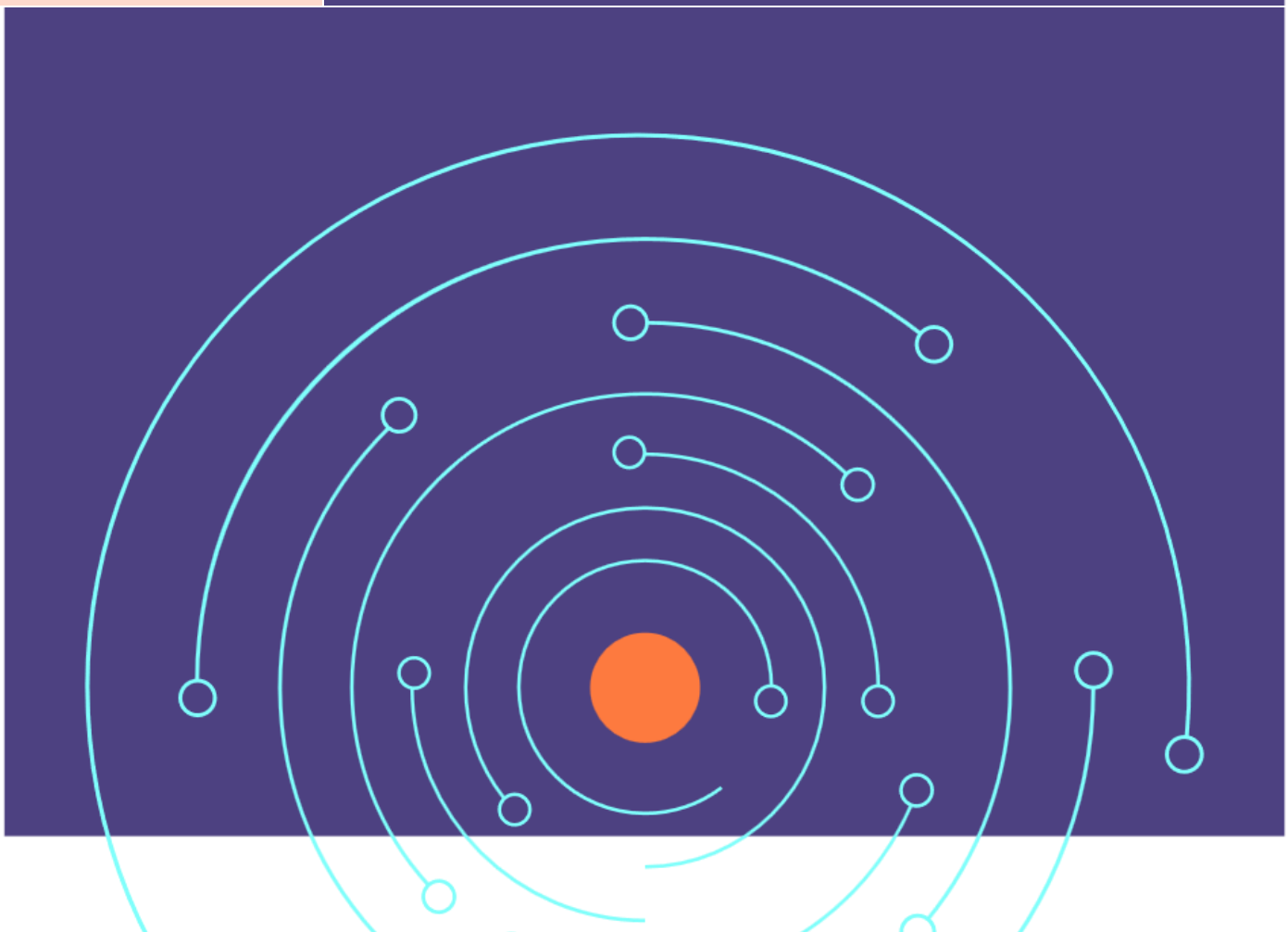

Table of contents

1. Component Description

Hyperledger Fabric is an open source blockchain developed under the Hyperledger project of the Linux Foundation. It is mainly aimed at organizations wishing to deploy permissioned blockchain networks. Its modular architecture permits the customization of the blockchain network to meet specific needs. An example of this is the support for pluggable consensus algorithms. Moreover, it offers communication privacy through channels, a membership mechanism to restrict channel access and access control lists (ACLs) to handle access to resources using various policies. It supports the deployment of smart contracts (chaincode) which allow the secure execution of custom processes and algorithms that modify the state of the network.

Within the context of the RE4DY project, Hyperledger Fabric will be used to deploy a permissioned blockchain and leverage its immutable and append-only transaction history for the implementation of our data provenance and traceability application. In particular, we will use the functionality offered by Hyperledger Fabric smart contracts to create a dataset lifecycle tracking system.

Communication with the Hyperledger Fabric network will be executed through a separate application responsible for communicating with the Fabric Gateway, an API running within network peers with the purpose of facilitating transaction submission.

Furthermore, Hyperledger Fabric is expected to have a minor role in the RE4DY user authentication process. Through a process initiated by the RE4DY authentication backend implemented in Spring Boot, RE4DY accounts will acquire certain attributes and will be associated with Hyperledger public addresses. Using Attribute-Based Access Control (ABAC) these addresses will be either allowed to or restricted from invoking chaincode methods that insert and update dataset information on-chain.

2. Input

As the chaincode will serve as a foundation for data provenance and traceability, the majority of the smart contract methods manage data entry on-chain. Depending on the contract method to be called, sending a transaction involves procuring various information regarding a dataset or its traceability events as transaction input. For instance, when recording the creation of a new dataset, it is necessary to provide a dataset ID, a number of dataset characteristics and a list of related datasets. Regarding retrieving on-chain data, Hyperledger Fabric supports JSON queries and key-based queries in its world state, a database that holds the current values of all objects.

3. Output

In Hyperledger Fabric, for each transaction invocation, the calling application is notified whether the transaction was valid, meaning it has been included in the chain, or invalid. For example, upon successful smart contract method invocation, the Fabric Gateway returns "Chaincode invoke successful" and includes the status code of the request, using the HTTP response status codes defined by RFC 9110. Finally, in cases where a value is returned with the function call, it is returned in JSON format.

4. Information Flow

Creating a dataset

A user may create a dataset by providing a dataset ID, a list of related dataset IDs and a number of dataset characteristics. Sending a transaction to the blockchain is required for such an operation. The tracking and management of the transaction lifecycle is the responsibility of the Fabric Gateway, which decouples the submission logic from the application where the transaction originates from. Initially, the client application sends a transaction proposal to the Fabric Gateway. The transaction proposal includes the name of the contract method to be invoked and optionally a number of arguments to the method. The Fabric Gateway forwards the transaction proposal to the Fabric Network. Peers in the Fabric Network execute the transaction locally to determine its validity according to the rules of the smart contract. If the transaction is valid, the peers endorse the transaction proposal. Depending on the policies agreed to by the members of the channel when the chain code definition was approved, a certain number of endorsements are required. After the specified number of endorsements is collected, the Gateway submits the transaction and its endorsements and waits for commit status events. During this time, the new block which includes the transaction is propagated in the network and the peers validate all transactions included in it. When this happens, the transactions are committed to the ledger and the chain state is updated to reflect the changes made by these transactions. Finally, the Gateway offers the capability to access a minimal API for any client application to receive and handle chain code events.

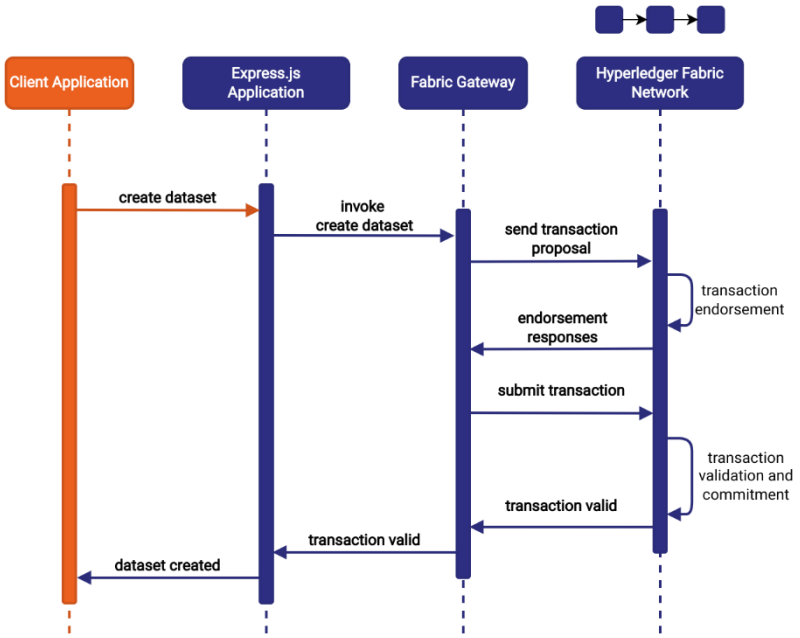

Figure 1 UML Diagram of the product creation process starting from the client application to the low-level transaction validation procedures taking place in the Fabric Network

Reading a dataset

A user provides a dataset ID in order to retrieve information for the related dataset. The Gateway will invoke the smart contract method which queries the state of the chain. The method will be executed on one Fabric Network peer and the function result will be returned to the Gateway and then the client application. It is noted that the Gateway will prefer a peer which is part of the same organization as the Gateway peer and select the peer with the highest block height, meaning the peer with the most up to date state of the chain. Also, the Gateway will ultimately select a peer from a different organization if no other peer is available in its own.

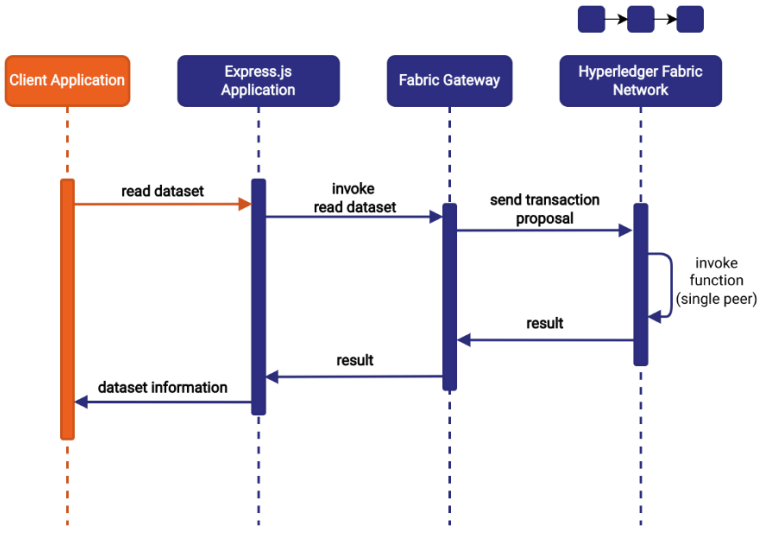

Figure 2 UML Diagram of the invocation of the product reading method of the RE4DY chaincode from the client application to the Fabric Network and back.

5. Internal Architecture

The Fabric Network is composed of a number of peers running peer software to participate in it.

The Fabric Gateway is a service running in every peer which offers the capability to use a minimal API for transaction submission. This facilitates the interaction with the Fabric Network as clients are not required to perform actions such as collecting transaction endorsements from other peers. Our Express.js application transacts with the Fabric Gateway and offers a single-entry point to the Fabric Network. Any client application as well as the authentication back-end may then interact with the chain using the Express.js application.

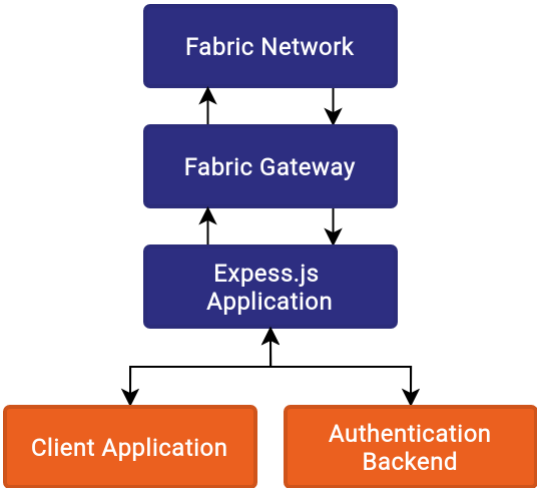

Figure 3 Internal architecture of the Hyperledger Fabric component and related services

6. API

The documentation of the Hyperledger Fabric component API is included in the Annex as a Swagger editor .yaml file.

7. Implementation Technology

In Hyperledger Fabric, chain code may be written either in the Go programming language, in JavaScript, using the node.js runtime environment and in Java. We have opted for JavaScript due to its ease of use and its large popularity in the Hyperledger Fabric community. Furthermore, there are application Software Development Kits (SDKs) available in JavaScript, Java, Python and Go. We will develop an application using Express.js which will implement an API that will provide an entry point to our Hyperledger Fabric blockchain network by interacting with it through the Fabric Gateway. In particular it

will be used to invoke contract methods which encompass Create, Read, Update, Delete (CRUD) operations for datasets and related data structures.

Hyperledger Fabric peer software is offered as a collection of Docker images and helper scripts which may be used to set up and connect to a private Fabric network, deploy smart contracts, invoke chain code methods, or query the chain state. Peer software is required for each participating node. Moreover, we have created new auxiliary scripts to facilitate development. In particular, we developed a Bash script for redeploying a certain smart contract without being forced to destroy and recreate the chain and a script for sending transactions as certain peers to simulate transactions from different users or users of different organizations.

8. Comments

The design and implementation of this component is currently in progress and its integration with the rest of the RE4DY tools is currently ongoing.