A2. Concrete Architecture

Team 7

Slides by Jack Fallows & Jiacheng Liu

Link to Presentation

https://youtu.be/yFDSxUB8-9w

Group Overview

- Jack Taylor (Team Leader) → 17jmt5@queensu.ca
- Jack Fallows (Presenter) → 19jef2@queensu.ca
- Jiacheng Liu (Presenter) → 19jl193@queensu.ca
- Maninderpal Badhan → 19msb14@queensu.ca
- Gabriel Lemieux → 19gml2@queensu.ca
- Nil Shah → 20ns3@queensu.ca

OVERVIEW

Introduction

- Conceptual architecture → intended as a blueprint for software
- Concrete architecture → realization of that blueprint
- How do we find things that may have gone wrong in our implementation?

- **Solution:** reflexion analysis

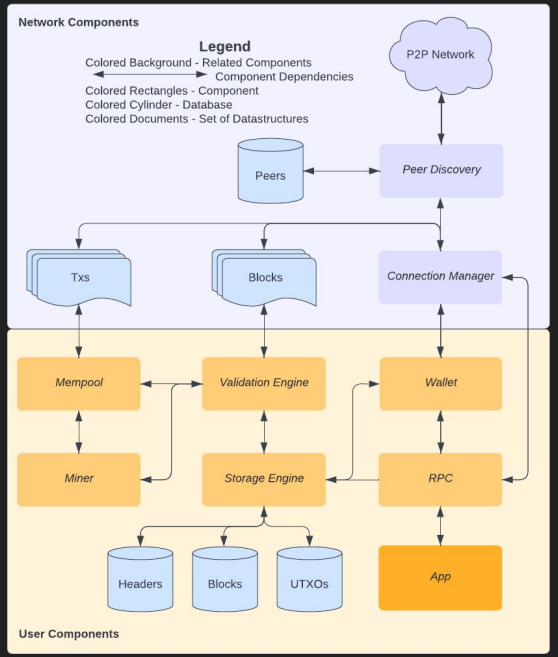
Overview of Procedure

- First, derive overarching concrete architecture for system
- Then, derive concrete architecture of a subsystem
- Finally, compare the derived architecture with the previously determined conceptual architecture

I. Top-Level Architecture

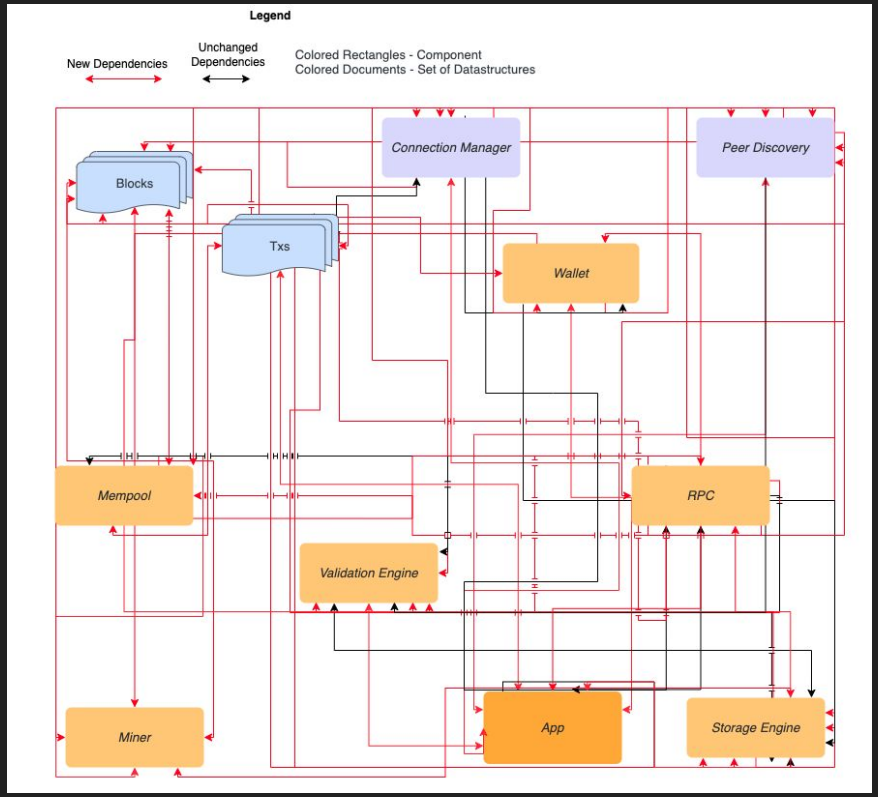
Conceptual Top-Level Architecture

- Top-level architecture was determined in A1
- Core components were used to derive the concrete architecture



Concrete Top-Level Architecture

- Lots of divergences
- Some only represent 1-2 dependencies - others represent 100+



Reflexion Analysis - Notable Divergences

- Wallet -> Transactions
  - Intuitively justified
  - Wallet needs to access transactions in order to validate ownership of coins
- Blocks -> Storage engine
  - block.h and block.cpp are grouped with the Storage Engine subsystem
- Transactions -> RPC
  - Txs require “univalue” to encode information before communicating it
- Peer Discovery -> Validation Engine
  - Peer discovery receives information about active chain from validation engine
- App -> Wallet
  - App needs to display wallet information such as balance and recent transactions
- RPC -> Blocks
  - RPC makes many calls to chain.h which is found in the Blocks subsystem

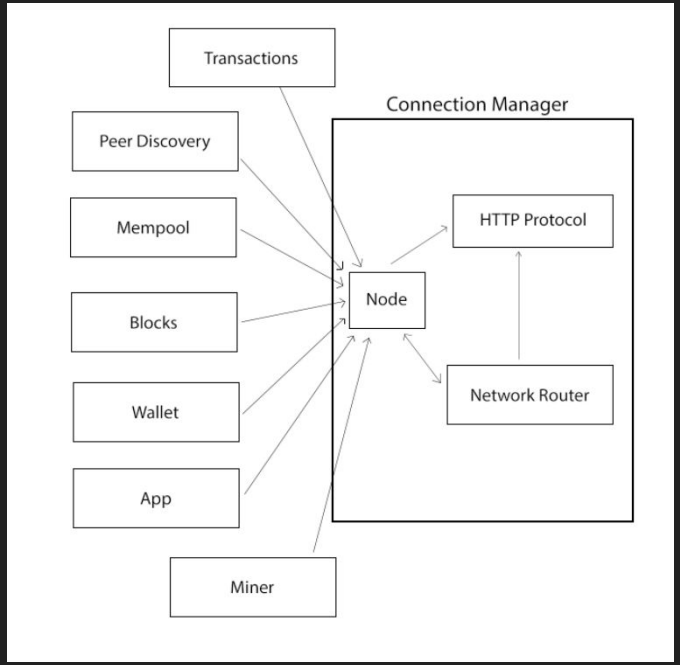
II. Subsystem Architecture

Overview

- Connection manager subsystem
- Allows nodes to collectively maintain Bitcoin network
  - P2P system allows for decentralized currency
- Also known as “network manager”

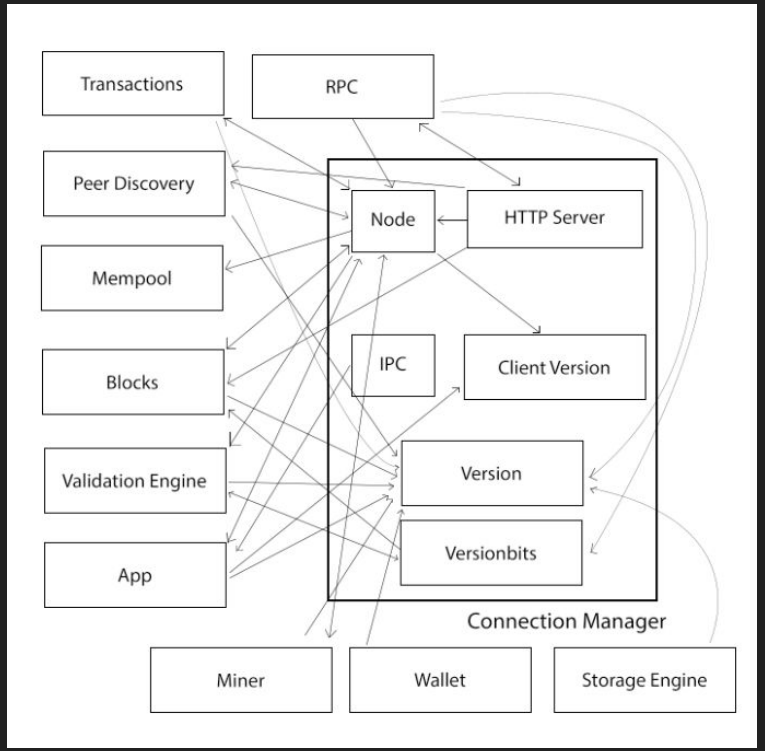
Conceptual Subsystem Architecture

- Node subcomponent manages all node-related functionality
- Network router and HTTP protocol manage networking functionality
- Outside dependencies from:
  - Peer Discovery
  - Mempool
  - Blocks
  - Wallet
  - App
  - Miner



Concrete Subsystem Architecture

- New components in concrete architecture
  - IPC
  - Version
  - Versionbits
  - Client version
- Validation engine and storage engine also included



Reflexion Analysis - Notable Divergences

- httpserver
  - RPC
  - Peer Discovery
  - Blocks
- Versionbits
  - Blocks
  - RPC
  - Validation Engine
- Clientversion <- App
- Version
  - Blocks, Validation Engine, Miner, Wallet, Peer Discovery, Transactions, RPC, Storage Engine, App
- IPC -> App

III. Derivation Process

Process

- Downloaded source code and sorted files into subsystems using Understand
- Manually checked each dependency to determine divergences
- Examined use of dependency in code to determine if divergences were justified and why
- Delegated work by project segment (top-level architecture, subsystem architecture, presentation)

Lessons Learned

- Should have extended conceptual architecture recovery process to consider new sub-components
  - Could have helped mitigate divergences by more effectively sorting files
- **Bitcoin Core source code is very disorganized**
  - Difficult to organize files into conceptual architecture categories when so many miscellaneous files are in /src
  - Emphasizes the importance of organization when working on large systems

THANK YOU FOR LISTENING