Enhanced Proposal of Bitcoin Core

CRYPTONITE – Group #16

Amanda Misek – 19aam18@queensu.ca

Benjamin Tiong – 19bkyt@queensu.ca

Bilal Imran – 20mbi@queensu.ca

Christian D'Souza – 20ceds@queensu.ca

Decheng Zhu (Eric) – 18dz6@queensu.ca

Vanshita Uthra – 19vu1@queensu.ca

Table of Contents

1.0 Abstract

This report describes the enhancement of Bitcoin Core's architecture with the proposal of a cryptocurrency exchanger feature. With the proposal of our new feature in the system, we discuss the necessary changes to our current architecture in order to support our enhancement. We provide two different implementations of our new exchanger feature. Our report also provides an SEI SAAM Architectural Analysis, which provides the stakeholders, non-functional requirements, and impact of the enhancement. Furthermore, we make use of a sequence diagram to explain the flow of a use case involving our enhancement. Finally, we report the potential risks as well as the impact on our concrete architecture with the addition of our enhancement feature.

2.0 Introduction & Overview

The purpose of this report is to present our proposed feature as a way of enhancing Bitcoin Core's concrete architecture. The report is sectioned into parts that include the first implementation, second implementation, SEI SAAM architectural analysis, potential risks and limitations, testing, and use cases.

For our implementations on the proposed feature for Bitcoin Core, we look into two different ideas on how to add our proposed feature into Bitcoin Core's concrete architecture. The first implementation makes use of a new subsystem, while the second implementation involves the creation of a subcomponent within our existing concrete architecture of Bitcoin Core.

The SEI SAAM Architectural Analysis section makes use of the two implementations of the proposed feature, where we identify the major stakeholders along with their most important non-functional requirement, the impact of each enhancement to each stakeholder and NFR, and the choice of implementation based on the impacts of each implementation.

The potential risks and limitations section details all the risks that would be taken in implementing our proposed feature into Bitcoin Core's system, and the limitations of our attempts to provide implementations that maintain the integrity of Bitcoin Core's peer-to-peer architectural style.

3.0 Proposed Enhancement

Transactions are one of the main features of Bitcoin Core, where the Bitcoin currency can be transferred from one wallet to another. However, with the popularity of various other cryptocurrencies such as Etherium and Cardano, the use of third-party applications are

needed in order to create a transaction with other cryptocurrencies. Our group has proposed an enhancement of the Bitcoin Core architecture through the use of an exchanger functionality, where the Bitcoin currency can be sold or purchased with the use of other popular cryptocurrencies without the use of any third-party application.

4.0 Implementation 1

Our first approach for implementing the currency exchange into the Bitcoin software architecture is to add a module, the exchanger module, which can exchange Bitcoin with any other popular cryptocurrency in the market instead of using a third-party application to trade Bitcoin currency with other currencies/cryptocurrencies. This new module interacts with the connection manager, bitcoin protocol, wallet subsystem, RPC, & general testing/debugging. The exchanger interacts with the connection manager by establishing connections with external currency networks and exchanges. New network protocols and interfaces are implemented to communicate with these networks which would be integrated with the existing connection manager. The exchanger would also interact with the Bitcoin protocol to perform currency exchanges. This is done by creating new types of transactions or modifying existing ones to support the exchange of Bitcoin for other currencies. The exchange functionality is also integrated with the validation engine to ensure that exchanges are verified and validated according to the Bitcoin consensus rules. The exchanger module would need to interact with the wallet subsystem to manage user funds and transactions. The exchanger also interacts with the RPC system to display its functionality to external clients and applications. Finally, the exchanger module would need to go through testing and debugging to ensure that it is functioning properly and securely. This would involve a combination of unit testing, integration testing, and manual testing to verify that the module is interacting with the other subsystems correctly.

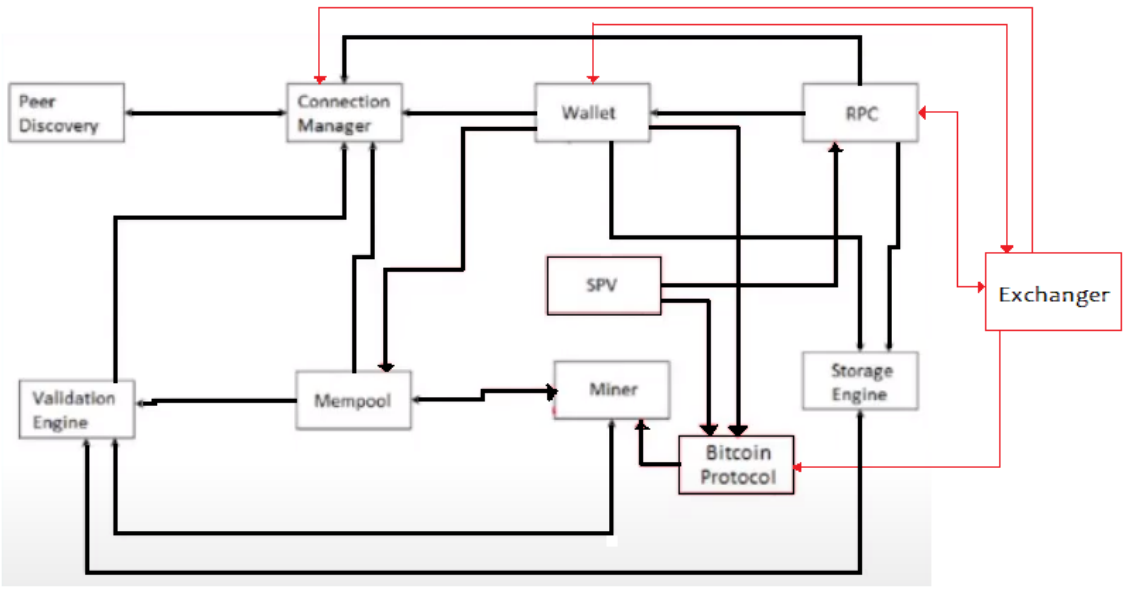

Figure 1: First Conceptual Change

5.0 Implementation 2

Our first approach for implementing the currency exchange into the Bitcoin software architecture is to add a module, the exchanger module, which can exchange Bitcoin with any other popular cryptocurrency in the market instead of using a third-party application to trade Bitcoin currency with other currencies/cryptocurrencies. This new module interacts with the connection manager, bitcoin protocol, wallet subsystem, RPC, & general testing/debugging. The exchanger interacts with the connection manager by establishing connections with external currency networks and exchanges. New network protocols and interfaces are implemented to communicate with these networks which would be integrated with the existing connection manager. The exchanger would also interact with the Bitcoin protocol to perform currency exchanges. This is done by creating new types of transactions or modifying existing ones to support the exchange of Bitcoin for other currencies. The exchange functionality is also integrated with the validation engine to ensure that exchanges are verified and validated according to the Bitcoin consensus rules. The exchanger module would need to interact with the wallet subsystem to manage user funds and transactions. The exchanger also interacts with the RPC system to display its functionality to external clients and applications. Finally, the exchanger module would need to go through testing and debugging to ensure that it is functioning properly and securely. This would involve a combination of unit testing, integration testing, and manual testing to verify that the module is interacting with the other subsystems correctly.

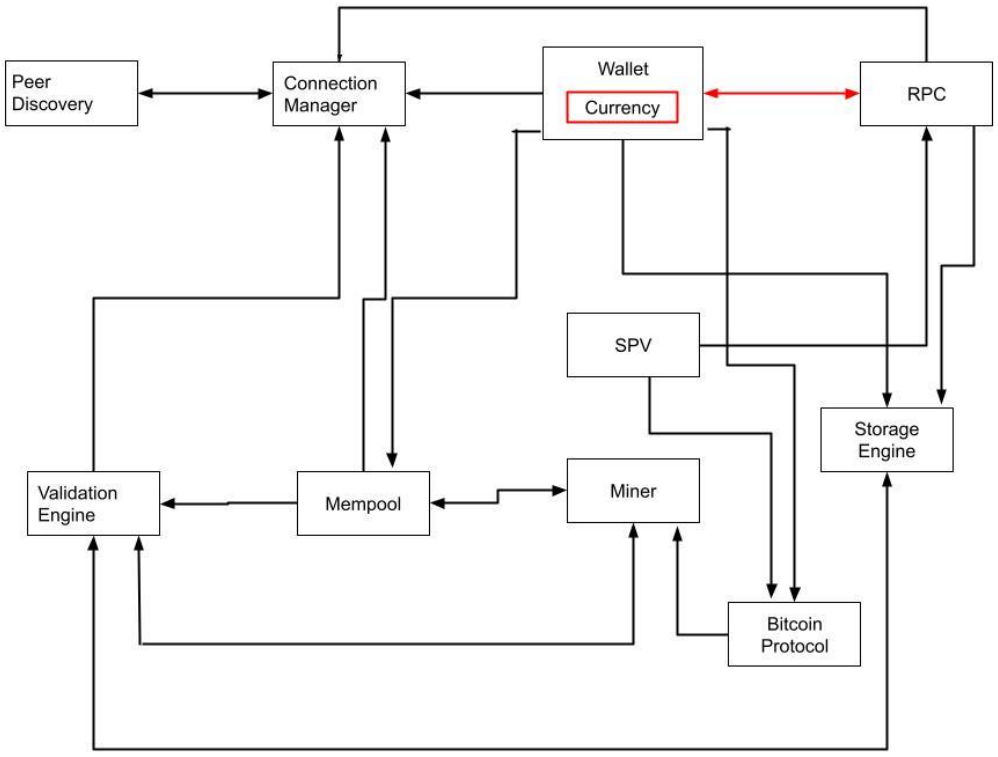

Figure 2: Second Conceptual Change

6.0 SAAM

6.0.1 The Major Stakeholders

There are three major stakeholders involved in the implementation of the new exchanger subsystem in Bitcoin Core, the developers, the exchange operators and the users.

The Bitcoin Core Developers: The developers responsible for maintaining and updating the Bitcoin Core software are considered key stakeholders in the implementation of the new exchange subsystem. They would need to ensure that the subsystem integrates smoothly with the existing codebase and does not introduce any bugs or security vulnerabilities.

The Exchange Operators: The exchange operators are responsible for implementing the exchange subsystem and making sure that it operates reliably and securely. They would need to work closely with the Bitcoin Core developers to ensure that the subsystem is compatible with the Bitcoin protocol and meets the needs of their customers.

The users: The users of Bitcoin who want to trade their Bitcoins for other cryptocurrencies are also considered to be stakeholders in the implementation of this new exchange subsystem. The users would have the ability to trade with different currencies with ease and have no worries about the security and reliability of the subsystem, as well as the availability.

Overall, the stakeholders involved in the implementation of this new exchanger subsystem would need to work together to ensure that the subsystem is implemented securely, and reliably.

6.0.2 The Most Important NFRs for each Stakeholder

Developers:
Scalability: The software should be able to integrate smoothly as the number of users and transactions increase.
Security: The system must be secure to avoid activities such as hacking and theft.
Reliability: The system should be able to process transactions rapidly while avoiding any errors and handling outages.
Maintainability: The developers should be able to run automated tests for all components

Exchange Operators:
Reliability: The exchange operators must ensure that the software is able to perform without any failure.
Security: The system must be secure against attacks.
Usability: The system should be easy to use for the user and must meet all their needs.

Users:
Availability: The system should be accessible to the user
Usability: The system must be easy to use so the user can get the full experience out of it.
Security: The system must be secure and the information of the user must not be leaked.

6.0.3 Evaluating The Impact On Each Identified NFR And Stakeholder

For users, the NFRs include availability, usability and security. As a result of the new implementations, this will allow the availability of the currency exchange to increase due to the less time taken to transfer to an outside application so will likely increase the system runtime. When it comes to usability, this will also increase as the ease of use on the customer side becomes less of a hassle. Finally, for security, since there is no need for a third-party application, the security risks decrease as their transactions are handled in one place.

For the developers, the NFRs listed include scalability, security, reliability and maintainability. After implementing the proposed systems, the scalability would increase for developers due to the ability to reuse code for the currency subcomponent. In terms of security, bitcoin developers would be able to maintain more control and handling of the exchanges. Similarly, developers will be able to maintain more reliability since there is no communication with the third-party application, thus also increasing maintainability through code reusability and easy access.

For the exchange operators, the following NFRs are reliability, security and usability. When it comes to the operators, the reliability increases since there is no outside application needed to be connected and contacted for these currency exchanges. Similarly, for security, these operators can oversee all transactions and be able to provide an in-house system to ensure minimal errors. Lastly, regarding usability, exchange operators will have the ability to find a seamless system which provides an easy-of-use solution to the many transactions.

6.0.4 Determining Which Enhancement Is The Overall Best

Implementation 1 is more suitable for users as it leverages the existing base components of the system, without the need for an additional subsystem inside the wallet component. However, implementing this approach would require additional effort and functionality to integrate all the necessary connections and rules. In terms of user experience, this approach would offer better availability, usability, and security.

Implementation 1 is better suited for developers as they have a greater scope of functionality they can cover, but also heavily increases the maintainability of the system. While it requires more effort to integrate all the connections and rules, it would be worthwhile if the goal is to implement a new exchanger with excellent functionality. In terms

of developer scope, this approach would offer better functionality, but sacrifice performance in terms of scalability, maintainability, and security.

Implementation 2 would be the ideal choice for exchange operators, especially if their team is smaller and not the primary focus of the project (which would be the case if Implementation 2 is chosen). As a smaller subsystem feature, it would be easier for the exchange operators to maintain and manage the system effectively. In terms of operational effectiveness, this approach would offer better reliability and security but suffers from its scope and usability.

Overall, Implementation 1 offers several excellent benefits for users and developers, which makes it a better but much more expensive and extensive enhancement than Implementation 2.

7.0.1 Use Case 1

Use case #1: Decentralized Payments (Peer-to-peer network)

To send out a payment in a distributed crypto payment system, the following steps are involved:

1. The user initiates a payment request through the Wallet Interface module, which serves as a graphical user interface between the user and the digital wallet.

2. The Storage Engine module contains the storage of the user's funds. Checking it using the Validation Engine module will return whether the user has sufficient funds.

3. The Exchanger module will exchange other currencies into Bitcoins according to the Bitcoin Protocol.

4. The Bitcoin Protocol module ensures that the transaction adheres to the rules and protocols of the distributed crypto payment system. This will check the SPV which is a subsidiary created by a parent company to isolate financial risk.

5. The Miner module will broadcast the new block to the network, and other nodes on the network validate the block and confirm that the transactions within it are valid.

6. The Validation Engine module will figure out whether the transaction request is valid or not.

7. The Connection Manager module receives the Valid request from Validation Engine and processes it according to the user's instructions. This may involve exchanging appropriate coins to use for the transaction, calculating the fees, and generating the necessary transaction data.

8. The RPC subsystem may be used to perform certain remote procedure calls, such as encrypting the transaction data or signing a message to be included with the transaction.

9. Finally, the Storage Engine module will store all relevant transaction data, including the main database, wallet database, and other databases used by the distributed crypto payment system.

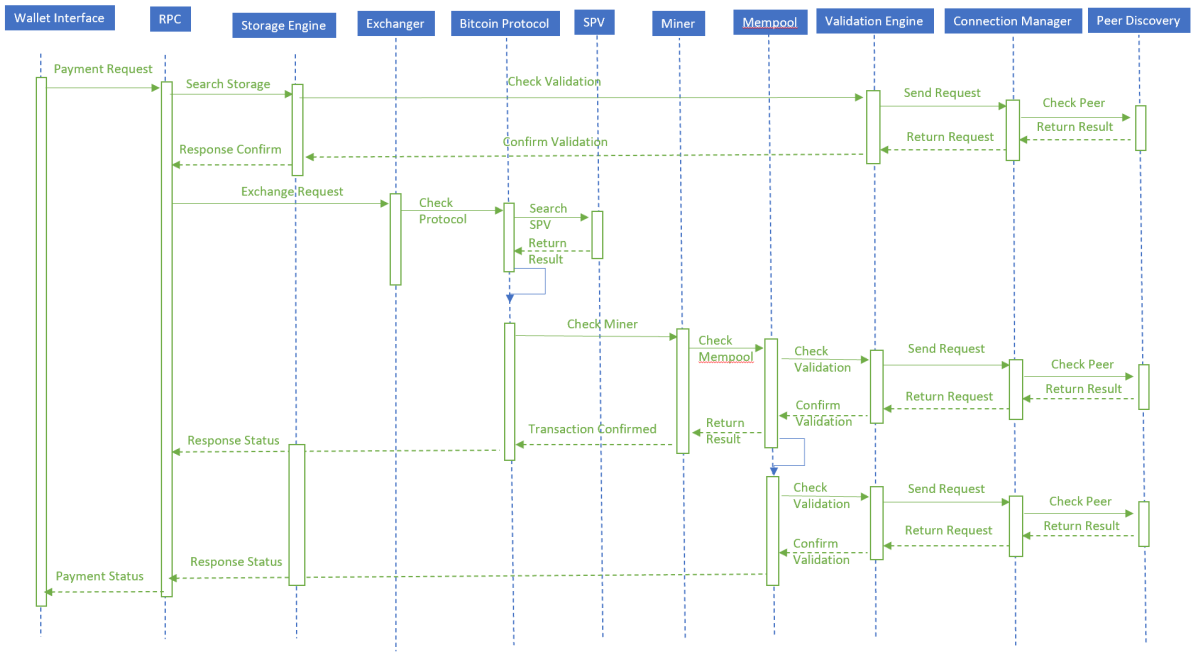

Figure 3: Use Case 1

7.0.2 Use Case 2

Use case #2: Verifying Transactions

To verify a payment in a distributed crypto payment system, the following steps are taken:

1. The user initiates a transaction by sending the required amount of cryptocurrency to the intended recipient's wallet address.

2. The Storage Engine module receives the transaction request and validates it, checking that the user has enough funds to cover the transaction and that the recipient's wallet address is valid.

3. The Bitcoin Protocol module ensures that the transaction adheres to the rules and protocols of the distributed crypto payment system. This will check the SPV which is a subsidiary created by a parent company to isolate financial risk. The Communication Manager facilitates communication between the Wallet Controller and Protocol Manager to ensure the integrity of the information passed between them.

4. The Miner module will broadcast the new block to the network, and other nodes on the network validate the block and confirm that the transactions within it are valid.

5. The Validation Engine module will figure out whether the transaction request is valid or not.

6. The Connection Manager module receives the Valid request from Validation Engine and processes it according to the user's instructions. This may involve exchanging appropriate coins to use for the transaction, calculating the fees, and generating the necessary transaction data.

7. The RPC subsystem may be used to perform certain remote procedure calls, such as encrypting the transaction data or signing a message to be included with the transaction.

8. The Storage Engine module will store all relevant transaction data, including the main database, wallet database, and other databases used by the distributed crypto payment system.

9. After the transaction is complete, the user can verify it by checking the transaction details on the blockchain, which is a public ledger that records all cryptocurrency transactions.

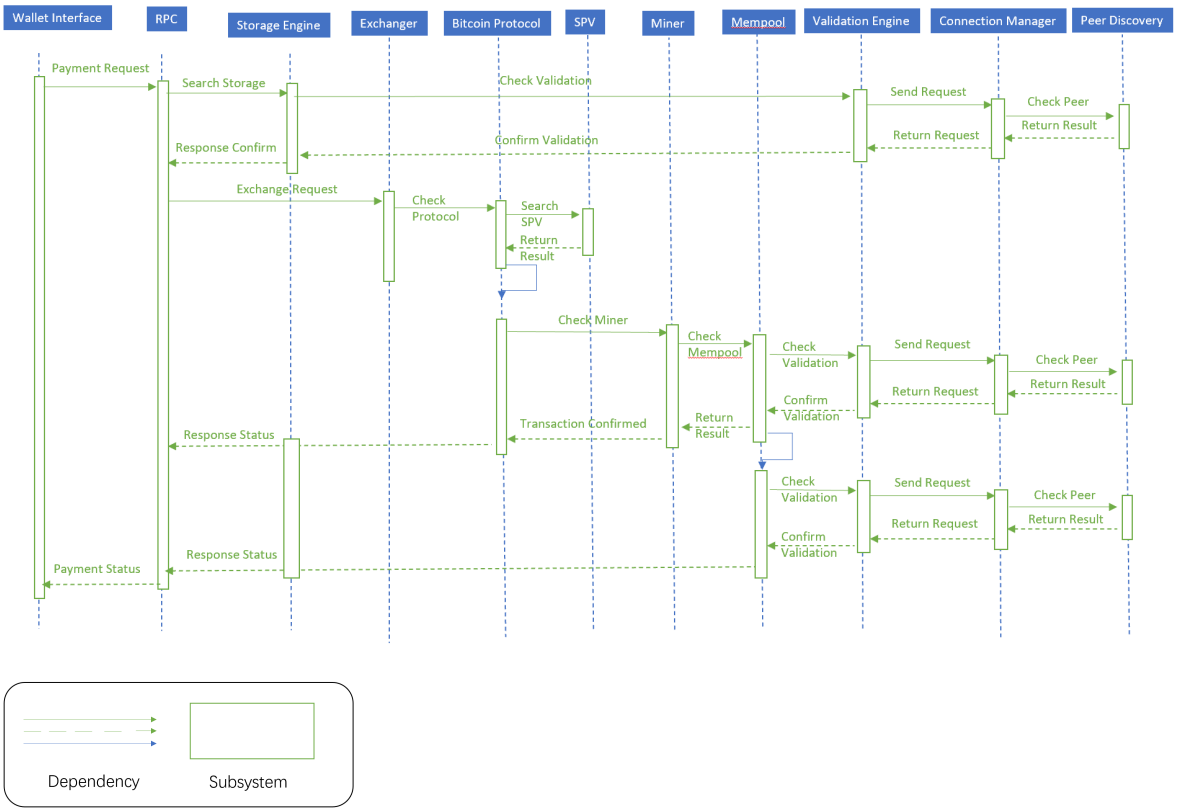

Figure 4: Use Case 2

8.0 Plan for Testing

Unit Testing:

- Verify that the exchanger module can establish connections with external currency networks and exchanges
- Test the implementation of new network protocols and interfaces to communicate with these networks, and integrate with the existing connection manager

- Ensure that the exchanger can perform currency exchanges by creating new types of transactions or modifying existing ones to support the exchange of Bitcoin for other currencies
- Test the integration of the exchange functionality with the validation engine to ensure that exchanges are verified and validated according to the Bitcoin consensus rules
- Verify that the exchanger module can interact with the wallet subsystem to manage user funds and transactions
- Test the implementation of the RPC system to display its functionality to external clients and applications
- Test the overall functionality of the module in different scenarios to ensure that it is functioning properly and securely

Integration Testing:

- Test the impact of the exchanger module on the pre-existing functionality of the Bitcoin software architecture
- Verify that the exchanger module interacts with the connection manager, Bitcoin protocol, wallet subsystem, and RPC system without causing any conflicts or issues
- Test the exchange functionality of the exchanger module with different currencies and cryptocurrencies to ensure that it is working as expected
- Verify that the exchanger module is integrated with the validation engine and is following the Bitcoin consensus rules
- Ensure that the module is functioning properly in combination with other modules of the Bitcoin software architecture

Manual Testing:

- Perform manual testing to verify the overall functionality of the exchanger module
- Test the user interface of the exchanger module to ensure that it is easy to use and intuitive
- Test the security of the module by trying to exploit any vulnerabilities or weaknesses.
- Verify that the module can handle different scenarios and edge cases, such as network failures or unexpected user behaviour

By combining these three testing approaches, we can ensure the implementation of the exchanger module for currency exchange is robust, secure, and fully functional within the Bitcoin software architecture.

9.0 Potential Risk

The implementation of our new exchanger module can come with some potential risks such as security risks, performance risks and user experience risks.

Security Risk: Any new module that interacts with the new exchanger module has potential security vulnerabilities that can be targeted by attackers. The exchanger module needs to be designed with security in mind and needs to go through various testing phases. For example,

there could be vulnerabilities in the exchanger module itself that could be exploited. If the module is not designed to handle proper input validation, then it can be attacked through injection attacks. Another security risk could be integrated with other modules. Since the exchanger interacts with the connection manager and wallet modules, any vulnerabilities or weaknesses in those could also impact the exchanger.

Performance Risk: Currency exchange involves complex calculations and processing that could impact the performance and scalability of the Bitcoin subsystem. The performance would be at risk when there would be a high volume of transactions occurring which could lead to a possibility of system failure.

User Experience Risk: The addition of the new exchanger module could potentially impact the user experience of the system, especially if it is not well-designed and user-friendly. For example, some risks could be loss of funds during transactions, privacy concerns and technical challenges.

10.0 Concurrency

Bitcoin Core provides software that connects the Bitcoin network and implements all the protocols. The concurrent abilities of the Bitcoin Core software are present in its ability to organize and operate simultaneous transactions and other interactions on the network. Concurrency is resembled through its performance, efficiency, and scalability. Regarding Bitcoin Core's performance, its ability to use multi-threading showcases many separate tasks being operated in parallel, thus executing concurrently. Additionally, concurrency is also an important piece of scalability within Bitcoin Core. The ability to adapt and expand the number of transactions occurring within the network and still being able to operate and manage them all is very important. The ability of the Bitcoin network is proving to provide a great hub for the Bitcoin Core to sync to provide an efficient process. Lastly, for efficiency, the system allows for numerous intertwined connections all throughout the network. Furthermore, as a result of the architectural structure being peer-to-peer, this allows many users, also known as nodes, to connect to the network and interact with each other through transactions and exchanges. The concurrent nature of Bitcoin Core allows for seamless interactions and connections between nodes, organized through the protocols and network.

11.0 Lessons Learned

While completing this project there were a few major lessons we learned that stood out. The first was being able to think of an enhancement, let alone two unique implementations for previously proposed architecture. We wanted to ensure the P2P structure of the Bitcoin Core system was still intact and simplified as possible so once we came up with the first implementation, the second was difficult to find. Likewise, the overall enhancement idea was difficult to implement without the use of the outside resources. We ended up dividing into teams and coming up with two different implementations as depicted in the report.

Secondly, a new concept learned in this assignment was the SAAM architectural analysis method. It was initially difficult to figure out the different viewpoints on all ends of the overall system including the exchange operators, developers, and users. It was difficult to come up with some of the NFRs as many of them seemed to align. In the end we learned a great deal on how to orient components and subsystems limiting outsourcing, and got to build our understanding of different viewpoints and who and what is affected as a result of our changes.

12.0 Naming Conventions

NFR - Non-Functional Requirements
SAAM - Software Architecture Analysis Method
P2P - Bitcoin Core's Peer-to-Peer Network
RPC - Remote Procedure Call

13.0 Data Dictionary

Stakeholders - An individual or group of individuals involved in the decision making of a system
Multi-threading - The ability to have multiple users on a system at once

14.0 Conclusion

This report discusses the architectural enhancement of the current Bitcoin Core system. Our group has proposed an enhancement through the use of an exchanger functionality where other popular cryptocurrencies can be used to purchase or sell Bitcoin currency without the use of any external applications. This enhancement is discussed through two possible implementations, introducing a new *Exchanger* component and adding a *Currency* subcomponent to the current Wallet component. The implementations explain the support of change, low-level interactions, high-level interactions and impact on the current directories. Furthermore, the report discusses the stakeholders and their non-functional requirements as well as the potential risks of implementing this enhancement. The use cases of the system are also stated and illustrated through a sequence diagram. In conclusion, by trying to edit and make changes to the proposed architecture from previous assignments, a deeper understanding of the architecture was obtained as we came up with some enhancements.

15.0 References

bitcoin developer. (n.d.). *Transactions*. Bitcoin. Retrieved April 11, 2023, from https://developer.bitcoin.org/examples/transactions.html

Daniel, Alan. “What Is Bitcoin Core Wallet – a Complete Review.” *Unblock.net*, 28 Sept. 2021, https://unblock.net/bitcoin-core-wallet-review/.

Editor. (2020, February 12). *Non-functional requirements: Examples, types, how to approach*. AltexSoft. Retrieved April 12, 2023, from https://www.altexsoft.com/blog/non-functional-requirements/#:~:text=Non%2Dfunctional%20requirements%20or%20NFRs,reliability%2C%20data%20integrity%2C%20etc.

Müller, et al. “The Bitcoin Universe: An Architectural Overview of the Bitcoin Blockchain.” *Lecture Notes in Informatics*, 2018, https://dl.gi.de/bitstream/handle/20.500.12116/16570/DFN-Forum-Proceedings-001.pdf.