FUJITSU

# Proof of Concept on Distributed Ledger for CSD-RTGS Linkages

# Fujitsu's Technical Report

Fujitsu Limited

## Contents

# 1. Introduction

## 1.1 Background

Fujitsu Limited has participated in a POC of a securities settlement system launched by the Asian Development Bank (ADB) to improve the efficiency and security of cross-border transactions in securities in the Asia-Pacific region using blockchain technology.

Currently, cross-border transactions in the ASEAN+3 region are processed through custodians and a global network of correspondent banks (via global centers in the US and Europe). The more intermediary banks there are, the higher the transaction costs and the longer it takes to settle the transaction, resulting in issues such as time lag in a settlement. In addition, since it is difficult to track these fees and timestamps accurately, there is uncertainty as to whether the recipient has successfully received the funds.

The POC hypothesized that by directly connecting related institutions (ASEAN+3 central banks and securities settlement institutions) using blockchain technology, it would be possible to "reduce credit and settlement risks from transaction execution to delivery of securities and money" and "reduce time lag in cross-border transactions due to time difference and other factors. The POC consisting of ADB and blockchain technology vendors was formed and worked on for the purpose of determining the feasibility of the hypothesis and identifying issues toward its realization.

For a detailed overview, please check the ADB website.

## 1.2 Overview

Three economies with different financial regulations and systems as shown in the figure below were examined as typical configurations of CSD and RTGS systems.

- Economy A: The central bank manages the RTGS and CSD-G for international only, while CSD-C for other credits is operated and managed by the private sector (e.g., stock exchange-related organizations).
- Economy B: The central bank operates and manages the RTGS system only, while the private sector operates and manages the CSD-G&C for all credits.
- Economy C: The central bank operates and manages the RTGS system and the CSD-G&C of all credits (government and corporate bonds).

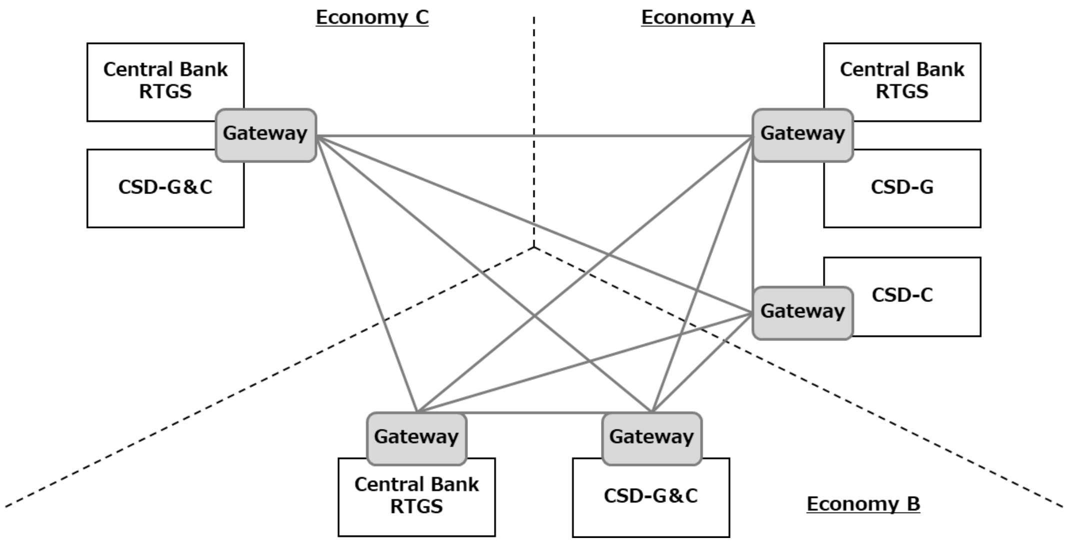


Figure 1: CSD-RTGS Linkage Model.

Adapted from "CSD-RTGS Linkage Model", Taiji Inui,

https://www.jsme2022f.jp/program/abstract/04_22f-inuitaiji.pdf

In this POC, the financial systems in each country were built with solutions from the blockchain vendors participating in this project (ConsenSys, R3, and Soramitsu), and the connection and coordination part between countries (between the systems of each vendor) was built with Fujitsu's solutions.

For more information on each vendor's solution, please visit each vendor's website.

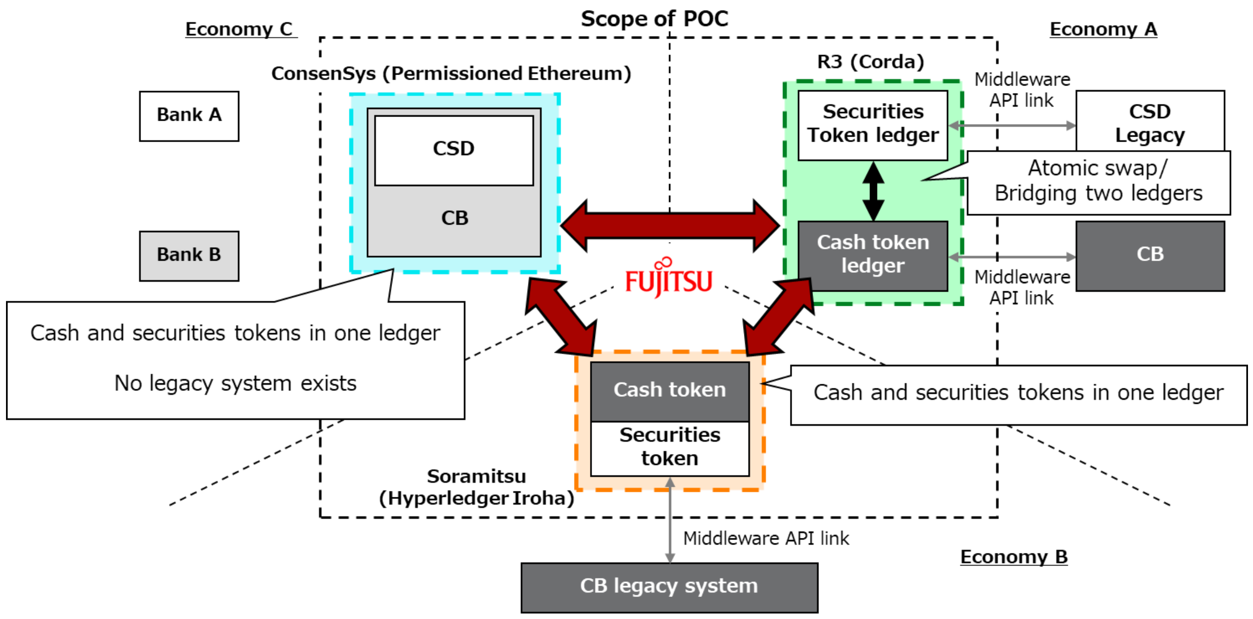


Figure 2: Scope of POC and Responsibilities of Each Vendor

## 1.3 Fujitsu's Role

Fujitsu roles implement an instantaneous DvP fund settlement using its blockchain technology, named "ConnectionChain", to intermediate DLT-based financial systems built by each vendor in each participating countries. ConnectionChain is a security technology that guarantees the transparency of transactions by securely linking blockchains with different implementation methods and operational objectives. ConnectionChain uses "Hyperledger Cacti" to integrate various DLT platforms for enabling execution of transactions across multiple different blockchains to be conducted securely and reliably.

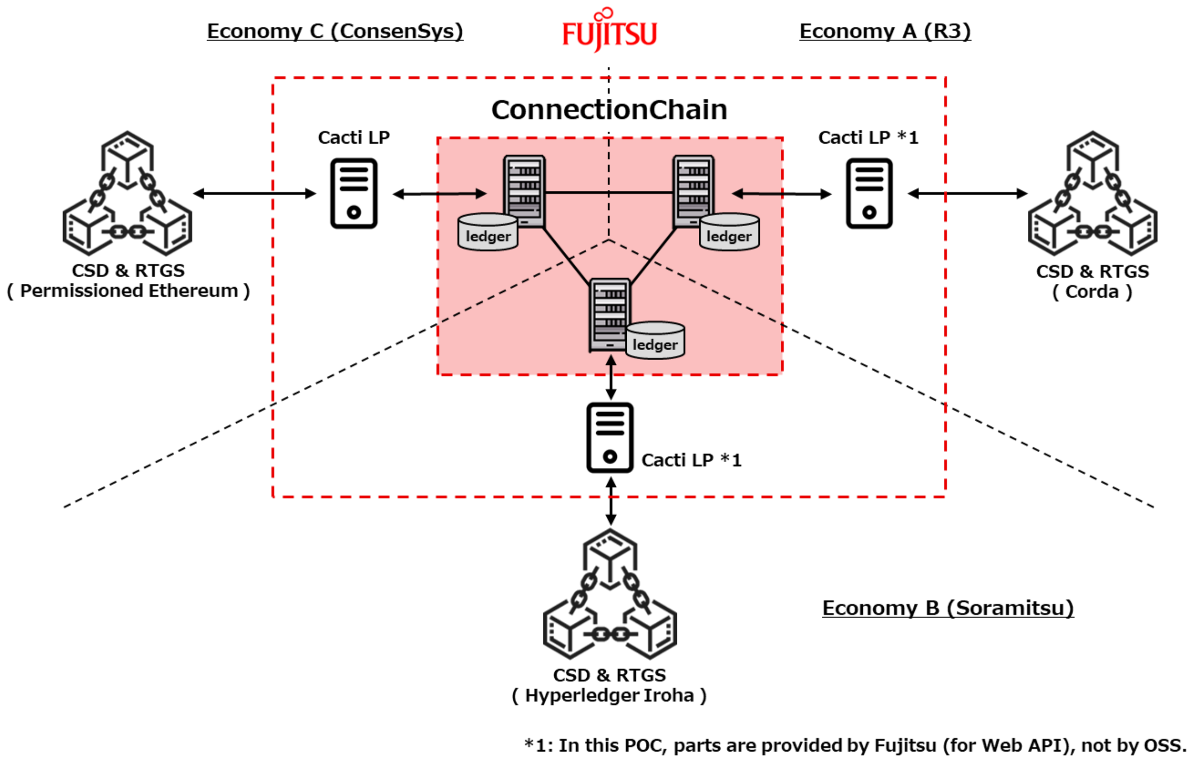


Figure 3: Collaboration by ConnectionChain and Hyperledger Cacti

## 2. Fujitsu's Solution (ConnectionChain + Hyperledger Cacti)

## 2.1 ConnectionChain

### 2.1.1 Abstract

ConnectionChain is an extension of smart contract technology that enables the automatic execution of a single-consistent transaction by connecting multiple blockchains with a new blockchain and the linking transaction processing related to the exchange of a series of currencies and other digital assets in each chain, and a transaction control technology that synchronizes the timing of transaction execution in each chain.

This technology ensures that all transaction processes are recorded as a trail in the connecting blockchain, even when crossing chains, thus guaranteeing the transparency of transactions.

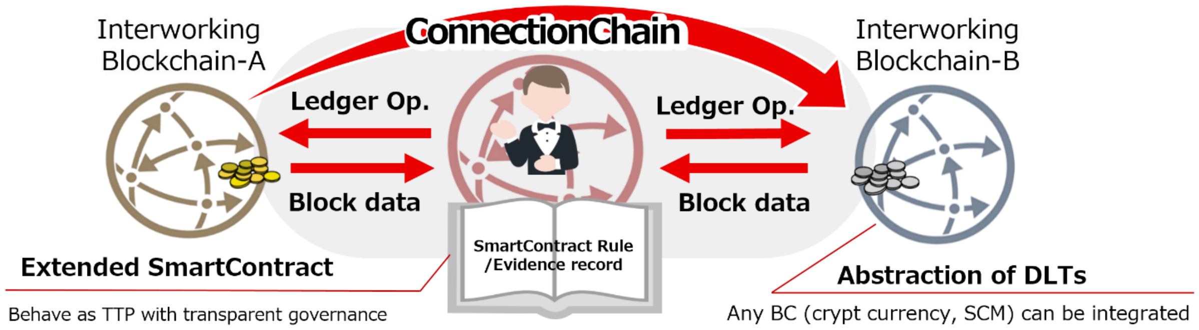

Interworking Blockchain-A
ConnectionChain
Interworking Blockchain-B
Ledger Op.
Block data
Ledger Op.
Block data
SmartContract Rule /Evidence record
Extended SmartContract
Behave as TTP with transparent governance
Abstraction of DLTs
Any BC (crypt currency, SCM) can be integrated

Figure 4: ConnectionChain Overview

### 2.1.2 Features

ConnectionChain has the following three features: Extended SmartContract, Abstraction of DLTs, and Multiple Scenario, which enable the exchange of various values by connecting multiple blockchains safely and securely.

- **Extended SmartContract**

To ensure the transparency of transactions, it is necessary to extend each smart contract that operates on the linked blockchain to processes involving multiple blockchains. Extended SmartContract can automatically execute remote operations on the linked blockchain and autonomously determine the next operation to be executed based on the success or failure of the operation. In addition, the history of all automatic operations is recorded in the connecting blockchain, ensuring the transparency of the linked transactions.

- **Abstraction of DLTs**

The Abstraction of DLTs is the function required for Extended SmartContract to work properly. This function is a technology that absorbs the differences between dozens of different blockchains and enables remote control and monitoring, and when combined with Extended SmartContract, it is possible to integrate blockchains that have been constructed and operated separately into a new service.

- **Multiple Scenario**

To integrate multiple blockchains and link them as a new service, agreed-upon operation rules are required. Multiple Scenario feature allows flexible rule-setting such as customizing other companies' services to your own by scripting of operation rules.

### 2.1.3 Architecture

ConnectionChain is composed of the following five architectures: ConnectionChain Ledger, CoreAPI, Ledger Plugin (LP) Server, Connected System, and Signature Server.

- **ConnectionChain Ledger**

ConnectionChain ledger is a consortium blockchain that manages the information used in the ConnectionChain and is composed of Hyperledger Fabric. The "environment information chaincode (smart contract in Hyperledger Fabric)" holds connected system information, and the "scenario management chaincode" holds the content and progress of the scenario to be executed and controls the scenario's operation transitions. In addition, the execution results of scenario execution and multiple systems are stored as scenario history information.

- **CoreAPI**

CoreAPI is an application that runs on a REST server and relays operations between connected systems. It performs connected system operation and event monitoring, analyzes operation results, and registers them in the scenario management chaincode using chaincode operations for ConnectionChain. The operation details of the connected systems are returned according to the scenario status and operation results from the scenario management chaincode. The scenario is monitored by the chaincode event monitoring, and operation requests are made to each connected system. Basically, the scenario progresses by repeating this process. In addition, a REST API is provided to register information to the chaincode for the ConnectionChain.

- **Ledger Plugin (LP) Server**

Ledger Plugin (LP) performs relay processing between the ConnectionChain and connected system. It converts operation requests from CoreAPI into execution requests according to the specifications of the destination system and notifies CoreAPI of the response and event operation results. It also uses filter information to determine whether events should be notified to CoreAPI or not. Filter information is defined in the scenario and is set via CoreAPI when the scenario is activated or an operation request is made. This experiment utilized the Ledger Plugin for Hyperledger Cacti, which is described below.

- **Connected System**

Connected System is an environment connected by ConnectionChain: an Ethereum private chain, a Hyperledger Fabric blockchain, a WebAPI server, and various other systems.

- **Signature Server**

Signature Server performs authentication, authorization, and signature for the connected system. It should only be built when authentication or signature information is required to operate the destination system. A function to issue access tokens tied to the destination system's account is required. Operations using the access token will depend on the specifications of the destination system.

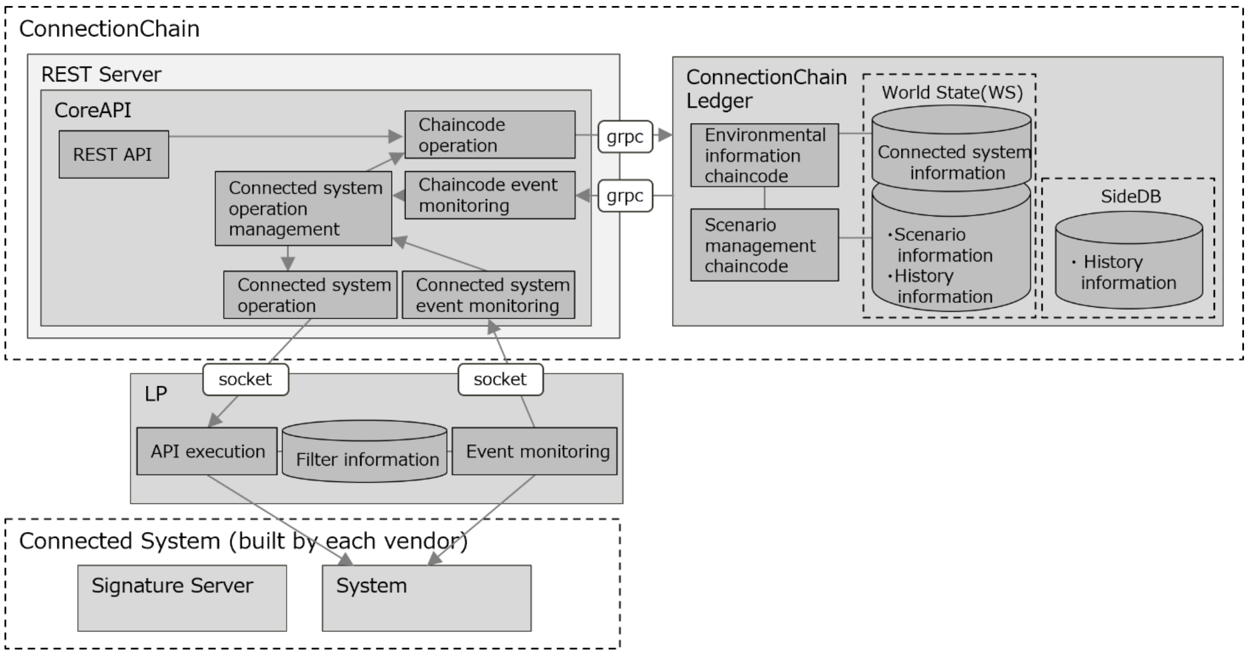


Figure 5: ConnectionChain Architecture

The scenario execution sequence is as follows.

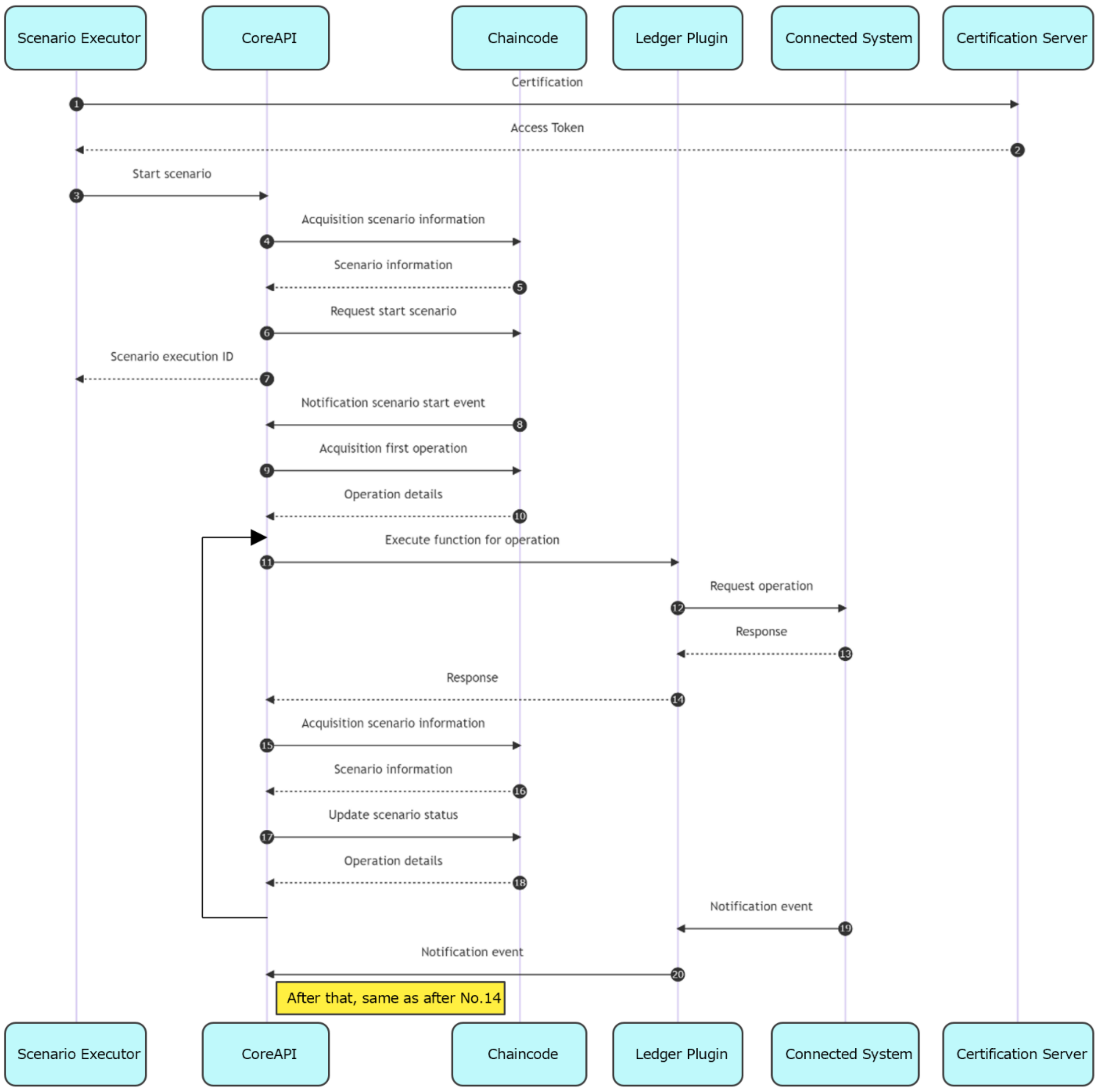


Figure 6: Scenario Execution Sequence

The press release is below:

https://www.fujitsu.com/global/about/resources/news/press-releases/2017/1115-01.html

## 2.2 Hyperledger Cacti

### 2.2.1 Abstract

Hyperledger Cacti (hereinafter called "Cacti") is the successor to the Hyperledger Cactus project. When new members joined from Weaver project, we chose this name, a plural form of Cactus. All the features that were in Cactus are also available in Cacti.

Cacti is a pluggable interoperability framework to link networks built on heterogeneous distributed ledger and blockchain technologies and to run transactions spanning multiple networks.

Absorbing differences such as the timing of when transactions are finalized on different blockchains, transactions between different assets can be performed safely and reliably on the blockchain, including the execution of necessary asset transfers and the execution of recovery processes in the event of failed transfers.

Please see below for details:

Hyperledger Cacti: https://www.hyperledger.org/use/cacti

Hyperledger Cactus: https://www.hyperledger.org/use/cactus

Whitepaper (Hyperledger Cactus):
https://github.com/hyperledger/cacti/blob/main/whitepaper/whitepaper.md

### 2.2.2 Architecture

Cacti consists of two major components, the Business Logic Plugin (BLP), which handles the application logic, and the Ledger Plugin (LP), which is the component that connects to various blockchain infrastructures.

In furthermore, the LP is mainly composed of a "Verifier" and a "Validator," and the Verifier passes smart contract execution instructions, etc. from BLP to Validator (intermediary role), and the Validator executes smart contracts (transaction issuance, block monitoring, etc.) based on the execution instructions it receives from the Verifier.

The overall architecture is shown in the following figure.

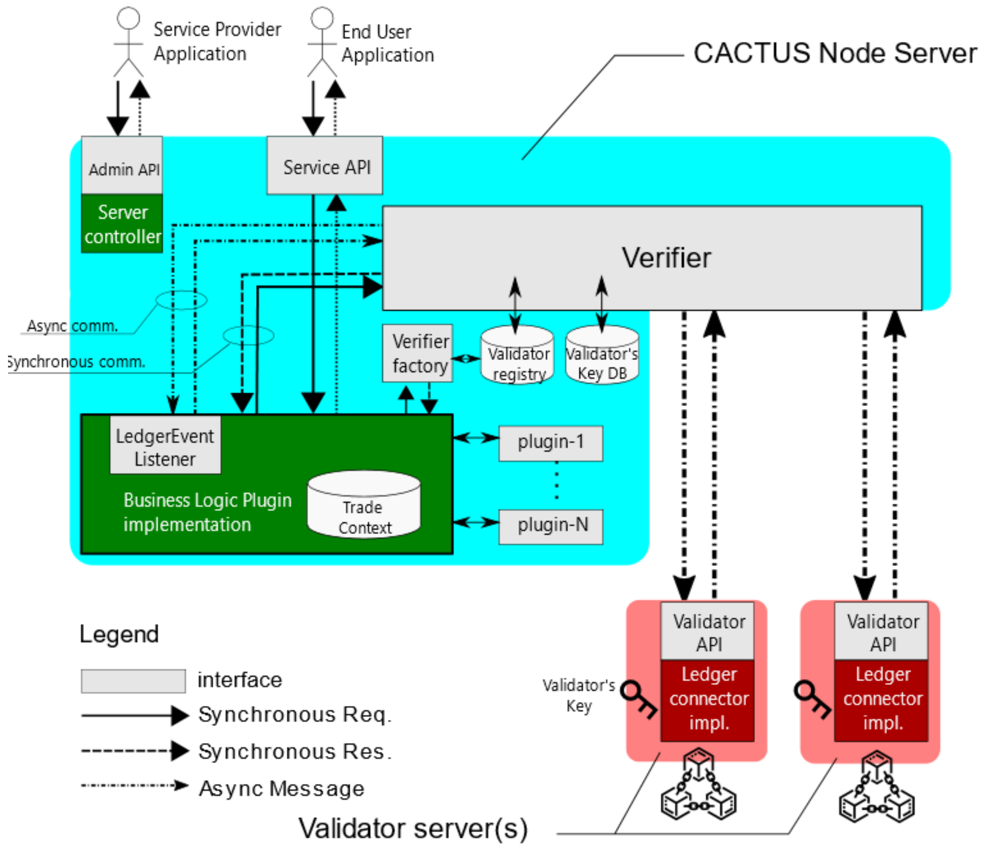


Figure 7: Overall Architecture

### 2.2.3 Definition of key components

- **Business Logic Plugin (BLP)**

The entity executes business logic and provide integration services that relate to multiple blockchains. The entity is composed by web application or smart contract on a blockchain. The entity is a single plugin and required for executing Cacti applications.

The CoreAPI of ConnectionChain used in this POC equivalent to this BLP.

- **Ledger Plugin (LP)**

The entity is the connecting components that connect BLPs to various blockchain infrastructures and are mainly composed of "Verifier" and "Validator".

- **Validator**

  The entity monitors transaction records of Ledger operation, and it determines the result (success, failed, timeout) from the transaction records. Validator ensure the determined result with attaching digital signature with "Validator key" which can be verified by "Verifier".

- **Validator Server**

  The server accepts a connection from Verifier, and it provides Validator API, which can be used for issuing signed transactions and monitoring Ledger behind it. The LedgerConnector will be implemented for interacting with the Ledger nodes.

- **Verifier**

  The entity accepts only successfully verified operation results by verifying the digital signature of the validator. Verifier will be instantiated by calling the VerifierFactory#create method with associated with the Validator to connect. Each Verifier may be temporarily enabled or disabled. Note that "Validator" is apart from "Verifier" over a bi-directional channel.

## 3. POC Result Details

In the POC, we verified the feasibility of cross-border DvP by connecting domestic DvP systems and remittance systems that simulate the counterparty countries of cross-border transactions (FCY System: Foreign Currency System), using Fujitsu's ConnectionChain and Hyperledger Cacti. The domestic DvP systems are developed by R3, ConsenSys, and Soramitsu respectively. The FCY System is developed by Fujitsu using Quorum. It has the ability to issue and transfer CashToken and "block" the required balance prior to transfer.

There are two cases of cross-border DvP, depending on the bond buyer's country. We have verified "Buyer Oversea" case where the buyer in the other side of domestic DvP country. We conducted actual connection testing with each vendor for the Buyer Oversea case. The detailed flow and test results with each vendor are attached as an appendix at the end of this report.

The intermediary role in conducting cross-border PvP was assumed to be played by the Central Banks on the Foreign Currency side (CBO: Central Bank Oversea), and Fujitsu emulated this role in the POC.

Actual connection testing confirmed that cross-border DvP transactions could be successfully executed with both vendors' systems. We also tested several failure cases and confirmed that they could be successfully handled.

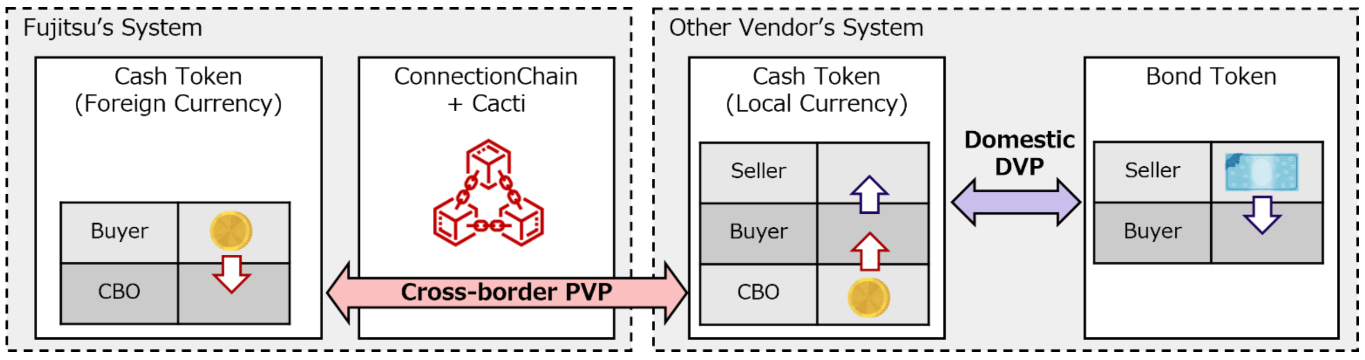


Figure 8: POC System Overview

The points that could be verified through the POC are listed below.

### 3.1 Immediately and Automated Cross-border Transaction

We verified that the "Inputfile" containing transaction information is entered into each vendor's system, and cross-border transactions are carried out immediately and automated.

In Fujitsu's system, Inputfile is entered into ConnectionChain via WebAPI, and the cross-border transaction process is automatically executed according to the scenario recorded in BC of ConnectionChain. The system also checks the format of the Inputfile and any input omissions at the time of input.

The series of operations, such as blocking the balance in one economy during cross-border PvP and transferring the money after confirming a successful transfer in the other economy, are also automated, eliminating the need for a human to check each bank's balance before transferring money, as in the past.

### 3.2 Decentralization of Systems

In cross-border transactions, it is a difficult problem who manages the cross-border system. Therefore, we have verified that the cross-border system can be decentralized using ConnectionChain, and that each participating country can decentralize the management of the system.

Specifically, each country will own the Hyperledger Fabric nodes and Core servers that make up the ConnectionChain, as well as the Cacti Ledger Plugin that serves as the gateway to each country's system and will share scenarios and history using BC. For the POC, Fujitsu built all of these components together, but each country can manage them individually during actual operation.

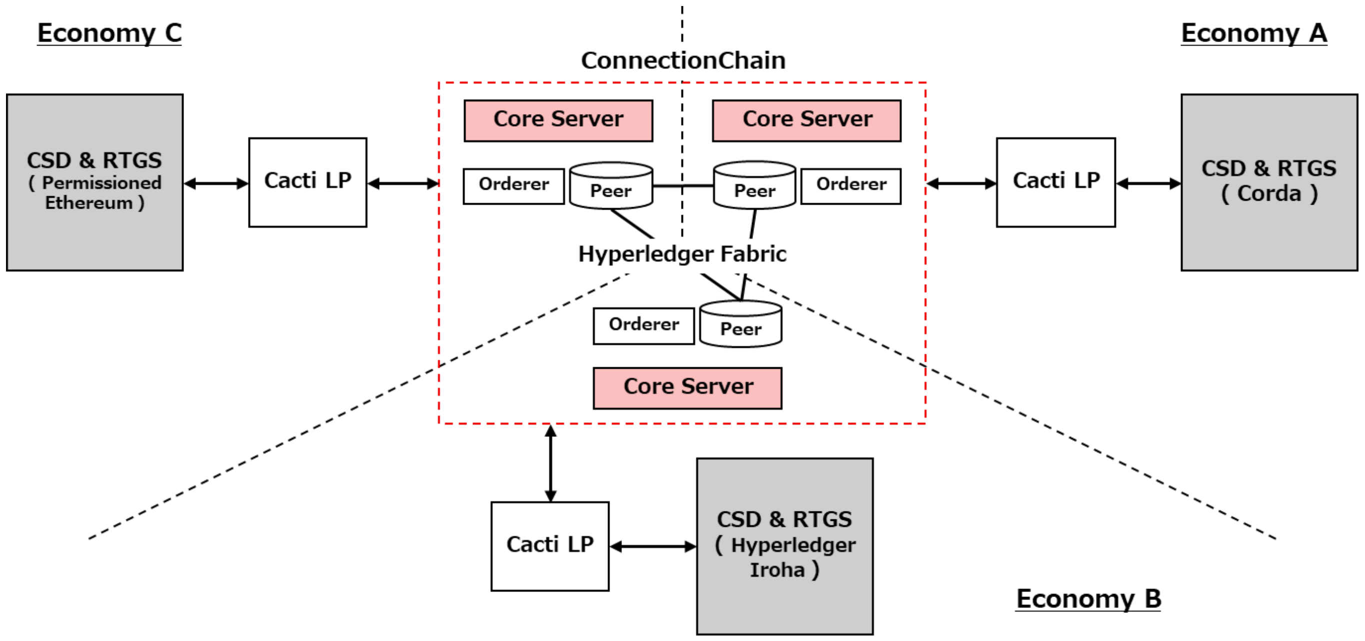


Figure 9: Decentralization of ConnectionChain to Each Economy

## 3.3 Error Handling

In cross-border transactions, it is essential to have a function to detect errors in each country's system and control the entire transaction. Therefore, we verified that the following two types of failure cases can be detected and handled appropriately.

### 3.3.1 Insufficient Funds to Execute Domestic DvP

This failure case assumed that the amount transferred by the overseas Buyer was less than the amount needed to execute the domestic DvP.

If the amount written in the Inputfile differs from the Buyer's deposit amount, the error can be detected by ConnectionChain at the time of cross-border PvP. However, if the amount written in the Inputfile is incorrect, the cross-border PvP will be executed and an error will occur during the domestic DvP.

Therefore, we verified that ConnectionChain can detect and handle errors that occur in each vendor's domestic DvP system.

Specifically, ConnectionChain received the status of domestic DvP by the event notification function of each vendor's system, and if an error is detected, the FCY side will unblock the buyer's funds. In this process, the CashToken transferred from the CBO on the LCY side will remain in the buyer's account, but we assume that refunds from the buyer will be made according to the governance of each country. This is because Fujitsu emulates the role of a central bank oversea and does not possess the authority to transfer local currency from Buyer accounts in other countries.

We consider that refunds are not required when funds are transferred directly from the CBO rather than from the Buyer when conducting domestic DvP.

### 3.3.2 Insufficient Data of Inputfile

This error case assumes that there are insufficient data or format violations in the Inputfile that describes the contents of the cross-border transaction. Therefore, it is possible to detect errors before the start of a cross-border transaction and stop the transaction.

## 4. Consideration

We will explain considerations about PoC system design and experiments.

### 4.1 Timing of Minting and Burning Tokens

We assume that central banks will use both the conventional checking account system and the CashToken system, which uses BCs for international remittances. Therefore, there are several possible patterns in the timing of conversion between checking account balances and token balances.

One pattern is "Onetime-mint," in which checking account balances are tokenized for each transaction and converted to checking account balances as soon as the transaction is completed. In this case, each participating bank does not have a token balance on the BC, and the checking account balance represents each bank's assets.

Another pattern is "Pre-mint," in which checking account balances are tokenized in advance and payments are made from the token balances held by the bank at the time of the transaction. In this case, each participating bank holds a token balance on the BC, and the current account balance and the token balance can be converted to each other at any time

When developing the system for the POC, it was unclear whether each vendor implementing the domestic DvP system would adopt Onetime-mint or Pre-mint, which made it difficult to start implementing the transaction flow.

When countries adopt a tokenized system, the timing of minting and burning tokens is expected to differ from country to country. Therefore, we consider that the connection system should be flexible enough to accommodate such a situation. Fujitsu's ConnectionChain offers flexibility through its multi-scenario functionality. We verified that it is possible to connect to both Onetime-minted R3 and Pre-minted ConsenSys and Soramitsu systems in the POC.

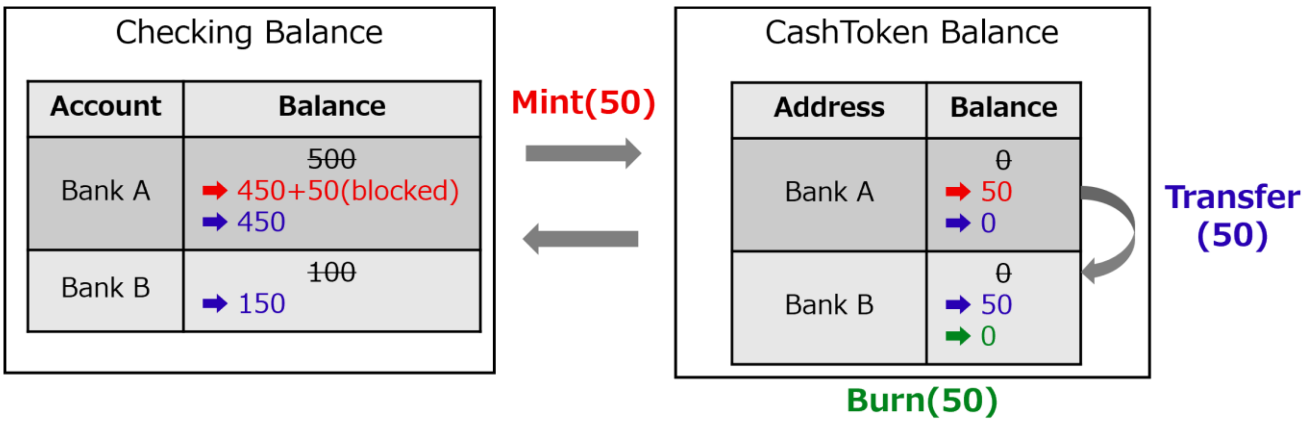


Figure 10: Transfer 50 Tokens in the Onetime-mint System

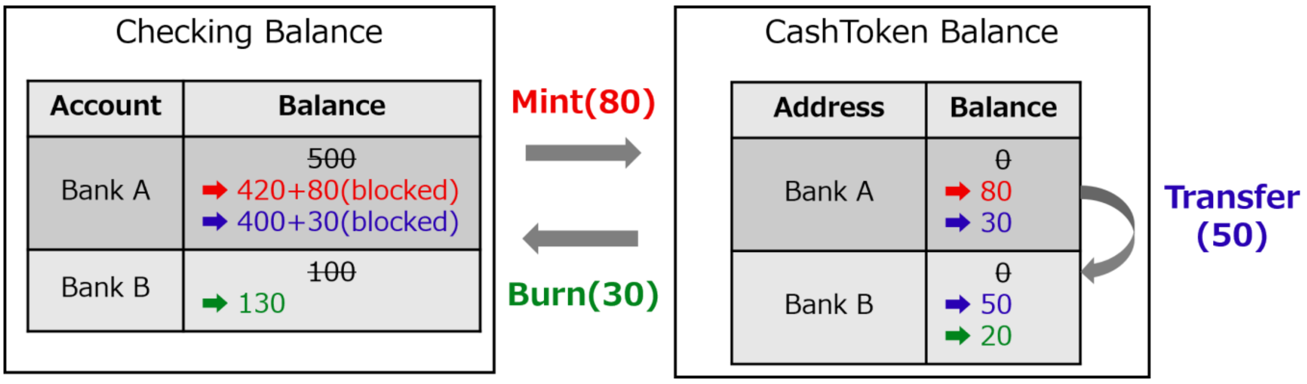


Figure 11: Transfer 50 Tokens in the Pre-mint System

## 4.2 Private Key Management

When accessing a system using BC, a secret key associated with the address is required to prove one's identity.

In the POC, private key signatures were required for CashToken transfers on the Soramitsu system. On the other hand, if the private key is managed by the system user, the risk of leakage increases, and in the event of a leakage, there may be a delay in responding to the leakage.

Therefore, Fujitsu used a method called a signature server. The private key is managed by the central bank in each country, and an access token in accordance with the OAuth 2.0 standard is issued each time a signature is required. A signed transaction can be received by sending the issued access token and the pre-signed transaction to a signature server.

The use of a signature server has the advantage that in the event of a private key leakage, it is possible to deal with the problem by simply changing the authentication information used to obtain the access token. In addition, limiting the transaction details (amount, destination, etc.) that can be signed for each access token deters unauthorized use of the token.

## 5. For the Future

### 5.1 Connectivity of Various Financial Platforms

We designed the DvP system to settle bond sales by combining two systems. They are one for a local DvP system that settles Bond Token with Cash Token in local currency and a cross-border PvP system that enables money transfer with currency exchange to supply Cash Token to be settled by the local DvP system. This hybrid design benefits to ease extending the scope of interworking.

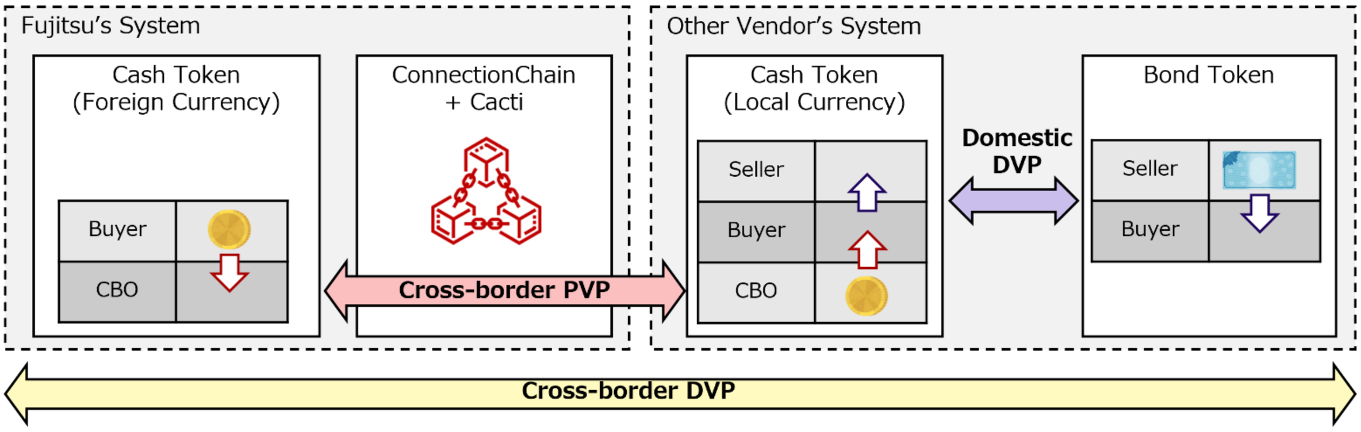


Figure 12: Cross-border DvP consisting of Cross-border PvP and Domestic DvP

However, it is also possible to connect the systems developed by each vendor to each other. Also, it is possible to realize 1:n connections instead of 1:1. This would allow for more complex transactions through multiple countries.

An example would be cross-border PvP via a third country's currency. For example, when transferring money from a Japanese bank to a Cambodian bank, even if the Japanese and Cambodian banks do not have accounts with each other, if both banks have an account with the Chinese bank, the money can be transferred via that account.

Fujitsu's ConnectionChain can be used for such complex transactions through multiple BCs.

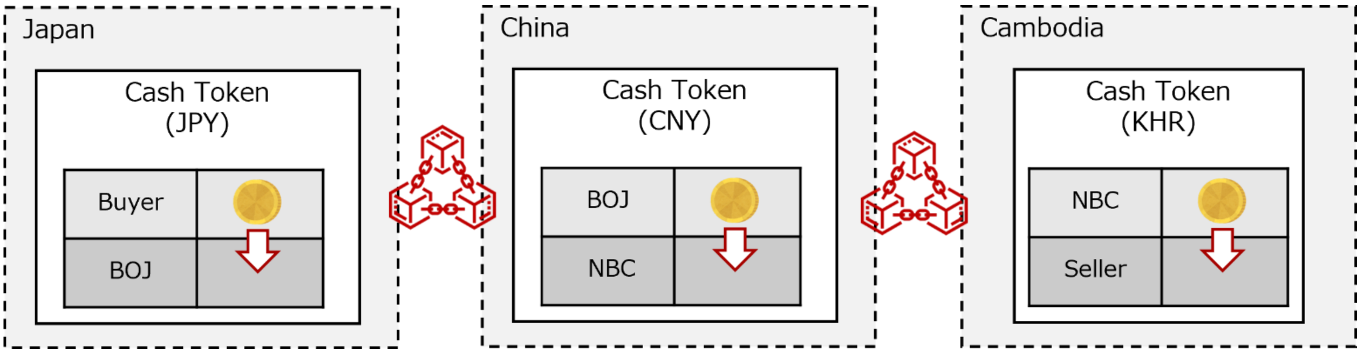


Figure 13: Cross-border PvP via Third Country

### 5.2 Dealing with Transitional Periods in the System

Even if countries adopt BC-based systems in the future, not all countries participating in cross-border transactions will revamp their systems at once.

There may well be cases where one side has a tokenized system, and the other side only has a legacy checking account system. In such a transitional period, the system connecting each country must be flexible enough to accommodate both legacy and BC systems. Fujitsu's system uses Hyperledger Cacti to connect to each country, which can support various BCs, and at the same time, it can also connect to legacy systems by using Ledger Plugin for WebAPI.

ConnectionChain is connected to R3 and ConsenSys through the WebAPI layer and to Soramitsu through the BC layer in the POC. We believe that ConnectionChain has the flexibility to handle such transitional periods.

## 5.3 Timeout of Cross-border Transaction

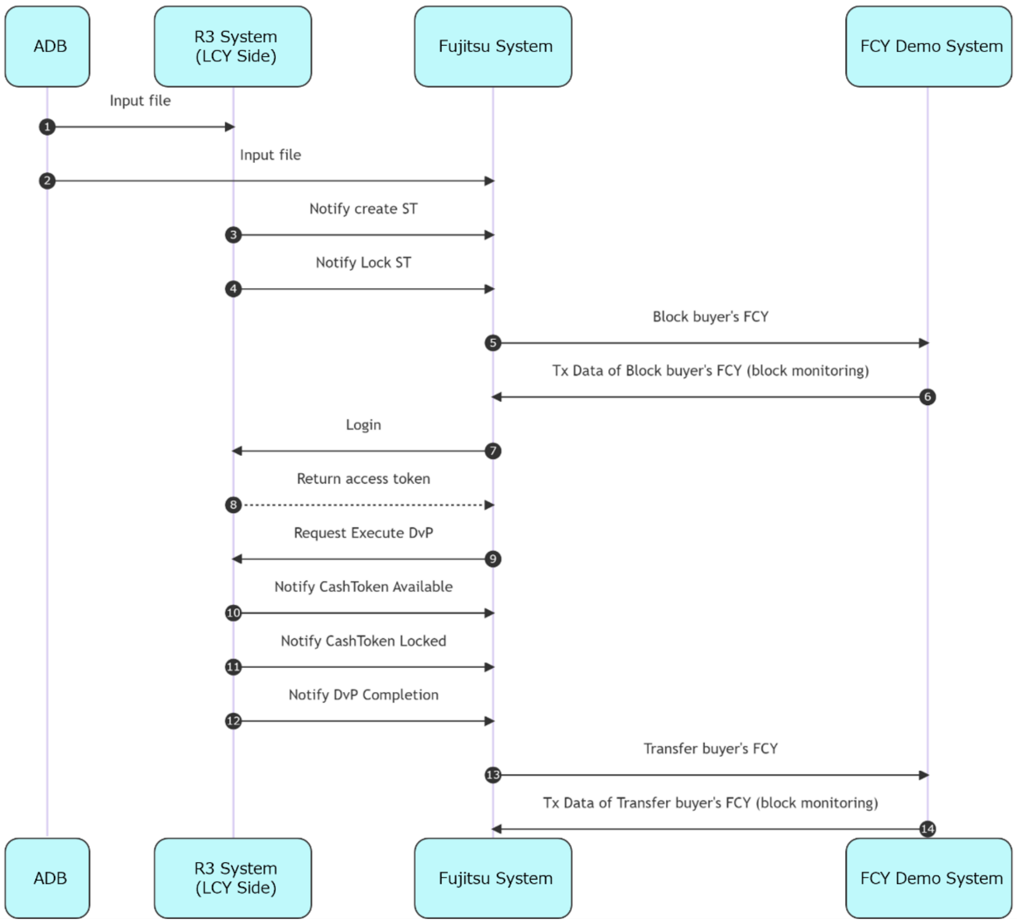
In cross-border transactions, the possibility exists that a timeout may occur due to one of the participants making the transfer but not the other by the deadline, or due to a system outage.

In such cases, it is necessary to properly cancel the transaction when the transaction due date is written in the Inputfile.

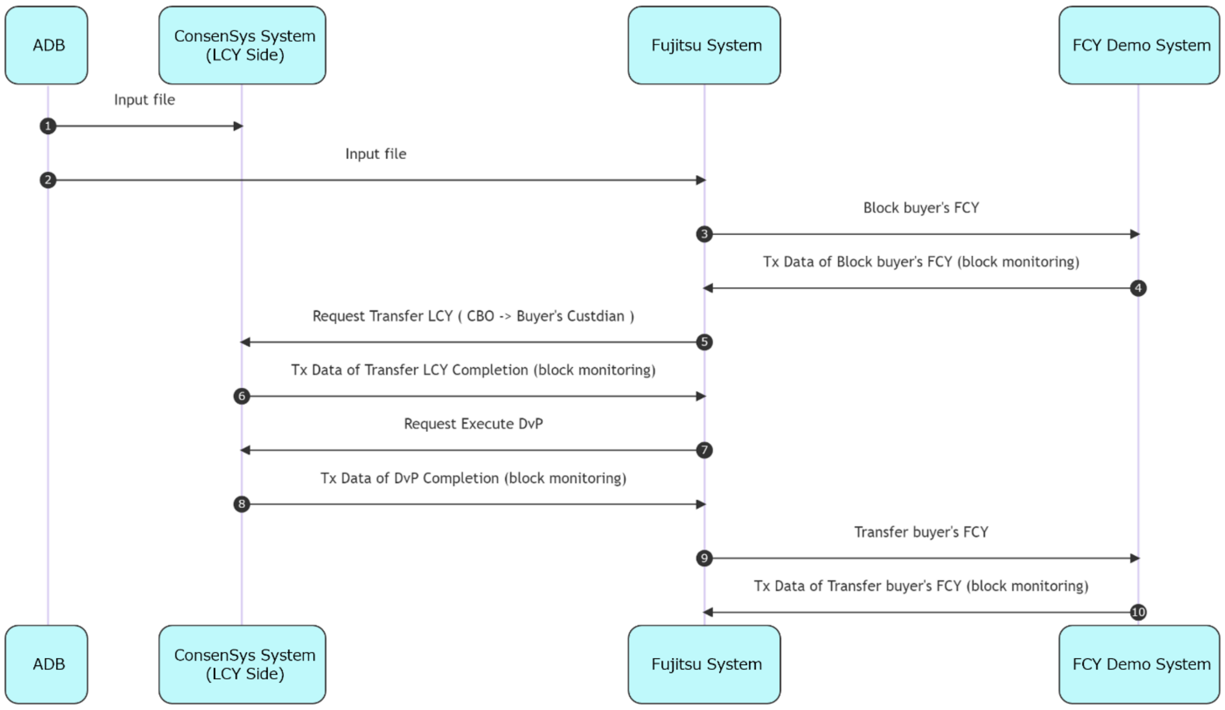
Although not implemented in the POC, by adding a component for timeout management in ConnectionChain, it is possible to cancel the scenario execution if the transaction is not executed by the deadline.

# 6. Appendix1: Transaction Flows of Each Vendor

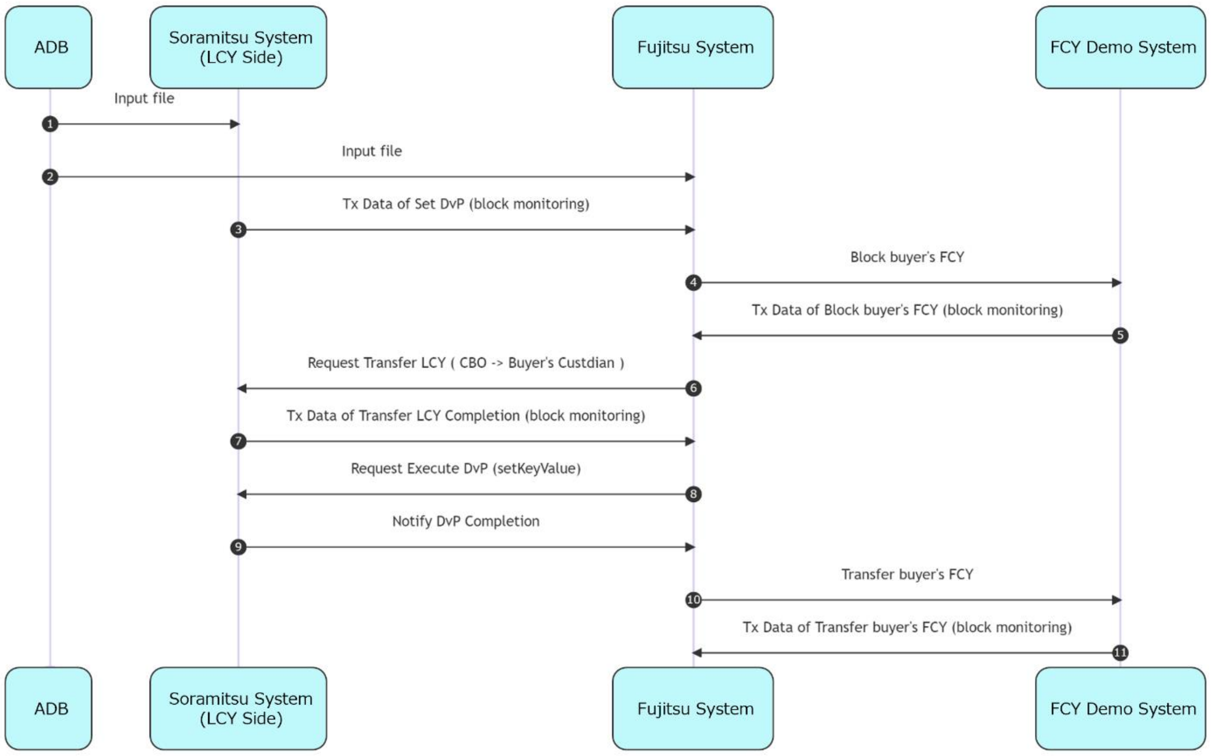
## 6.1 R3



## 6.2 ConsenSys


ADB
ConsenSys System (LCY Side)
Fujitsu System
FCY Demo System
1 Input file
2 Input file
3 Block buyer's FCY
4 Tx Data of Block buyer's FCY (block monitoring)
5 Request Transfer LCY ( CBO -> Buyer's Custdian )
6 Tx Data of Transfer LCY Completion (block monitoring)
7 Request Execute DvP
8 Tx Data of DvP Completion (block monitoring)
9 Transfer buyer's FCY
10 Tx Data of Transfer buyer's FCY (block monitoring)
ADB
ConsenSys System (LCY Side)
Fujitsu System
FCY Demo System

## 6.3 Soramitsu


ADB
Soramitsu System (LCY Side)
Fujitsu System
FCY Demo System
1 Input file
2 Input file
3 Tx Data of Set DvP (block monitoring)
4 Block buyer's FCY
5 Tx Data of Block buyer's FCY (block monitoring)
6 Request Transfer LCY ( CBO -> Buyer's Custdian )
7 Tx Data of Transfer LCY Completion (block monitoring)
8 Request Execute DvP (setKeyValue)
9 Notify DvP Completion
10 Transfer buyer's FCY
11 Tx Data of Transfer buyer's FCY (block monitoring)
ADB
Soramitsu System (LCY Side)
Fujitsu System
FCY Demo System