Horizon Payment Switch

Distributed Open Banking Theory

Document Version: 1.0

Release Date: March 7, 2021

Document Change History				
No.	**Doc Version**	**Release Date**	**Author**	**Change Description**
1	1.0	2021-03-07	Mahyar Esteki	Create First Version

Contents

1 Abstract

Performance and security are always the distance of a coin in the banking and fintech systems, which have exactly the opposite position to each other. Balancing these two parameters is one of the main concerns of the world's banking systems. The issue becomes more complicated when needs such as providing extensive services to all merchants are also prioritized by a bank or banking institution. On the other hand, compliance with current standards in financial transactions is one of the principles of today's banking systems. To balance this set of conflicting demands, there is a comprehensive solution called **“distributed open banking theory”** that ensures improved system performance while maintaining security standards and international banking business models.

2 Problem Definition

Over the past two decades, banking and fintech operations have shifted to centralism. Banks and financial institutions have sought to collect and manage data centrally by developing core banking systems. Fintech systems have also developed central switches and ESBs to centrally monitor and manage data related to banking transactions. This has led to increased security and efficiency risks of banking systems. Such architectures mainly waste server resources on fintech systems. Network administrators will not be able to allocate and manage the hardware resources available in the bank or financial institution based on the growth rate of transactions in specific categories.

For example, Citibank intends to manage its credit card transactions in four states: Texas, New York, Arizona and Montana. According to the latest official statistics released on July 1, 2020, the population of these four states is as follows. [1]

- **Texas:** 29360759
- **New York:** 19336776
- **Arizona:** 7421401
- **Montana:** 1080755

Managing the transactions of these four states in a central system is a big mistake in the field of banking management decisions. A central management system imposes the overhead of operations on two populous states (Texas and New York) to two less populous states (Arizona and Montana). Also, if the system becomes unavailable due to large financial transactions in New York, the other three states will be barred from receiving banking services.

It should be noted that the nature of financial activities in these four states is quite different. Due to Wall Street activities, New York's financial transactions, both in number and in value, are even higher than in a state like Texas with a population of more than 29 million.

Perhaps the best example that can prove that the era of centralized storage of financial data is over and that central systems are no longer as secure as before is **"The Distributed Ledger"**. The distributed ledger is a socially distributed duplicate, shared, and synchronized digital data spread geographically across countries, countries, or institutions.[2] There is no central manager or centralized data storage. Distributed general ledgers are databases that are maintained and updated based on the understanding mechanism and data architecture accepted by the network participants.[3]

Researchers and experts have used this concept to develop **"the blockchain"**. The blockchain has been described as "an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way".[4] Although the purpose of this study is not to develop peer-to-peer networks and cryptocurrencies in banking systems, the concept clearly shows that information centralism is no longer emphasized even in sensitive financial operations. It may not be possible to implement complex structures such as those in a blockchain for banking system. But distribution of operations and storage can be used to develop a more secure infrastructure for managing banking transactions.

The distribution of operations in banking systems is not only important for performance improvement, but also from a functional point of view. It is because of this basic need that many banks have resorted to using the open banking model. Open banking is a financial services term as part of financial technology that refers to: [5]

1. The use of open APIs that enable third-party developers to build applications and services around the financial institution. [6]
2. Greater financial transparency options for account holders ranging from open data to private data.
3. The use of open source technology to achieve the above. [7]

3 Background

In 1976, John Fralick Rockart and Joav Steve Leventer [8] reported on the serious shortcomings of centralized systems. They have found that organizations' lack of acceptance of distributed systems is a lack of understanding of the meaning of the term. They stated in their report:

> **“Unfortunately, the term "distributed processing" means different things to different people. The term is mentioned, however, mostly in relation to hardware configuration. For the purpose of this paper, a "distributed system" is a system of interconnected computers (CPU's) with their own mass storage, each at a different organizational location. Such a system may offer some of the advantages of both centralization and decentralization, and thus is an important alternative for consideration.” [8]**

They also list the results of using distributed databases as follows:

- Better control and direct use of data by those who are most concerned with it.
- Substantially lower communication costs
- Lower overall installation costs
- Faster implementation of a company wide information system".

Shirley Lu, in a study by MIT CSAIL Computer Systems Security Group, emphasizes grouping banking transactions into smaller, more categorized databases.[9] She and her colleagues explain that in order to develop a consistent distributed banking system (CDBS), it is necessary to group transactions and divide them at the data layer level. This will cause the division of transactions in the operational sector based on the same groupings.

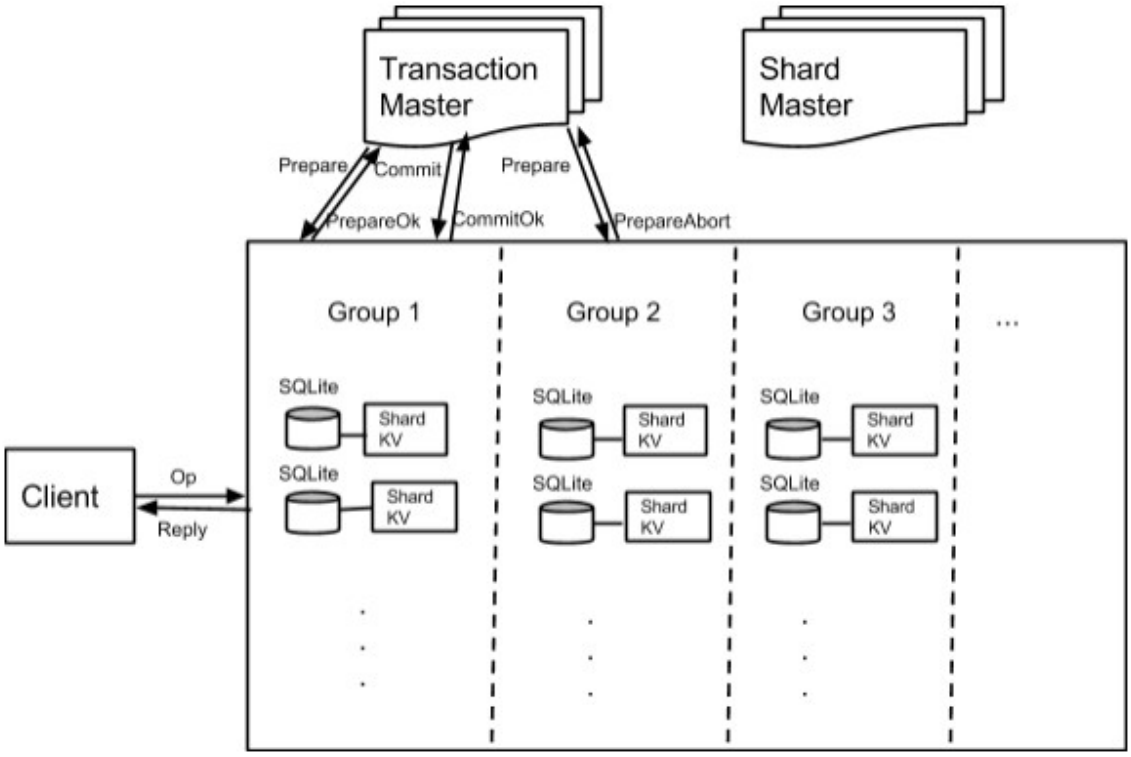

Figure 1: consistent distributed banking system (CDBS) [9]

In a related study, Isabel Simplot-Rail [10] noted that in interbank transactions (such as what happens in MasterCard, Visa Card or PayPal), in addition to the payment channels in the bank in question, governance money transfer channel will also be involved. Transactions can be sent from anywhere to any destination. Thus, banks are faced with a wide range of financial operations that are requested from a wide range of geographical areas.

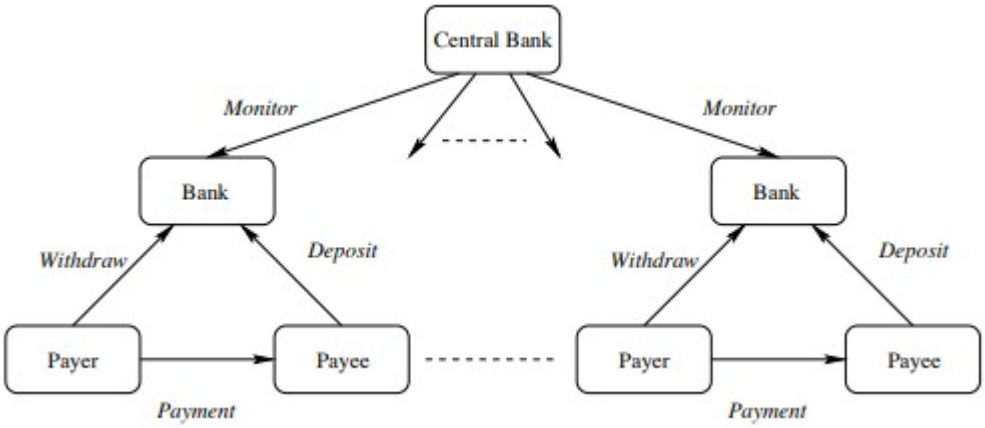

Figure 2: Distributed Electronic Banking Model [10]

Patel and Garg, in a joint study on developing a platform for mobile agent distribution and execution (PMADE), found that in an interface channel for mobile banking software, it would be better to distribute operations across multiple servers. [11]

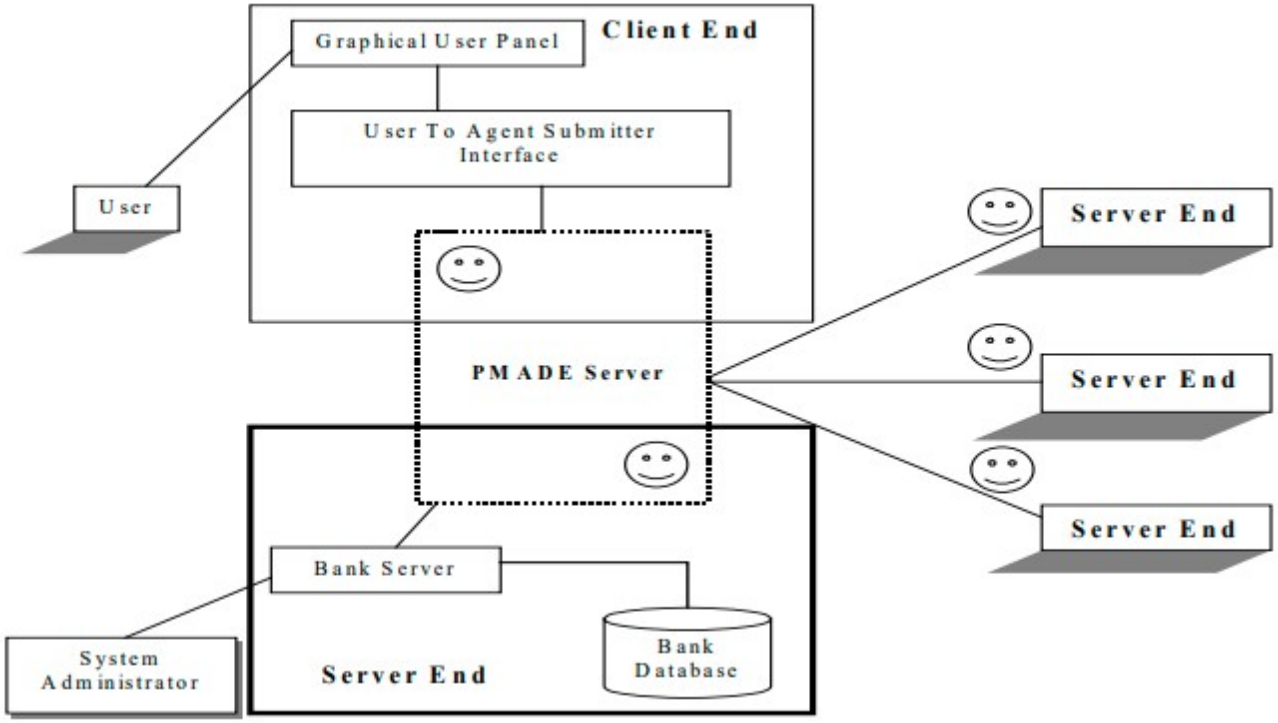

Figure 3: PMADE in architecture of the banking system [11]

In a joint article, Dr. Sebastian Heuser and Dr. Simon Alioth[12] explain that in open banking systems, operations should be distributed between different servers based on the type of customers, service areas, or branch groupings.

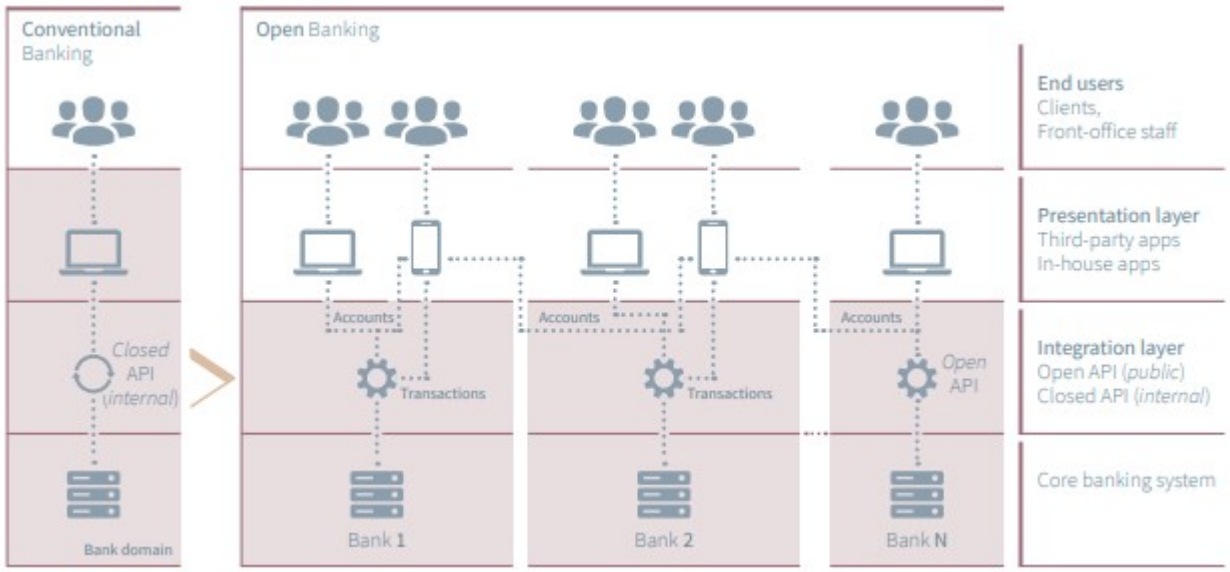

Figure 4: An open banking system [12]

4 Disadvantages of Central Banking Systems

- High risk when the system becomes unavailable.
- High risk of interruption of service in all states or provinces covered at the time of interruption.
- Dependencies of the states or provinces covered by each other in terms of performance.
- High sensitivity in maintenance and monitoring.
- High risk transaction management when faced with a global market.
- High global downtime probability.
- Impossibility of distributing hardware resources.

5 Distributed Open Banking (DOB)

According to the definitions in the IEEE-829:2008 standard, banking systems are classified at level 3 (critical systems). This means that banking systems are highly sensitive in terms of security and performance. These types of systems, while having optimal performance, must have acceptable security. Providing these two requirements for a bank using a centralized information system is like trying to solve a traveling salesman problem on several different maps using only a specific path!

It is quite impossible to predict that a user will be connected to a banking network from which region of a country or even the world and through which terminal. It is also not possible to determine to what extent a hacker is able to penetrate a banking system. Therefore, connecting all terminals to a

central system and storing all bank data on a centralized database is a completely obsolete and dangerous method. All operations and data must be divided according to different modules in a banking system and installed and stored on separate servers. The importance of meeting this requirement will increase when faced with fintech systems.

Fintech systems deal with a wide range of operations that must be provided to the bank's customers on a large set of terminals. In addition, it is even possible that one fintech system will have to interact with several banking systems simultaneously. Therefore, it is possible that an open banking system even deals with a complex set of different protocols when receiving requests and sending responses. For example, a payment gateway may communicate with clients using RESTful APIs. But it will have to send its message to the governance payment switch in accordance with ISO-8583. This operational complexity increases the sensitivity to system performance. However, it is not possible to violate the security red line in banking systems. Therefore, overloading the security controls of all transactions on a centralized system cannot be the cornerstone of system performance concerns.

To resolve this conflict, operational responsibilities and data warehousing should be divided between different servers according to the grouping of transactions in the open banking layer and the classification of modules in the core banking layer. In the field of fintech, the criterion of action is transaction. Transactions can be classified depending on various parameters. City, state or province, type of terminal, version used by the client, and several other items can each be used as criteria for classifying transactions. But the choice of criteria for grouping operations in the core banking systems is quite different. Because in such systems the principle is the system modules. In other words, operations are grouped based on the type of customer activity in the system. Whether the customer wants to get a loan or open a deposit determines in which system his activity should be managed and where his data should be stored. Choosing the best criterion will depend entirely on the management conditions and hardware facilities of the bank.

However, there are ways to ensure that the best practice is selected to implement an appropriate grouping to distribute operations in the banking system. This operational distribution must be implemented in such a way that end users of the system in the open banking layer do not feel drastic changes and integrity is maintained in the data layer without imposing data redundancy on the system. For example, if an operational distribution is to be implemented in the core banking system, the BIAN[1] service landscape can be used.[13] This standard helps system developers and architects to have a good understanding of the modules and operational divisions in a core banking system. Figure 5 clearly shows that the domains and modules of a core banking system are based on the activities and natures that a customer deals with in a bank. As a result, operational and information distribution on the servers of a core banking system must be implemented in the same way.

1 The Banking Industry Architecture Network is created to establish, promote and provide a common framework for banking interoperability issues and to become and to be recognized as a world-class reference point for interoperability in the banking industry. [14]

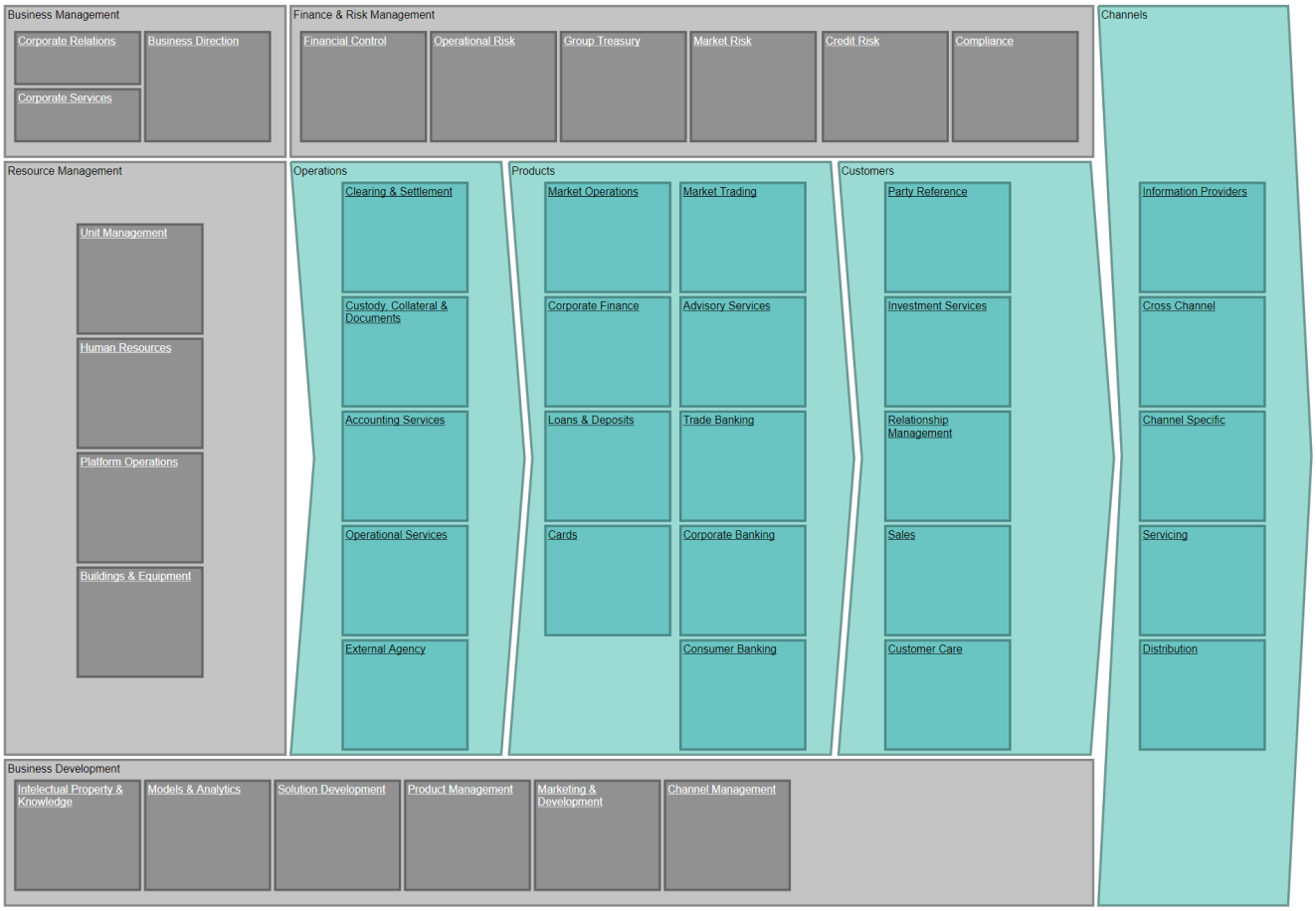

Figure 5: BIAN Service Landscape 9.0 [13]

One of the most common ways to implement the distributed pattern in fintech systems is to set up enterprise service bus (ESB) as an interface channel between the client and the core banking system services. In a joint study, Fernandez[15] states that the operational division in an ESB system will be based on the list of services of the main system. However, if the back-end of an ESB in the bank is powered by a bank switch that simply receives the messages and converts them to ISO-8583 format, service-based operations distribution is not used in this case.

However, it should be noted that it is not always possible to implement a transactional distribution based on operational divisions in a fintech system. When payment gateways in a densely populated country with a complex structure or even globally (such as Swift or MasterCard) provide service to customers of an extensive banking network, if transactions are distributed on the basis of segmentation Operationally, the risk of disconnection across one or more services will increase for all customers. In such systems, the population receiving services and their geographical distribution are of particular importance. Because they will determine the amount of network traffic (transaction rate per second) of a banking system.

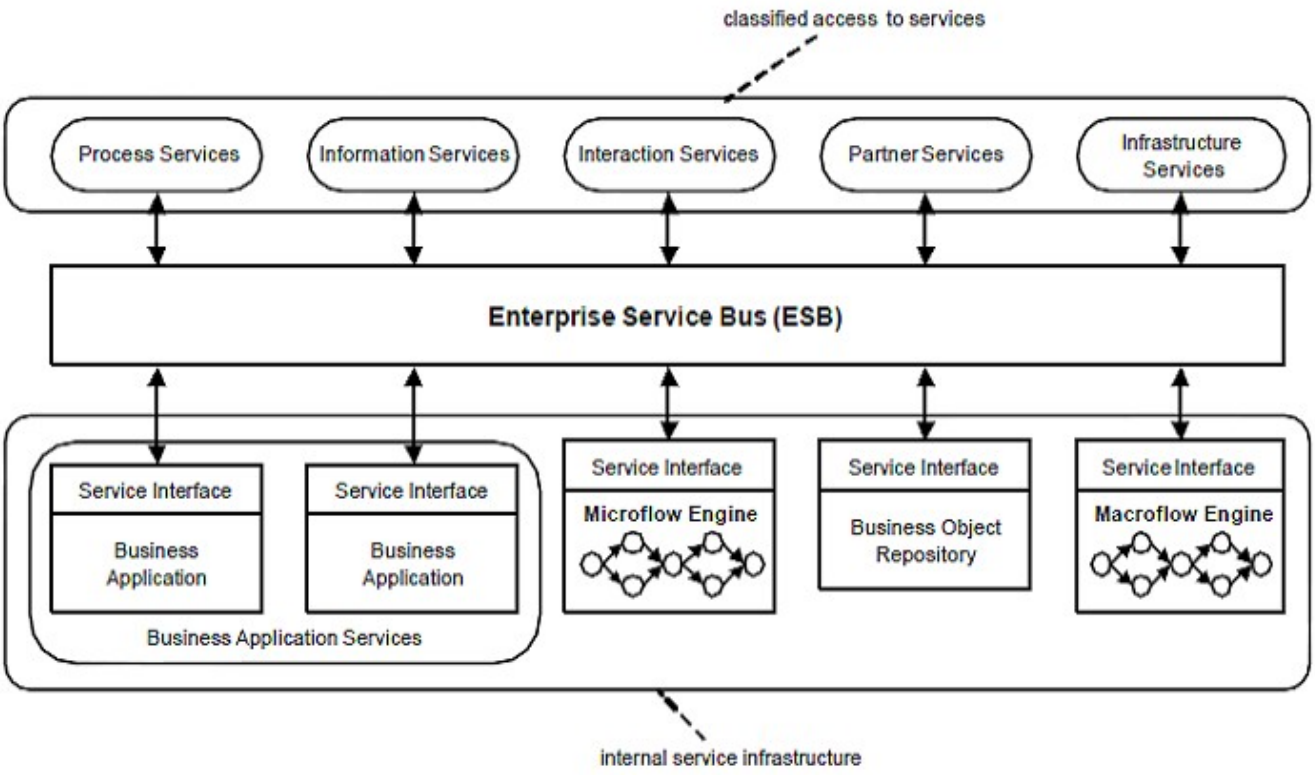

Figure 6: an enterprise service bus (ESB) with operational distribution [15]

There will certainly be no flawless distribution in information systems. But in banking systems, it is necessary to choose the distribution that has the highest level of security and the highest possible performance. It may seem difficult to strike a balance between two opposing properties, such as security and performance. What paves the way for achieving this balance is an examination of all the parameters that affect system performance and business risks. An effective architecture must be able to answer the following questions.

- Is there a possibility of a total system shutdown? If so, what will be the impact of the answer on all clients?
- Are clients functionally interdependent? What are these dependencies? Can they be reduced?
- What is the definite risk of the system?
- How to group requests sent to the system?
- Does the system serve a wide geographical area? How far is it?
- Are services provided to clients atomically? How dependent are the services?
- Who is the recipient of the service?

- Is slowing down or interrupting the system due to operational overhead in one area justifiable for clients in another?

These questions are a little difficult to answer for a banking system that serves customers around the world. In such systems, interruptions or delays in service delivery are not easily justified. The importance of customers cannot be measured depending on where they live or how much they use the system. Therefore, architecture should be considered for such systems so that while focusing on high-traffic areas, proper service to all customers of the business is a priority of that architecture.

Before transactions in a Fintech system require an operational distribution, they need to be geographically distributed. It is undeniable that the population and the rate of industrial and economic growth of a province or state affect the transaction rate sent from that geographical area. Therefore, the population and volume of economic activities in a geographical area will affect the overhead of a banking network. Admittedly, the best way to allocate hardware resources to any information system is to measure the potential amount of processing required to it. The number of requests sent to a system from different locations is not always constant. The number of requests sent from different points to a system may fluctuate. But they all follow very clear patterns, the most important of which is the level of activity of applicants in that area. If the goal is to allocate hardware resources fairly and based on actual usage, one possible solution is to distribute the system based on geographical segmentation.

It can be said that the best possible distribution for a Fintech system is a combination of two geographical and operational divisions. In such a way that for each province or state in a country, a separate copy of the desired payment gateway or switch is executed on an independent server and the desired operations are implemented as a microservice in that version. All clients communicate with the system through a common gateway, but the processing of requested operations based on their IP address will be left to the server of the province or state in which they reside.

6 Advantages of DOB

- Minimal dependence between states or provinces in terms of performance and efficiency.
- Ability to better serve global markets.
- Ability to distribute hardware resources based on the consumption of each region.
- Low stress in maintenance and monitoring.
- Low global downtime probability.

7 Strategy Approach

Since the early 1980s, most of the world's banks have gradually shifted to the **“customer-centric model”**. This model was a very good solution for managing customer activities and made it possible for customers to perform various banking operations in other branches, regardless of where the

account opening branch is located. This approach greatly contributed to the development and expansion of modern banking solutions. Because by opening an account in any branch, the customer is able to use e-banking services in any place. In this model, the customer number and the type of activity are the criteria for managing banking processes. Therefore, core banking systems should manage and distribute their processes according to the **“who and for what”** logic.

But the range of customers of a bank may be so wide that it is very difficult to manage them with this logic in various modern banking channels. Channel management systems and banking switches may even process a variety of requests based on bank cards issued by other banks. It is impossible to separate transactions based on customers and card numbers. Even distributing transactions based on bank card prefixes, which represent the card issuing bank, will be very difficult. Because basically there are many changes in the level of banks and card prefixes. To implement such a strategy, the top managers of a bank must accept the constant changes on their internal network servers and clusters, which will impose a huge cost on that bank. Therefore, the priority of determining “who is requesting services” should be reduced. But the divisions of a country approved by the parliament of each country and the global demarcation approved by the United Nations are more stable and have a clear and logical limit. In addition, the range of IP addresses in each region is well defined and the distribution of requests and processes can be managed accordingly. Therefore, the strategy of distributors in open banking systems should be based on **“what request is sent from where”**.

8 What is Zone and Zoning?

The success rate of a system's distribution of processes depends on its operational division. It is very important on what basis the criteria for the distribution of operations are selected. Are these criteria selected in accordance with the characteristics of the relevant business? In redistributed open banking theory, this distribution will be based on geographical zoning. The act of dividing the system into different areas operationally and geographically is called “zoning” an each of the created sections is called “zone”. Zoning can have different layers and each system can have more than one zoning layer according to its complexity and size. The number of users and the size of the business affect the number of layers of zoning and its complexity.

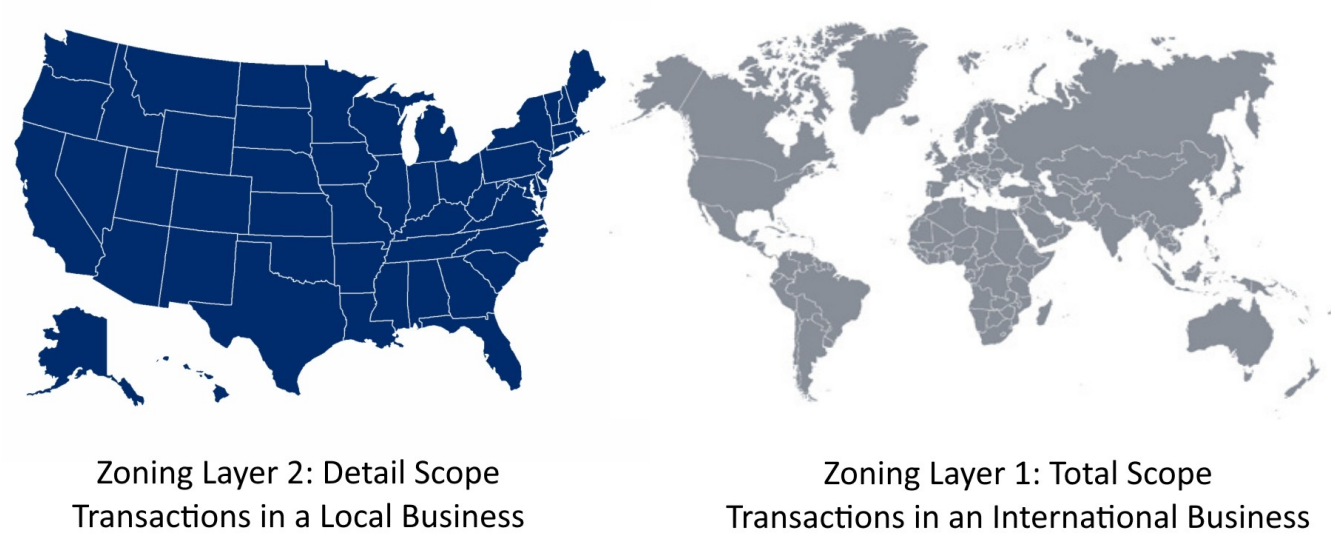

Figure 7: a complex zoning layers

It must be acknowledged that the zoning of a system will be very much influenced by the strategy and business model. But even the managers of a business can not increase the layers of zoning indiscriminately and demand unusual details from the administrators of the system. Because increasing each level to system zoning will impose many costs on the organization. In many cases, over-definition of zoning layers has imposed huge costs on organizations in the areas of hardware, networking, and configuration. This happens when senior executives have a full-fledged business to receive more detailed information from clients in addition to improving system performance. Therefore, the first step to implementing a proper zoning in the system is to maintain balance in its design. For this purpose, it is better to have a think tank with the participation of system architects and senior business managers to provide a reliable zoning using causal methods, such as Ishikawa diagram.

9 Load Balancing

Load balancing is a general concept for creating a proper distribution of requests in enterprise systems. “Load balancing” is defined as the methodical and efficient distribution of network or application traffic across multiple servers in a server farm. Each load balancer sits between client devices and backend servers, receiving and then distributing incoming requests to any available server capable of fulfilling them.[16]

To improve the performance of a distributed open banking system, there is a serious need for load balancig to manage requests. One of the common tools for load balancing in the system is “reverse proxy server”. A reverse proxy server is a type of proxy server that typically sits behind the firewall in a private network and directs client requests to the appropriate backend server. A reverse proxy provides

an additional level of abstraction and control to ensure the smooth flow of network traffic between clients and servers.[17]

The distribution of requests in an open banking system should be managed based on the zoning implemented in the system. If zoning is based on the type of operation, a message should be sent to the associated server based on the requested operation code. However, if the zoning is based on geographic areas, the request must be sent to the server associated with that geographic area based on the IP address of the requesting client. All these measures are taken for proper distribution of processing. Therefore, business managers accept a logical complexity to improve system performance. Accepting such complexities will not be without consequences. To reduce the negative effects of such complexities, a reliable zoning in the system must be ensured. In fact, load balancing is the practical step of implementing zoning in request management of a system.

Without balancing the system load, a proper processing distribution can never be achieved. And of course, without distributed processes, a distributed architecture can never be achieved. As a result, requests must be routed to more than one server to get rid of centralized processing systems. In such cases, it is better to consider a preliminary zoning for the system. Each area in this subdivision can consist of several areas in the final design. Therefore, until all network and hardware requirements are met, requests for multiple areas can be managed on one server or docker. Certainly there is a possibility of changing zoning based on organizational conditions, geopolitical changes, policies of partners with governments, and ext. In such cases, it should be possible to change and reorganize the servers according to the new zoning. Therefore, it is expected that the load balancing mechanism in the system has the necessary flexibility to apply such changes.

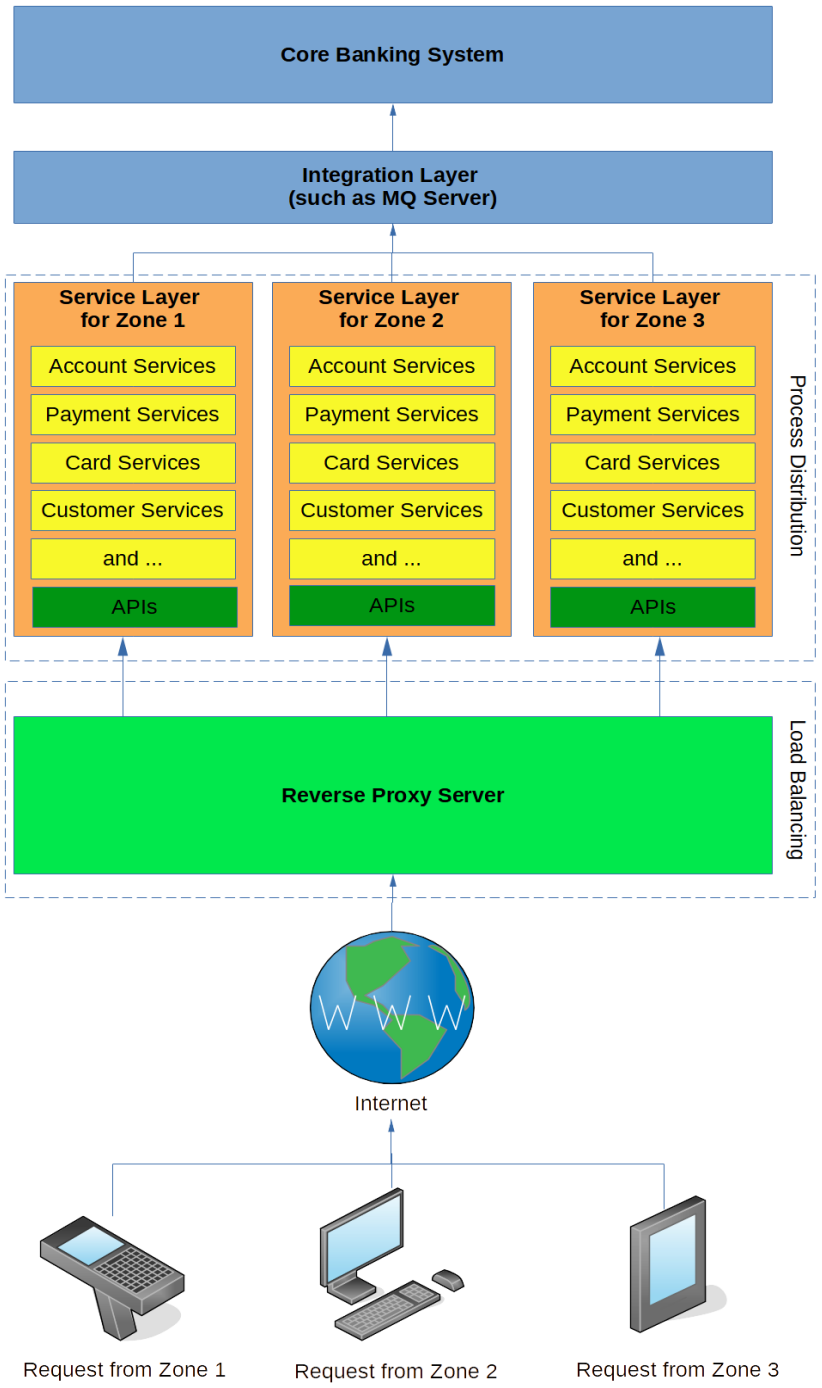

Figure 8: Load balancing in DOB

10 Data Partitioning

The information available in open banking systems consists of the following two types of data.

- **Basic information:** which is mainly extracted from the information available in the core banking systems and is used only for reading and comparing. This type of data will not change much in most cases.
- **Transaction information:** which includes data received from applicants and responses sent to them. The value of this type of data is constantly changing and the number of records is increasing every moment. But in many cases the servers involved in the processing only have access to add to this part of the data and they are not allowed to read on this type of data.

It is best to keep the basic information on an in-memory database that is synchronized with the main database and eventually with the data changes in the core banking system database. This will significantly reduce data retrieval from the main database when accepting large numbers of requests, which will ultimately improve system performance. It should be noted that this part of our data is shared between all zones and therefore it is better to keep it in a part of the database cluster that is shared with servers in all zones.

On the other hand, the number of requests and the number of records recorded for transactions varies depending on each Zone. There are many examples [18] that clearly show that it is possible to store a common data type in separate tables depending on a specific factor, which is called “table partitioning” in the database. Using this method, the data of a system can be stored in a distributed distribution. For example, transactions received from ATMs in three states can be received and processed on three different servers. On the other hand, the information of these transactions is stored on three separate partitions of the transaction table. This technique makes it possible, in addition to storing data separately, to be able to fetch it collectively.

As a result, users will be faced with a system that collects data separately. But it is also possible to present it together. This forms the basis of the distribution of operations in enterprise systems. Multiple servers may perform a single task in parallel on transactions received from different regions. In such cases, these servers must be able to each store their associated transaction data individually without having to worry about running other servers. It should be noted that the maximum distribution of operations in banking systems occurs when the system is trying to store information about transactions or operations performed in the database.

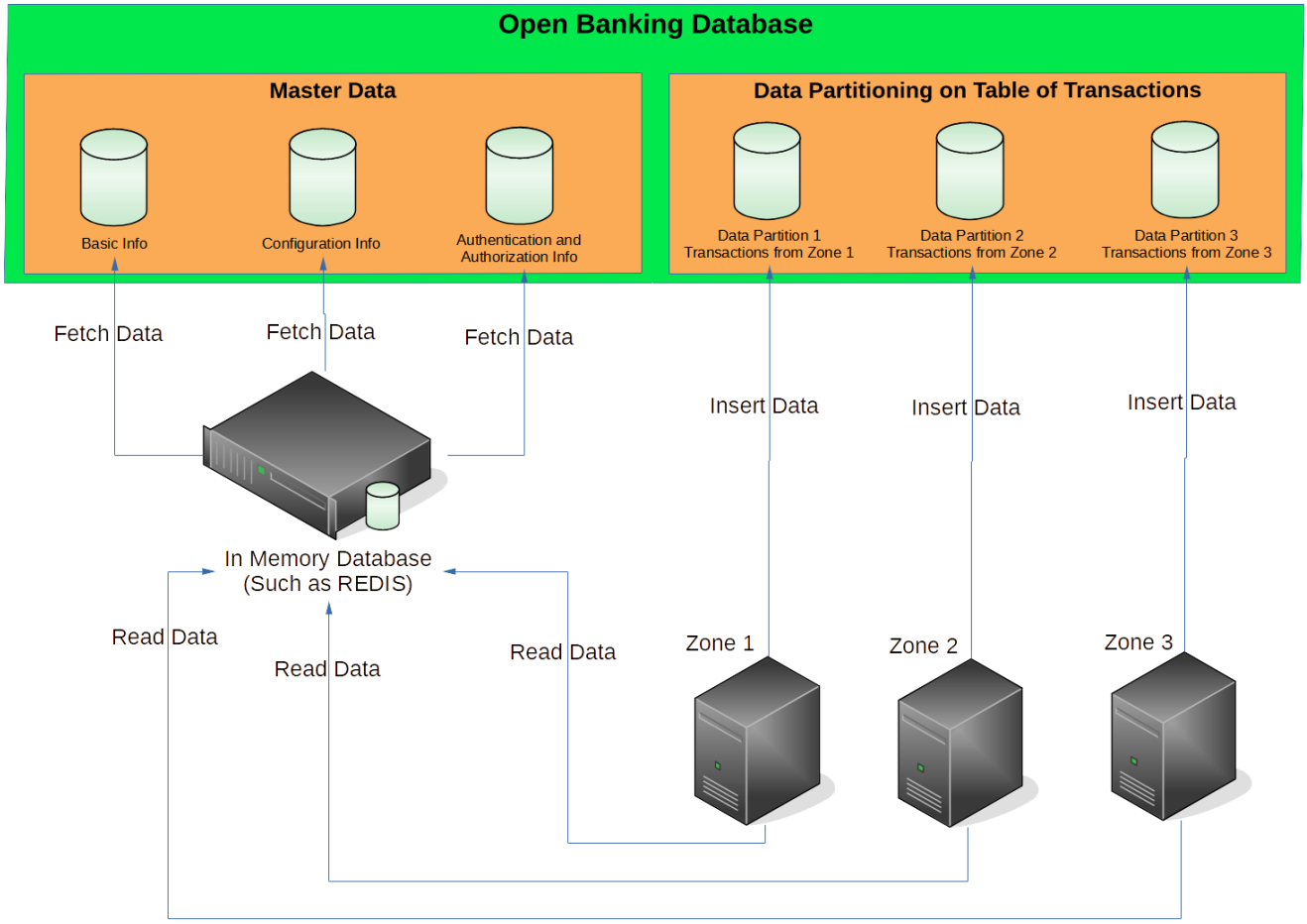

Figure 9: Data Partitioning in DOB

11 Change Detection

The worst kind of changes in the open banking system are the changes in the back-end of the system. Sometimes there may be changes in the input and output levels, even in the structure of a simple ISO[2] message. For example, if the client was sending the withdrawal message to the switch with a specific structure until today, it may need to be updated the next day due to the changes made in the switch to adapt to the new structure of the withdrawal message. Therefore, changes may occur in the back layers of the system that affect all clients. In such cases, even current versions of user interfaces installed on clients may lose their functionality.

To solve this problem in distributed open banking systems, it is better to have a service or API in the system so that the structure of the input and output of any type of ISO 8583 standard message can be inquired by using it. That is, by sending two values of message type code (ISO-1 field) and

2 ISO-8583 Messages

processing code (Field P-3), the structure of the request message and the response message can be received. For example, if a message of type 0200 is queried from the center, depending on the type of processing requested, the fields related to that message should be notified to the client in a maximum of four types: 0200, 0210, 0220 and 0230. This can help develop clients that have automated adaptability and updating mechanisms.

12 Risks of DOB

Based on what has been observed in the previous sections, four major risks threaten the implementation of distributed open banking systems. The main ones are related to process zoning, which is also the root of next risks. The last risk is related to the technical field. These risks include:

- **Processing zoning risk:** Excessive complexity in zoning can have devastating effects on system performance. Also, increasing the level of zoning minority can significantly reduce the speed of distribution of transactions. Therefore, zoning should be done based on parameters that justify the level of zoning logically and functionally.
- **Load balance risk:** Failure to properly distribute operations in the system may have adverse consequences for the distribution of hardware resources. It is better to distribute the system load based on the zoning done.
- **Data partitioning risk:** As with transaction-related operations, data must be segmented according to the zoning performed to avoid centralization in data storage. Data partitioning is one of the best solutions for this issue. It is best to partition transaction data horizontally.
- **Change management risk:** Lack of awareness about possible changes in the back-end of the system can be very catastrophic for clients and service providers. Clients and service providers should be able to inquire about the structure of incoming and outgoing messages from the center at specific times (for example, at the beginning of each day) to be aware of possible changes.

13 Conclusion

Due to the development of communication tools and the growth of financial transactions and increasing their diversity in banking systems, the implementation of distributed systems to speed up the response and improve data security seems to be a debt. Open banking is one of the logical tools to implement such a system. But to ensure the distribution of the system and the efficiency of this operational division, we must consider the standards and concepts that guarantee the proper and principled development of a distributed open banking system. These actions can expose the system to a range of risks in the areas of processing, data, and networking. But proper management of these risks can increase the benefits of a distributed system relative to its risks.

14 References

[1] "Population, Population Change, and Estimated Components of Population Change: April 1, 2010 to July 1, 2020 (NST-EST2020-alldata)". Census.gov. United States Census Bureau. Archived from the original on December 22, 2020. Retrieved December 22, 2020.

[2] "Distributed Ledger Technology: beyond block chain (Report)". UK Government, Office for Science. January 2016.

[3] Scardovi, Claudio (2016). "Restructuring and Innovation in Banking". Springer. p. 36. ISBN 978-3-319-40204-8. Retrieved 21 November 2016.

[4] Iansiti, Marco; Lakhani, Karim R. (January 2017). "The Truth About Blockchain". Harvard Business Review. Harvard University. Archived from the original on 18 January 2017. Retrieved 17 January 2017

[5] Open Banking Working Group. "The Open Banking Standard" (PDF). HM Treasury. Archived (PDF) from the original on 18 April 2017. Retrieved 18 April 2017.

[6] Premchand, A.; Choudhry, A. (Feb 2018). "Open Banking APIs for Transformation in Banking". 2018 International Conference on Communication, Computing and Internet of Things (IC3IoT): 25–29. doi:10.1109/IC3IoT.2018.8668107.

[7] Brodsky, Laura; Oakes, Liz (September 2017). "Data sharing and open banking". McKinsey & Company. Archived from the original on 8 November 2017. Retrieved 7 November 2017

[8] John Fralick Rockart, Joav Steve Leventer. "CENTRALIZATION vs DECENTRALIZATION OF. INFORMATION SYSTEMS: A CRITICAL SURVEY OF CURRENT LITERATUR". REPORT CISR-23, SLOAN WP 845-76. April 1976. core.ac.uk

[9] Shirley Lu, Min Zhang ,Chongyuan Xiang. "Consistent Distributed Banking System". MIT CSAIL Computer Systems Security Group. 6.824: Distributed Systems. Spring 2014.

[10] Isabelle Simplot-Ryl, and Issa Traore and Patricia Everaere. "Distributed Architectures for Electronic Cash Schemes: A Survey". The International Journal of Parallel, Emergent and Distributed Systems, Vol. , No. 1–28. August 29, 2008.

[11] R.B. PATEL and K. GARG; "Distributed Banking with Mobile Agents: An Approach for E-Commerce"; WSEAS TRANSACTIONS ON COMPUTERS; Issue 1, Vol. 3, January 2004. ISSN: 1109-2750

[12] Dr. Sebastian Heuser, Dr. Simon Alioth. "The Open Banking Revolution". synpulse. Digital Banking. 2018.

[13] "BIAN Service Landscape 9.0". Banking Industry Architecture Network. Published on October 28, 2020

[14] “About BIAN”. Banking Industry Architecture Network. 2021

[15] Eduardo B. Fernandez, Nobukazu Yoshioka, and Hironori Washizaki. "Two patterns for distributed systems: Enterprise Service Bus (ESB) and Distributed Publish/Subscribe". ACM Digital Library. DOI: 10.1145/2578903.25799146. October 2011

[16] "What is load balancing? - Load Balancing Definition", Citrix, 2020

[17] "What Is a Reverse Proxy Server?", Nginx, 2020

[18] Aurel, “Setting up distributed database architecture with PostgreSQL”, DEV blog, 20 October 2019