Trade Net Trade & Portfolio Management Software:

Software Architecture Description

Srikar Devarakonda(sd1454)
Armand Nokbak Nyembe(an499)
Narendra Kumar Mallela(nkm111)
Vijayant Kanwar(vk198)

CSE 4233/6233
Fall 2015

Table of Contents

1. Architectural Documentation (Document Control Information)

1.1. Date of issue and status

Wednesday, 12/9/2015.

1.2. Issuing organization

Trade Net Brokerage and Financial Services.

1.3. Change history

Date	**Change Log**
11/20/2015	Document started.
11/27/2015	Added identification of stakeholders and their concerns
11/30/2015	Added logical structure
12/03/2015	Added architectural views
12/08/2016	Added architectural rationale and consistency among views
12/09/2015	Final touches and concluded the document

1.4.Summary (System Overview)

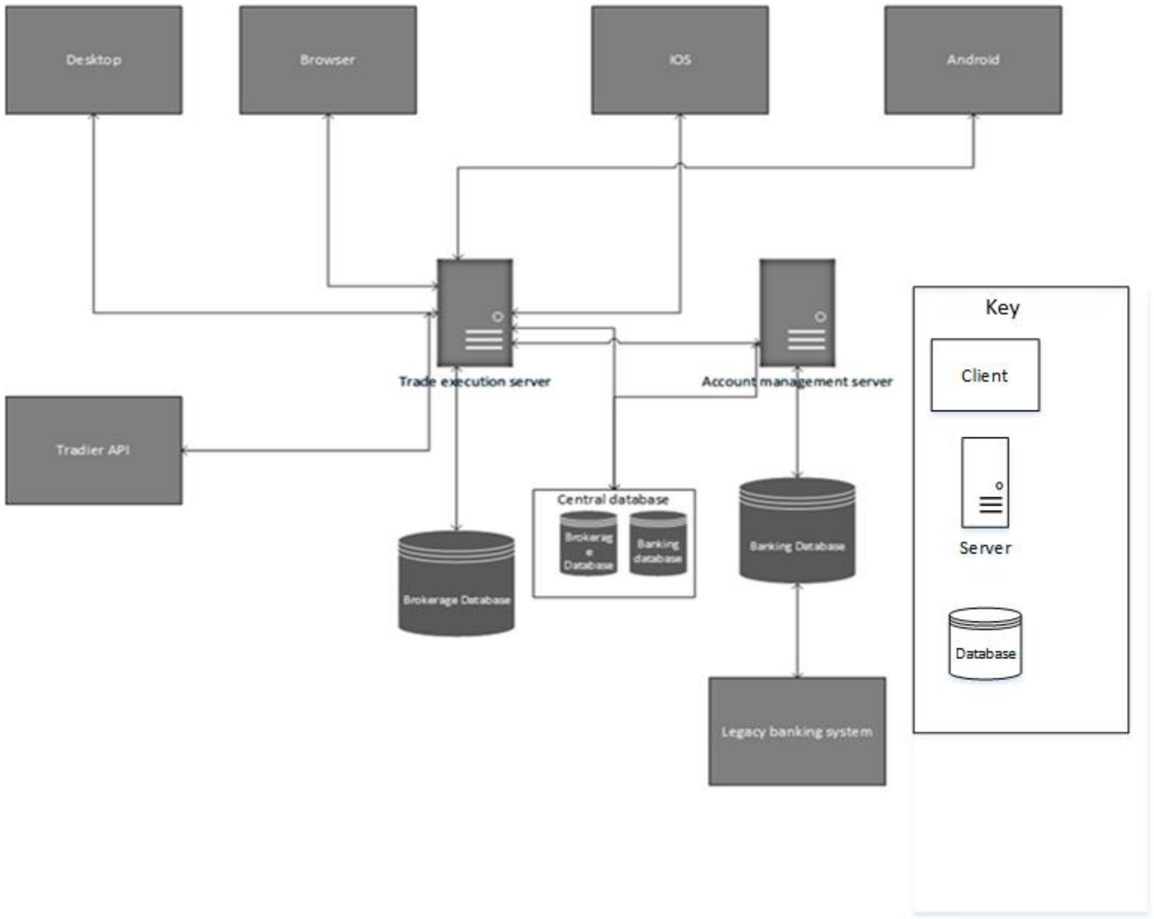

Figure 1: TNBS Context Diagram

1.5.Context

The TNBS helps bank customers purchase stock shares from their bank accounts. The system offers four (04) different clients that let the customers see what shares are available, what they cost, but also let them purchase them using their bank account’s balance. The TNBS is represented by the context diagram above.

1.6.Glossary (A-Z)

- AD: Architecture Document (this document).
- Application : “It is a software program that runs on a computer” [1].
- Browser: “A browser is an application program that provides a way to look at and interact with accessible information on the world wide web” [3]
- Clients: Computer program or hardware that accesses services provided by a server.
- COTS: Commercial off-the-shelf software.
- Database: “ A database is a collection of information that is organized so that it can easily be accessed, managed, and updated” [2].
- Libraries: “Collections of similar objects which are used occasionally” [4].
- Server: Computer program or hardware that services requests from other applications called clients.
- TNBS: Trade Net Brokerage System.

1.7.References

[1] Christensson, Per. "Application Definition." *TechTerms*. Sharpened Productions, 12 October 2008. Web. 20 November 2015. http://techterms.com/definition/application>.

[2] Leake, Allan. "What Is Database? - Definition from WhatIs.com." *SearchSQLServer*. Techtarget, 1 Apr. 2006. Web. 20 Nov. 2015. <http://searchsqlserver.techtarget.com/definition/database>.

[3] "What Is Browser? - Definition from WhatIs.com." *Browser Definiton*. WhatIs.com, 1 Nov. 2007. Web. 20 Nov. 2015. <http://searchwindevelopment.techtarget.com/definition/browser>.

[4] "What Is Library? - Definition from WhatIs.com." *SearchSQLServer*. Web. 20 Nov. 2015.

[5] Whas is iOS clinet? -Definition from WhatIs.com." *Browser Definiton*. WhatIs.com, 1 Nov. 2007. Web. 20 Nov. 2015.

http://www.webopedia.com/TERM/I/iphone_app.html

1.8.Acknowledgements

- Armand Nokbak
- Narendra Mallela
- Srikar Devarakonda
- Vijayant Kanwar

2. Identification of Stakeholders and Concerns

2.1 Stakeholders

In this document, the stakeholders are the parties that affect or are affected by the TNBS. They are listed below by alphabetical order.

2.1.1 Developers (Software):

The software developers are the programmers responsible for coding the application.

They care about the architecture document (AD) because they need to know and understand the different components and modules of the system, their interactions, and the protocols used to communicate and transmit data.

2.1.2 __Project Manager__:

The Project Manager is the person responsible for making sure that all the functional and non-functional requirements of the software are met within schedule.

He is concerned about the architecture document (AD) as he wants to make sure that the system covers all the aspects and requirements.

2.1.3 __Database Administrator__:

The Database Administrator is responsible for the maintenance of the database. From the AD, he gets the structure of the database models, the database size and performance, and the number of people interacting with the database to provide the stakeholders with the access specified in this document.

2.1.4 __Information Technology Manager__:

The Information Technology manager is the person in charge of the hardware and software infrastructures within the Trade Net Brokerage and Financial Services organization.

He cares about the AD because he needs to know where to deploy every software artifact and the properties needed by them, such as what ports should be available, what processor speed is needed, and what security and access measures should be implemented.

2.1.5 ***Business Analyst:***

A business analyst is the person who bridges the gap between the business and the technical teams. He is responsible for gathering requirements from the business side of the company and change it into system specifications.

He is concerned about the AD to make sure that it addresses all the business requirements.

2.1.6 ***Software Quality Assurance Analyst:***

The Software Quality Assurance Analyst is the person responsible for managing and executing all the activities for the end to end testing of the software. He should understand the possible causes of failure for each module and component in the architecture document so he.

2.1.7 ***Maintenance:***

Once the software has been deployed, the software maintenance team is concerned about the quality of the product. They use the AD to know the exact module in the architecture where they can fix defects and add patches to make modifications.

2.2 Concerns

2.2.1 Software developer (programmer):

- Does the AD shows sufficient details to guide the development?

The programmer needs to find enough guidelines in the AD to design system artifacts.

- Does the AD give enough references for assembling components?

Since the TNBS incorporates several components, the programmer needs to know how to assemble them.

- Is the system as described in the AD, compatible with existing systems?

The TNBS must be compatible with the existing legacy banking system.

2.2.2 Project manager:

- Schedule estimation

The Project Manager is responsible for estimating and scheduling tasks. He/She can find each module in the AD of TNBS and can assign the tasks based on the dependencies and priority.

- Feasibility and risk assessment

Since all the views are documented in the AD of TNBS, the project manager can access the possibility of risk from all perspectives in order to reduce the possibilities and consequences of said risks.

- Requirements traceability

The Project Manager can use views from the AD to trace back the system’s requirements.

2.2.3 Database Administrator:

- Structure and type of the database.

Based on the type of users (privileges) who will be accessing the TNBS and also the scope of the system, the database admin can easily identify what kind of database best suits the security level required by the project. The Data model view offers more insight on the database schema definitions to implement.

- Establishing the needs of users and monitoring user access and security.

The Uses Style view can give an idea of how each database will be accessed users, and what privileges to grant them.

- Developing, managing and testing back-up and recovery plans

By analyzing the security risks through all the views of the TNBS AD, the Database Administrator can take regular backups to avoid the risk of losing the data.

2.2.4 Information Technology Manager:

The Information Technology Manager will use the AD's deployment view to know where to place each of the system artifacts. He/she will also use the AD to figure out non-functional requirements such as performance and security in order to provide the hardware components to meet those requirements.

2.2.5 Business Analyst:

The Business Analyst gathers requirements and through the AD, verifies that all functional and non functional aspects of the system are met.

2.2.6 ***Software Quality Assurance Analyst:***

Using the AD, the Software Quality Assurance Analyst can plan and design the test cases to make sure that all the functional and non functional requirements of the system are met.

2.2.7 ***Maintenance:***

- Maintain compatibility with existing systems

The maintenance department uses the AD as guide to ensure compatibility as the system evolves.

- Guidance on software modification

Software can be written based on the functionality of each module which are addressed by the AD through the module view and the uses view.

3. Architectural views :

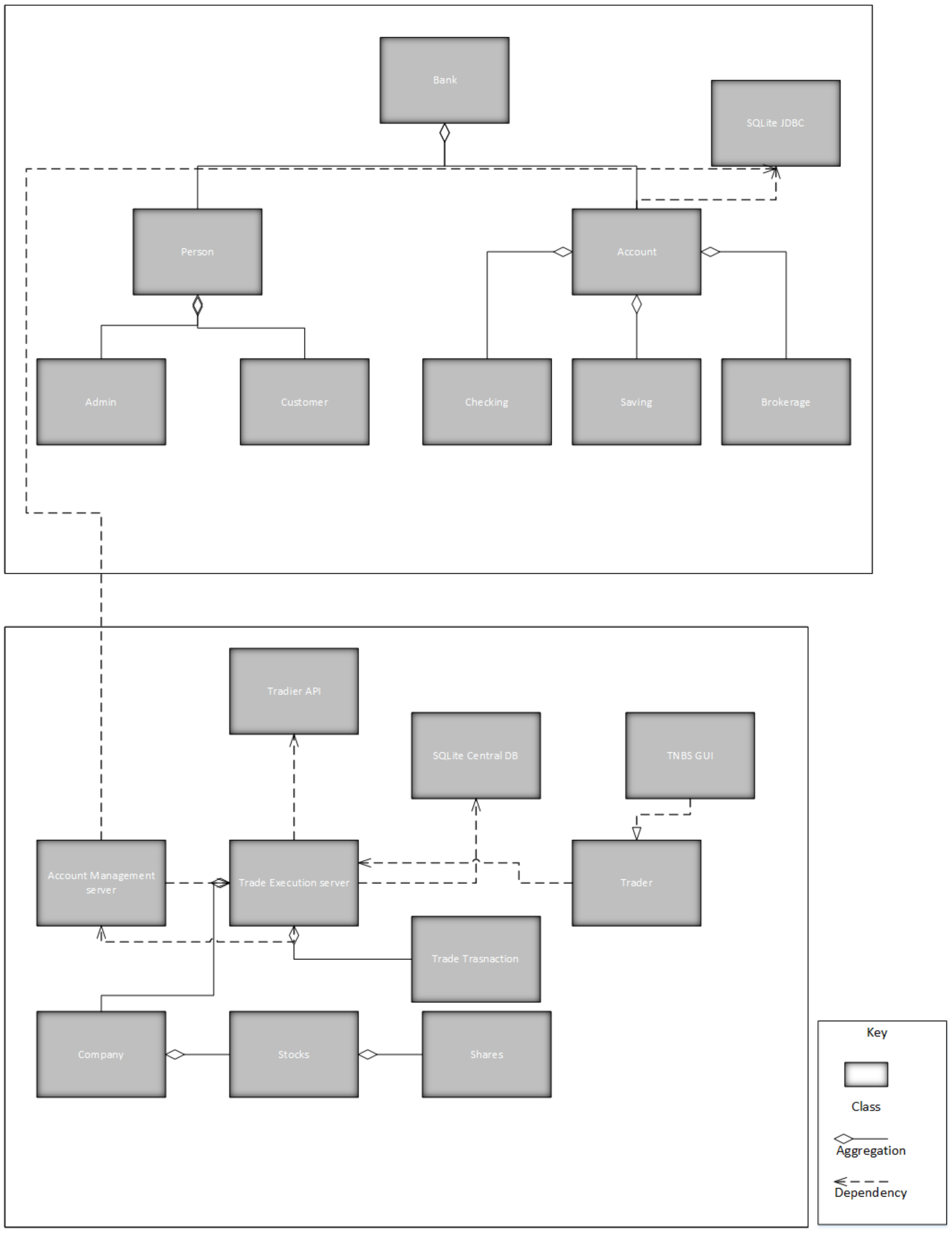

Figure 2: Class diagram of the TNBS system

Element Catalogue :

Element	Description/Function
Bank class :	The parent class of the legacy banking system responsible for banking transactions.
Person class:	Parent class for admin and customer responsible for the classification of the users of the system.
Admin class:	Admin is responsible for managing the user accounts of the bank.
Customer class :	Customer is user of the system who performs transactions.
Account class:	Account is responsible for the classification of the user accounts savings and checking.
Checking class :	The checking deals with all sorts of cash transactions made by the customer having a checking account.
Savings class :	The savings deals with all sorts of cash transactions made by the customer having a savings account.
Brokerage class:	Brokerage deals with the buying and selling of shares based on the account balance.
Transaction class:	Transaction deals with all money transactions and maintains a log for all the transactions.
SqlLite JDBC:	The database which stores all the details about the users and the transactions they made.
Account Management Server class:	Responsible for communicating with

	legacy banking system and aid in transactions.
Trade Net Execution Server class:	Authenticates the users and aids the users in buying and selling the shares.
Tradier API class:	Responsible for providing real time market data.
Company class:	Gives the list of companies and their share value.
Stocks class:	Contains the value of stocks of the companies.
Shares class:	Contains the individual share values of the companies.
Trader class:	Helps the user is buying and selling the shares by selecting the appropriate menu.
Trade transcation class:	Handles the trade transactions of the user
SqlLite_Central class:	Stores the buying and selling history of the user.
TNBS GUI class:	The user interface of the brokerage system.

3.1 Module Views :

***3.1.1 Layered view*:**

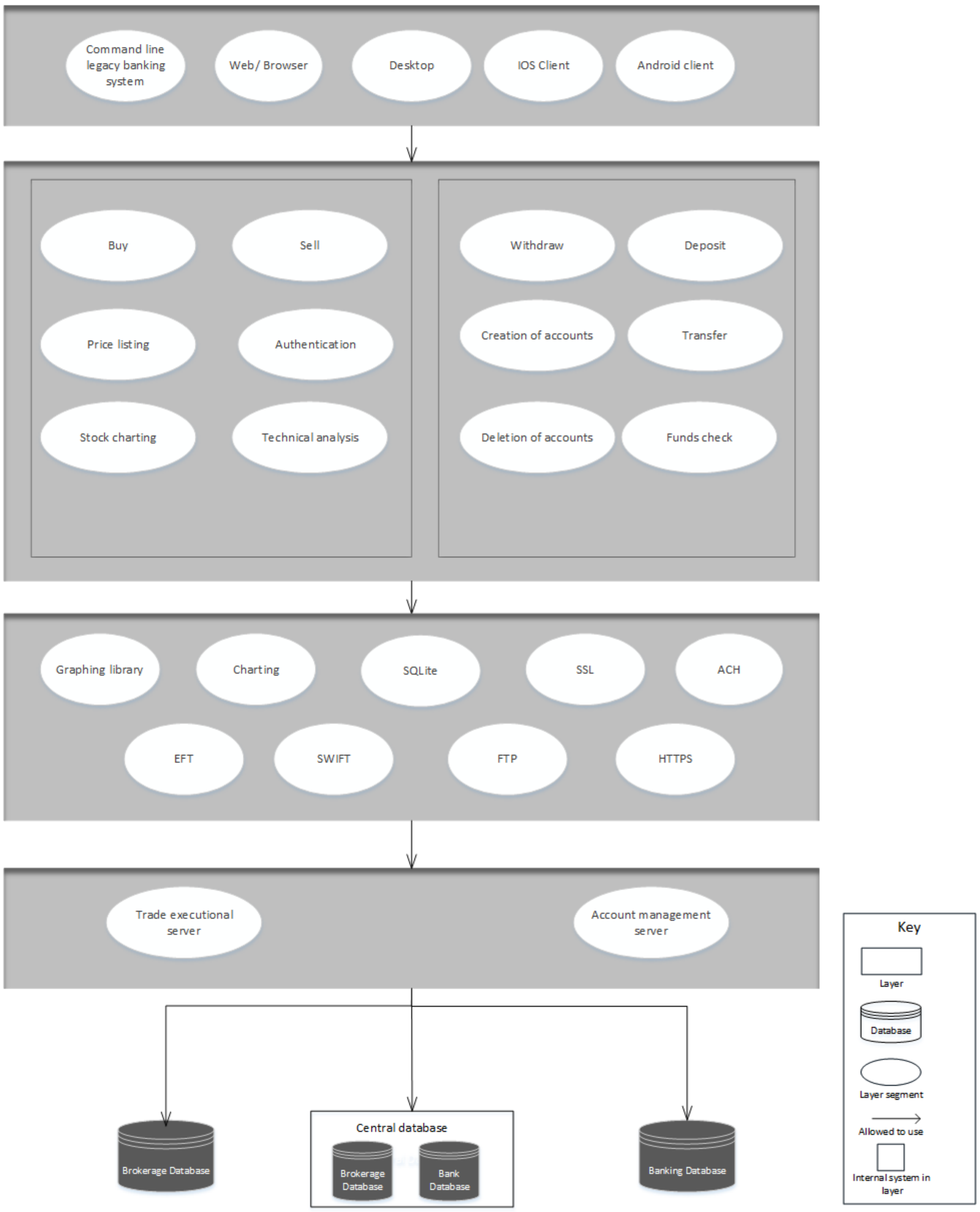

Figure 3: Layered view architectural style

Based on the above Layered Style view, the modules are :

- Command Line Banking System
- Web/Browser Client
- Desktop Client
- iOS Client
- Android Client
- Buy
- Sell
- Price listing
- Funds check
- Stock charting
- Technical Analysis
- Authentication
- Withdraw
- Deletion of Account
- Creation of Account
- Transfer
- Deposit
- Trade Execution Server
- Account Management Server
- Banking Server
- Central Database
- Banking Database
- Brokerage Database
- Tradier API
- Graphing service
- Financial charting service

***<u>Description</u>*:**

The layered style consists of the 5 layers which completely separates the software to provide a unidirectional relation with the layers below that.

The topmost layer contains the set of services which contains interface used by clients. These include legacy command line banking system through which the customers can do the bank transactions and three type of clients through which TNBS can be accessed through desktops from workstations and through iOS, and Android applications from mobile phones of the iOS and Android.

The second layer from the top contains all the functions that can be performed by clients through the system. Functions such as buying and selling of shares, stock charting, and authentication can be performed on the second layer’s first sub module and the functions such as withdrawal, depositing, and checking funds can be found on the second layer’s second module.

Third layer contains all the protocols through which the system can connect to the external libraries like graphing, charting for stock charting, SSL protocol that can be used for authentication, https which can be used for secured connection, transaction protocols like ACH, swift, EFT for safe transactions and SqLite for transacting with the database.

Fourth layer from top consists of the servers which perform the major operations. Trade execution server can be used for buying, selling of stocks, authentication of users with the account management server. Account management server stores all the account information and its along with the trade execution server can be used for authenticating users and Banking server can be used for storing the bank customer details.

Bottom layer contains all the databases. Central database contains the details of all the transaction details and logs. Banking database contains all the customer bank details and the brokerage database contains the details of the transactions done through the trade execution server.

Element Catalog:

Element	Description/Function
Command Line Banking System	Interface for the banking system.
Web/Browser Client	Helps in accessing the brokerage system via browser.
Desktop Client	Helps in accessing the brokerage system via desktop.
iOS Client	Helps in accessing the brokerage system via iOS device.
Android Client	Helps in accessing the brokerage system via Android device.
Buy	Helps in buying the shares.
Sell	Helps in selling the shares
Price listing	Displays the prices of the shares of the companies
Funds check	Helps in checking whether they have sufficient funds.
Stock charting	Helps in stock charting.
Technical Analysis	Helps in technical analysis.
Authentication	Helps in user log-in.
Withdraw	Helps in withdrawing the funds.
Deletion of Account	Helps in deletion of the user accounts
Creation of Account	Helps in creation of the user accounts.
Transfer	Helps in transferring the funds.
Deposit	Helps in depositing the funds.
Trade Execution Server	Helps in retrieving real-time market data
Account Management Server	Helps in checking the availability of the funds.

Banking Server	Helps in performing banking transactions.
Central Database	Stores the data of the TE server and AM server.
Banking Database	Stores the data of the banking server.
Brokerage Database	Stores the brokerage transaction details of respective users.

Rationale:

The Layered view can address the concerns of the developer. He needs to have details of the system with the differentiation of each functionality. He also needs to have an idea about how to assemble the components which can better understood by this view as it can give clear picture of the back end and front end. In future if we want to newly available COTS product it can give the compatibility level of the new component with the existing component.

3.1.2 *Uses Style :*

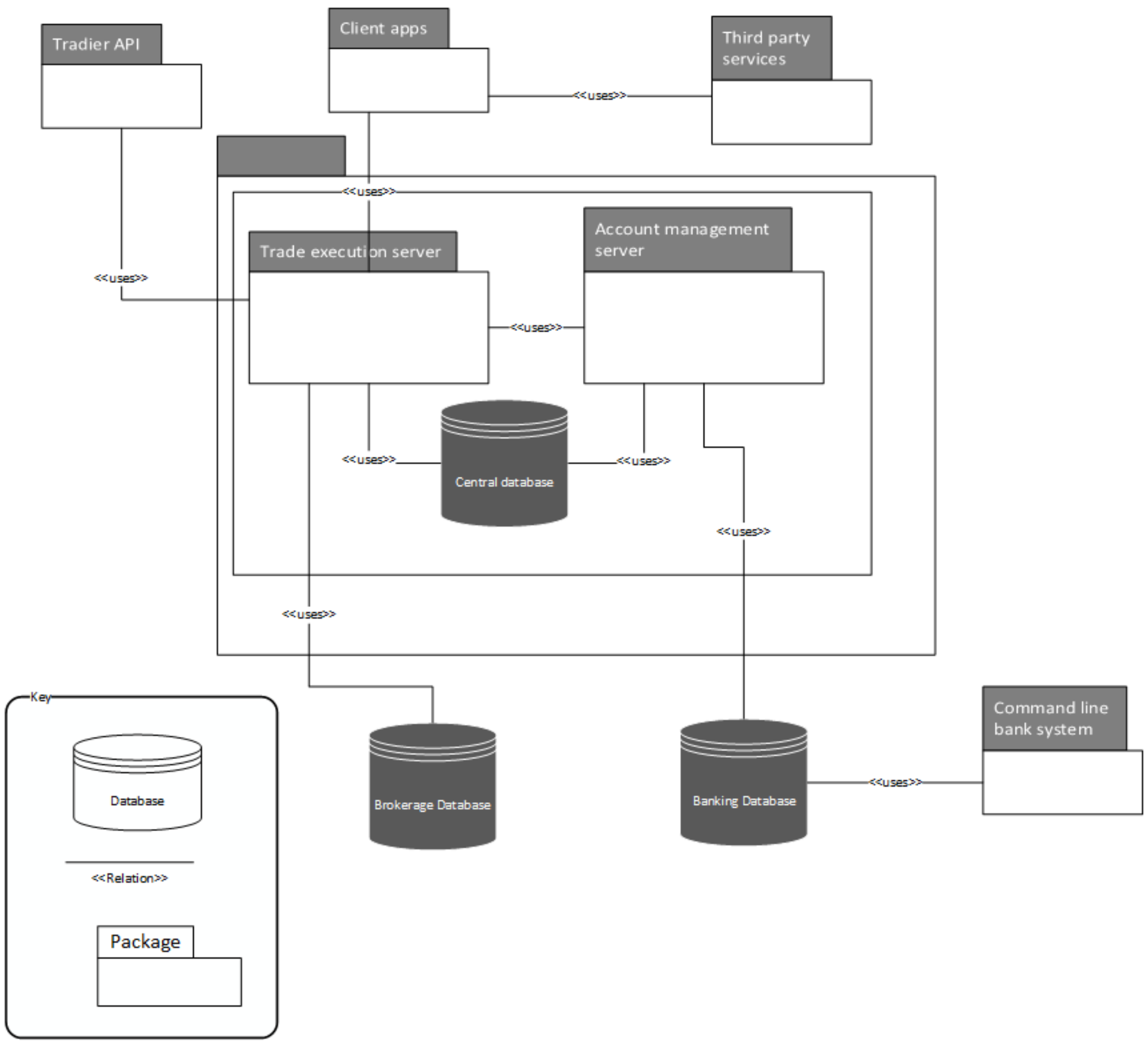

Figure 4: Uses style

The software modules in the system are

- Command Line Banking System
- Clients(Web/Browser Client, Desktop Client, iOS Client, Android Client)
- Trade Execution Server
- Account Management Server
- Banking Server
- Central Database
- Banking Database
- Brokerage Database
- Tradier API
- Third Party Services (Graphing service & financial charting service)

Description:

The uses style mainly addresses the trade execution server and the account management server which are the two main components of the TNBS. As shown here, clients use the trade execution server to perform operations like buying and selling shares. These clients applications also use third party applications like graphing and charting libraries to represent stock charts. The Trade execution server uses the Tradier API through which we can get the real time market data and it also uses the account management server which can be used for authentication of the users, as well as the brokerage database to store all transactions performed by the server and central database. The Account management server is used to authenticate user credentials. it uses the central database to store all the transactions through the account management server. It uses the bank database to store all bank customer details.

Element Catalog :

Element	Description/Function
Client Apps	Helps in accessing the Trade Net application.
Legacy Banking system	Helps in accessing the banking system via command line interface.
Trade Execution Server	Helps in retrieving real-time market data
Account Management Server	Helps in checking the availability of the Cash.
Central Database	Stores the data of the TE server and AM server.
Banking Database	Stores the data of the banking server.

Brokerage Database	Stores the brokerage transaction details of respective users.
Third Party Services	Provides financial charting and Graphing functions for the customers.
Tradier API	Provides real-time market data to the customers.

Rationale:

This view can better address the concerns of the maintenance team. In future, it helps to locate problems and also to identify alternative solutions. It can also be used to identify dependencies between components.

3.1.3 *Data Model Style* :

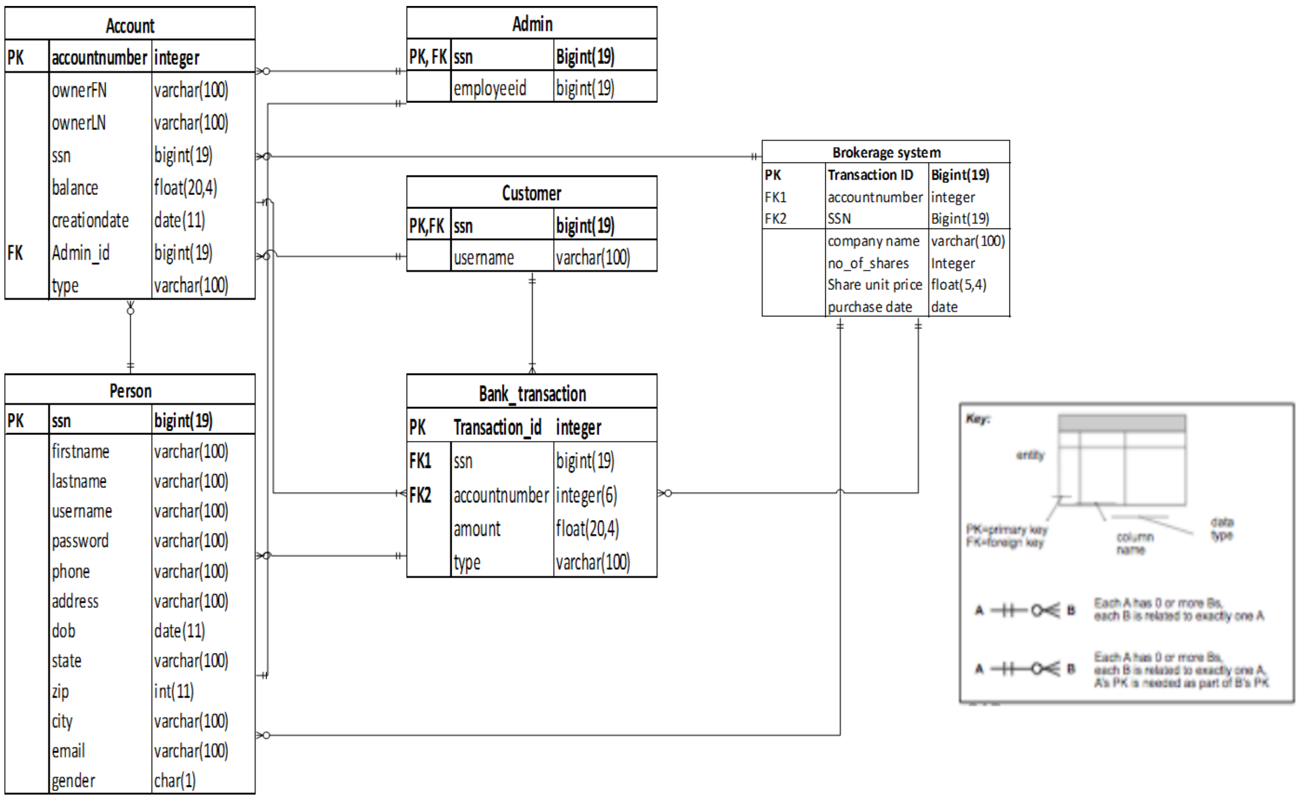

Figure 5: Data model view

The modules in the data model view are

- Person
- Account
- Bank Transactions
- Admin
- Customer
- Brokerage_Transaction

<u>Element Catalog :</u>

Module	Description/ Function
Person	Person consists all the details about the person who creates an account in order to access the Trade Net System.
Account	Account consists of the type of the account details the user uses in order to purchase the shares.
Bank Transactions	Stores all the transaction details of the users.
Admin	Contains the details of admin of Trade Net System.
Customer	Contains the log-in details of the customer
Brokerage_Transactions	Contains the transactions made on the Trade Net system.

<u>Description:</u>

The different entities in our Trade Net System are Person, Account, Bank Transactions Admin, Customer, Brokerage_Transaction. The Person entity has firstname, lastname, username, password, phoneno, address, gender, SSN as attributes and stores the details.The account entity has ownerlastname, ownerfirstname, balance, creationdate, type, accountnumber, SSN, adminid as attributes. The Bank_Transaction has SSN, accountnumber, amount, transactionid as attributes. Admin entity has empid as attribute. Customer entity has username, password as attributes. Brokerage_Transaction entity has

SSN, accountnumber, transactionid, companyname, nofshares, shareunit, purchase price as entities. Each entity reference the other entities with the help of foreign key and primary key references.

Rationale :

The data model view can better address the concerns of the database administrator and his team. It deals with all the attributes that each entity holds, the entities and the relationship between them. This view helps the database admin to structure the database to carry out his tasks.

3.2 Run-time Structure and Dynamic Views:

The component and connector (C&C) view illustrates the union of all the possible execution traces, and combination of these items such as an execution trace. While designing an architecture for the system, we tend to include all the functionalities of the system i.e. we are including all the components needed, these components need suitable interfaces to interact with each other so we include connectors for these components. If we take a particular component and want to analyze that particular execution trace, we can do that by selecting the particular components architecture. For example, consider a client-server architecture in which a client accessing the server of the banking system. This can provide the execution trace of a particular transaction from the entire system. If we want to analyze, how the entire system is working and responding in different execution traces we can do that for the component and connector style as it includes all the functionalities of the system. This can provide a execution trace of complete systems processes: A component and connector (C&C) view illustrates the union of all the possible execution traces, and combination of these items such as an execution trace. While designing an architecture for the system we tend to include all the functionalities of the system i.e. we are including all the components needed, these components need suitable interfaces to interact with each other so we include connectors for these components. If we take a particular component and want to analyze that particular execution trace, we can do that by selecting the particular components architecture. For example, consider a client-server architecture in which a client accessing the server of the banking system. This can provide the execution trace of a particular transaction from the entire system. If we want to analyze, how the entire system is working and responding in different execution traces we can do that for the component and connector style as it includes all the functionalities of the system. For example, consider a peer-to-peer architecture for distributed data storage for a company. This can provide a execution trace of complete systems processes.

3.2.1 Client-Server view:

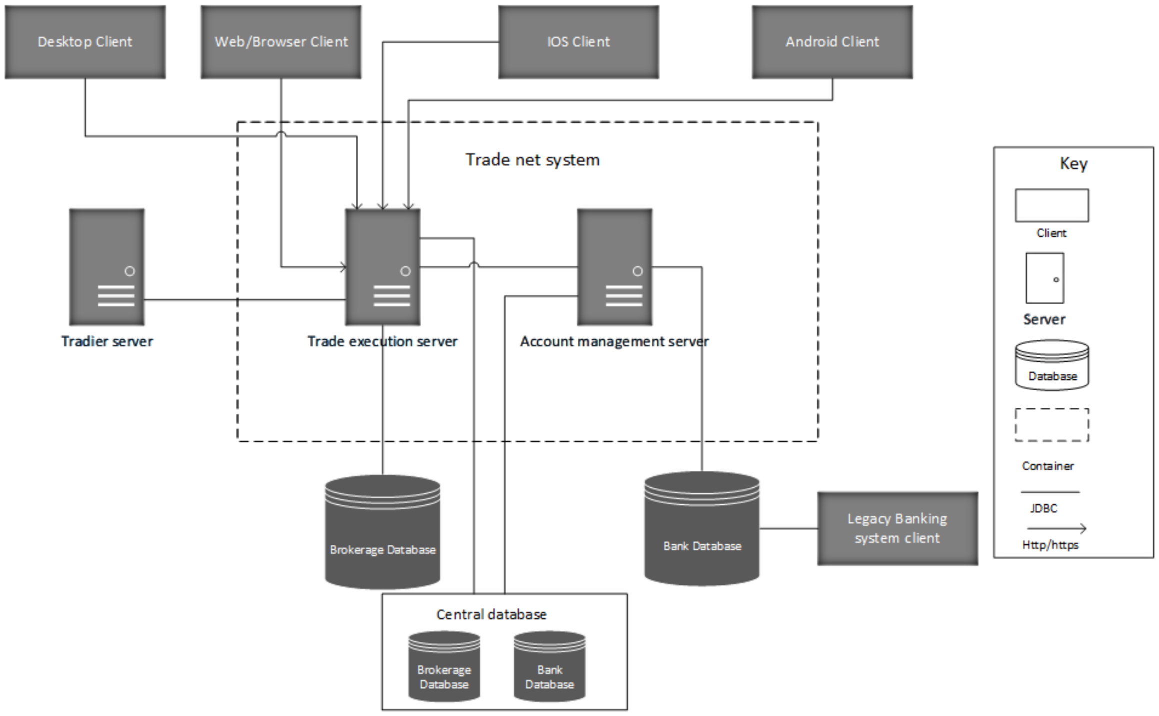

Figure 6: client-server view

Description:

The components in the system are Web/Browser, Desktop, iOS, Android, Trade Execution Server, Account Management Server, Banking Server, Central Database, Banking Database, Legacy Banking system. Through Web/Browser, iOS, Android clients we can access/request the services of the TNBS and it contains the Servers like Trade execution server and account management server through which these requests can be addressed. The requests from the servers are redirected to the respective databases which are used for the data retrieval. The Tradier API collects the real-time market data for the users to buy and sell the stocks.

Element Catalog :

Module	Type	Description/Function
Web/Browser	Client	Helps in accessing the brokerage system via browser.
Desktop	Client	Helps in accessing the brokerage system via desktop.
iOS	Client	Helps in accessing the brokerage system via iOS device.
Android	Client	Helps in accessing the brokerage system via Android device.
Trade Execution Server	Server	Helps in retrieving real-time market data
Account Management Server	Server	Helps in checking the availability of the funds.
Banking Server	Server	Helps in performing banking transactions.
Central Database	Database	Stores the data of the TE server and AM server.
Banking Database	Database	Stores the data of the banking server.
Legacy Banking system	Client	Helps in accessing the banking system via command line interface.
Brokerage Database	Database	Stores the brokerage transaction details of respective users.

<u>Process & their modules :</u>

Processes	Modules
Accessing the TNBS.	Web/Browser Client, Desktop Client, iOS Client, Android Client.
Accessing the Banking System.	Command Line Banking System, Account Management Server, Banking Server, Banking Database.
Checking the real-time market values.	Clients, Tradier API, Trade Execution server.
Buying and Selling of shares.	Clients, Tradier API, Trade Execution server, Account Management Server,Brokerage Database.
Authenticating the user.	Clients, Trade Execution server, Account management server.
Checking the availability of funds.	Account Management Server, Banking Database.

Accessing the banking system will communicate with the buying and selling of shares process to to exchange the details about the sellers and buyers bank account details involves the and it also communicates with the authentication process for valid user authentication.

Accessing the TNBS is the process which communicates with the user and the TNBS. When user log-in it checks the validity through the authentication process.

If the user is valid it allows the user to buy sell shares and after the selecting user shares to buy/sell it checks with the real time market data for the value of their shares and gives the results to the trade execution server.

Banking system exchanges the information about the details of the customer with the other processes like buy/ sell shares.

Check the availability of the funds process gives the information about the details of the funds available in the bank account to the buy /Sell shares and access the banking system processes.

<u>Rationale:</u>

This view can address the concerns of the quality assurance analyst. Through this client server style quality assurance analyst ensures the requests from the clients efficiently and ensures that all the requests are served without any server down time. It also ensures that there is no transaction failure which can enhance the trust of the customers, which in turn helps to assure the quality.

3.2.2 Service Oriented architectural view:

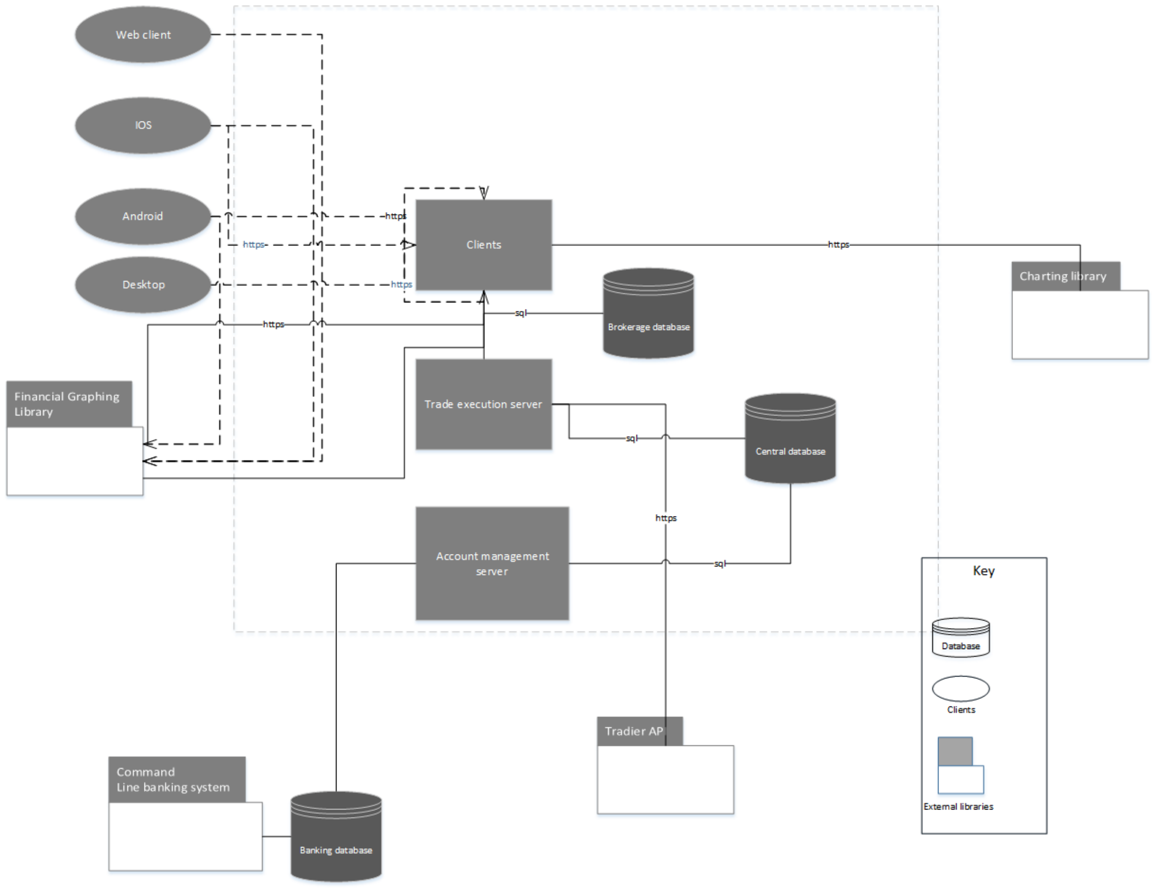

Figure 7: Service oriented architectural style

Description:

The components in the system are mobile client, web clients,Trade Execution Server, Account Management Server, Banking Server, Central Database, Banking Database, Legacy Banking system. The clients access the TNBS through the web browser and mobile applications. These clients uses the external service providers for graphs and charts.These clients internally connected to the trade execution server which is used to perform the buy/sell shares operation.Trade execution server is connected to account management server for the authentication of users before the transaction. It also uses the external service provider Tradier API for accessing the real time market data.It connects to the central database for storing all the transactions and logs. It also connected to brokerage database to store all the transactional data.Account management server is also the part of the internal system.It uses to authenticate the user and manages all the banking activities. It uses the banking database to store all the customer bank details. Central database used to store all the customer bank details.

__Element Catalog :__

Module	Type	Description/Function
Web/Browser	Client	Helps in accessing the brokerage system via browser.
Desktop	Client	Helps in accessing the brokerage system via desktop.
iOS	Client	Helps in accessing the brokerage system via iOS device.
Android	Client	Helps in accessing the brokerage system via Android device.
Trade Execution	Server	Helps in retrieving real-time market data
Account Management Server	Server	Helps in checking the availability of the funds.
Banking Server	Server	Helps in performing banking transactions.
Central Database	Database	Stores the data of the TE server and AM server.
Banking Database	Database	Stores the data of the

		banking server.
Tradier	Server	Provides real-time market data.
Financial Graphing	Services(3rd party)	Provides financial graphing services for the users.
Charting	Services(3rd party)	Provides charting services for the users.
Brokerage Database	Database	Stores the brokerage transaction details of respective users.

<u>Process & their modules :</u>

Processes	Modules
Accessing the TNBS.	Web/Browser Client, Desktop Client, iOS Client, Android Client.
Accessing the Banking System.	Command Line Banking System, Account Management Server, Banking Server, Banking Database.
Checking the real-time market values.	Clients, Tradier API, Trade Execution server.
Buying and Selling of shares.	Clients, Tradier API, Trade Execution server, Account Management Server, Brokerage Database.
Authenticating the user.	Clients, Trade Execution server, Account Management Server.
Checking the availability of funds.	Account Management Server, Banking Database.
Charting Service	Clients, Trade Execution server, 3rd party Service Provider.
Financial Graphing Service	Clients, Trade Execution server, 3rd party Service Provider.

Accessing the banking system will communicate with the buying and selling of shares process to to exchange the details about the sellers and buyers bank account details involves the and it also communicates with the authentication process for valid user authentication.

Accessing the TNBS is the process which communicates with the user and the TNBS. When user log-in it checks the validity through the authentication process.

If the user is valid it allows the user to buy sell shares and after the selecting user shares to buy/sell it checks with the real time market data for the value of their shares and gives the results to the trade execution server.

Banking system exchanges the information about the details of the customer with the other processes like buy/ sell shares.

Check the availability of the funds process gives the information about the details of the funds available in the bank account to the buy /Sell shares and access the banking system processes.

Charting service takes the data from the clients and use the charting libraries for preparing charts and the communicate or send the charts to the clients for displaying them.

Graphing service takes the data from the clients and use the graphing libraries for creating graphs and the communicate or send the graphs to the clients for displaying them.

Rationale:

This view can better address the concerns of the developer. Through this he can know the which external services are required and which internal components are making use of them which can be used to check the compatibility and also the he can know the system internal and external components through which he can develop the code based on the internal dependencies of the system.

3.2.3 Pipe & Filter :

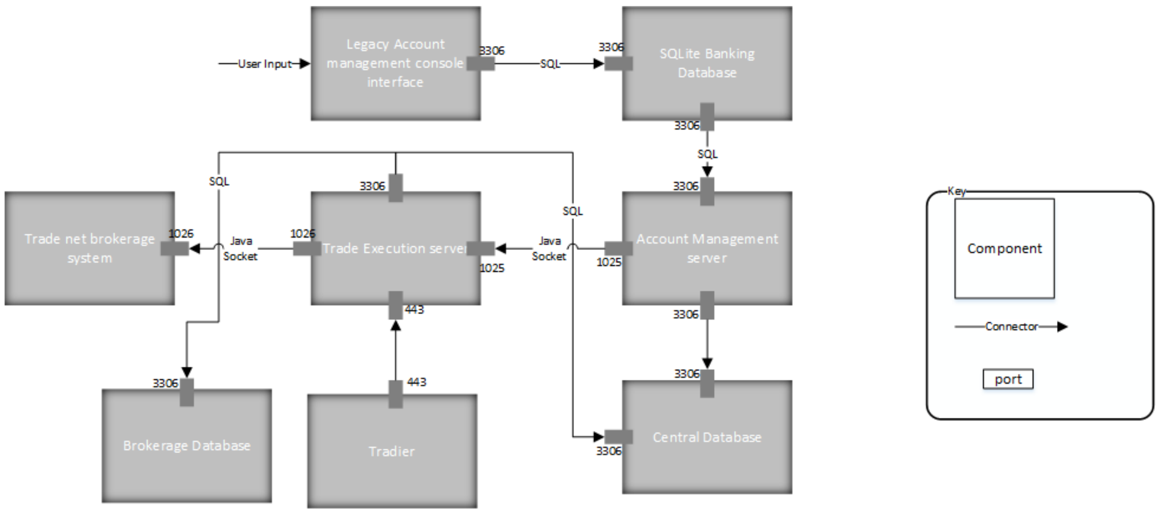

Figure 8: Pipe & filter style

Description:

The components in the system are Trade Execution Server, Account Management Server, Banking Server, Central Database, Banking Database, Legacy Banking system. The clients can access the legacy banking system for the banking transactions.It connects to the SQLite database connection port for bank transactions. It has user defined ports which facilitate in communication between the components when an user accesses the system. It mainly highlights the flow of information across the system. The client initiates the access to the Trade Net System through the clients service providers. Clients authenticate the users suing the authentication protocols depending on the type of the information queried.Customers can initiate the request to access the TNBS through the desktops and mobile apps. All the requests from the clients are send to the web server. If the customer requests the buy/sell the trades the request is redirected to the trade execution server. When the customer selects the the shares to buy or sell it checks the real time market data with the Tradier API. Checking of the funds which are sufficient for the sale is then checked and response from this is sent to the customer. If the customer requests are to check the bank balance of their accounts the requests are redirected to the account management server. Then the account management server checks the details of the customer through the banking database.Command line banking system also has the command line user interface through which customers can check the balance and transfer the amount to the other accounts.

Element Catalog :

Element	Description/Function
TNBS	Interface for the TNBS.
Trade Execution	Helps in retrieving real-time market data.
Account Management Server	Helps in checking the availability of the funds.
Legacy Banking System	Helps in accessing the banking system via command line interface.
Central Database	Stores the data of the TE server and AM server.
Banking Database	Stores the data of the banking server.
Brokerage Database	Stores the brokerage transaction details of respective users.

Tradier API	Provides real-time market data.

Process & their modules :

Processes	Modules
Accessing the TNBS.	TNBS Clients.
Accessing the Banking System.	Command Line Banking System, Account Management Server, Banking Server, Banking Database.
Checking the real-time market values.	Clients, Tradier API, Trade Execution server.
Buying and Selling of shares.	Clients, Tradier API, Trade Execution server, Account Management Server, Brokerage Database.
Authenticating the user.	Clients, Account Management Server, Trade Execution server.
Checking the availability of funds.	Account Management Server, Banking Database.

Accessing the banking system will communicate with the buying and selling of shares process to to exchange the details about the sellers and buyers bank account details involves the and it also communicates with the authentication process for valid user authentication.

Accessing the TNBS.is the process which communicates with the user and the TNBS. When user log-in it checks the validity through the authentication process.

If the user is valid it allows the user to buy sell shares and after the selecting user shares to buy/sell it checks with the real time market data for the value of their shares and gives the results to the trade execution server.

Banking system exchanges the information about the details of the customer with the other processes like buy/ sell shares.

Check the availability of the funds process gives the information about the details of the funds available in the bank account to the buy /Sell shares and access the banking system processes.

Rationale :

This view can better address the concerns of the Information Technology Analyst. He is mainly concerned with the flow of the information from one component to another component, the type of components and connectors used, and their compatibility with other components as well. This view is used by IT analyst to verify the performance of the system with help of flow of information between the components.

3. Allocation Views :

3.3.1 Overview:

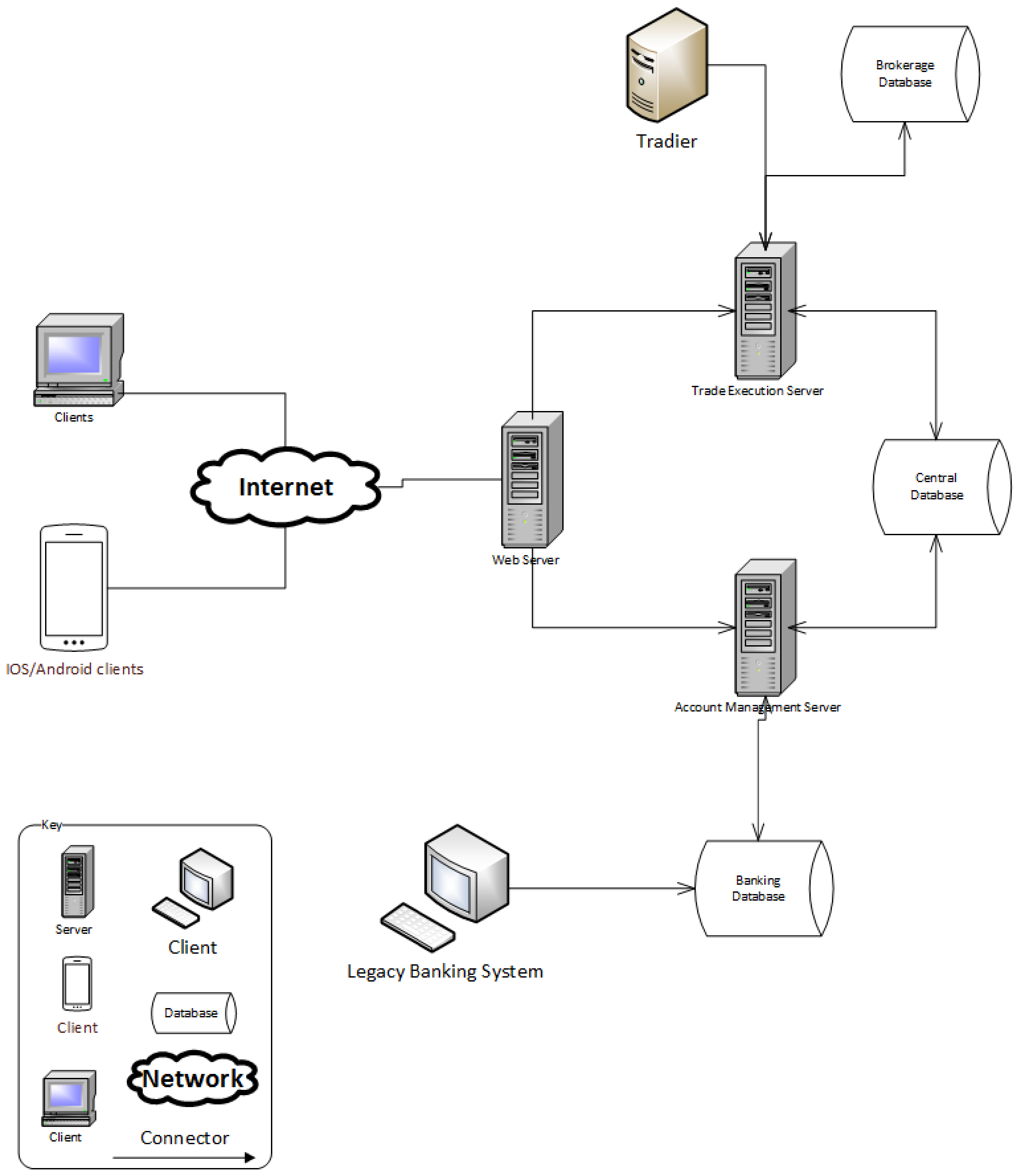

Figure 9: Deployment Style Overview

<u>Description:</u>

Customers can initiate the request to access the TNBS through the desktops and mobile apps. All the requests from the clients are send to the web server. Then the web server accepts all the requests and redirects the requests to the respective servers. if the customer requests the buy/sell the trades the request is redirected to the trade execution server. When the customer selects the the shares to buy or sell it checks the real time market data with the tradier api. Checking of the funds which are sufficient for the sale is then checked and response from this is sent to the customer. If the customer requests are to check the bank balance of their accounts the requests are redirected to the account management server. Then the account management server checks the details of the customer through the banking database.Command line banking system also has the command line user interface through which customers can check the balance and transfer the amount to the other accounts.

3.3.2 Deployment View:

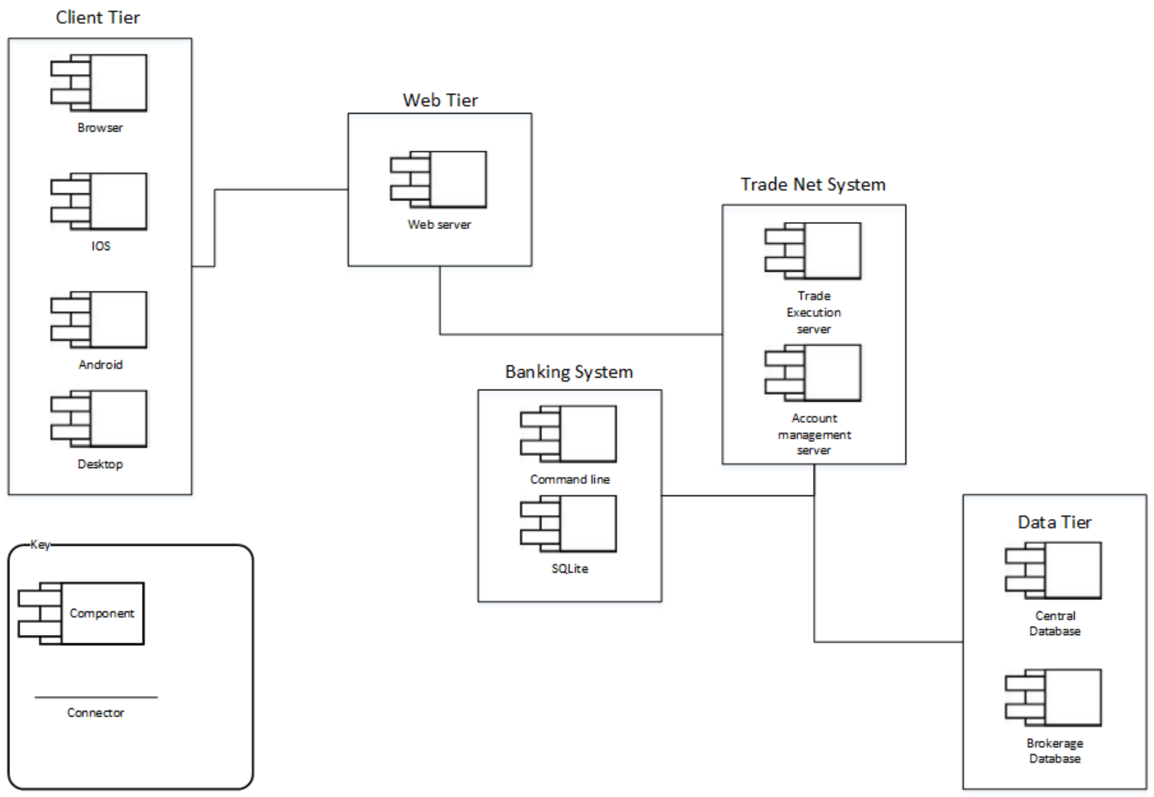

Figure 10 : Deployment style

Description:

In this deployment view TNBS is described as tired architecture.There are client tier, web tier, Trade Net system, Banking system, Brokerage database and Data tier. In these client tier consists of the clients through which we can access the system. Desktop and web browser can be used to access the system from workstation. iOS and Android clients can be used to the access the system from the mobile applications. Client tier is connected to the web tier for requesting the services of the TNBS and web tier redirects these request to the Trade net system. This system consists of the trade execution server and account management server which performs the main operations of TNBS. Trade Net system is connected to the data tier for storing all the transactions performed by the trade execution server and account management server. Trade execution server is connected to the brokerage system for logging all the transactions and account management server is connected to the banking system which contains the command line interface and SQLite database for serving the requests from client.

<u>Element Catalog :</u>

Element	Description/Function
Web/Browser	Helps in accessing the brokerage system via browser.
Desktop	Helps in accessing the brokerage system via desktop.
iOS	Helps in accessing the brokerage system via iOS device.
Android	Helps in accessing the brokerage system via Android device.
Trade Execution	Helps in retrieving real-time market data
Account Management Server	Helps in checking the availability of the funds.
Banking Server	Helps in performing banking transactions.
Central Database	Stores the data of the TE server and AM server.
Banking Database(SQLite)	Stores the data of the banking server.
Brokerage Database	Stores the brokerage transaction details of respective users.
Web Server	Connects the Trade Net requests from the client to Trade Net server.

<u>Hardware components:</u>

Web Server

Trade execution server

Account management server

Desktop

Mobiles

<u>Location:</u>

Web server, Account management server, Trade execution server located at the remote location and locally which is made available through the ports and protocols.If one of the servers fails other server serves the purpose of the users.

Desktops can be of personal or shared desktops where personal computers are located in their residences and public computers can be in the labs,stock exchanges offices and brokerage firms.

Mobile is portable device which can be used to access the TNBS with a proper Internet connection.

<u>Communication capabilities:</u>

Web server can accept the requests from the clients and redirect the requests to the trade execution server. When the trade execution server responds to the requests it will send the response to the respective clients.

The Trade execution server receives the requests of the clients and it performs buy/sell operation on the shares and communicates its response to the clients through the web server.It also access the customer details from the account management server.

Account management server contains the details of all customers. Trade execution server requests are served by the account management server and it communicates with the banking server for authentication of all customers.

Desktop can send the requests to the servers and it receives the responses which is displayed using xml code.

Mobile can send the requests through the applications of the iOS and Android and the response is displayed through the xml code.

Processes of hardware items:

Web server are associated with the processes like the accessing the TNBS and sending the responses to the clients.

Trade net execution server is associated with the process like buy/sell shares, checking the details of the customers and checking the real time data with the Tradier API.

Account management server is associated with the process like checking the funds and customer details of their accounts.

Mobile and Desktop can be used to access the TNBS and display the charts and graphs.

Rationale:

This view can address the concerns of the project manager. It helps him to assign the work by knowing the dependencies among all the components and he can analyze on the critical tasks in making a successful deployment of the project.

4. Consistency among Architectural Views (Mapping Between Views)

4.1 Consistency among Client server and Deployment Style:

Step 1: Ensure that all components have been allocated to hardware

- Desktop Client : This component from Client server style is allocated to the hardware in the client tier.
- Web/Browser client : This component from client server style is allocated to the hardware in the client tier.
- iOS Client : This component from client server style is allocated to the hardware in the client tier.
- Android client : This component from client server style is allocated to the hardware in the client tier.
- Trade execution server : This component from client server style is allocated to the hardware in the core tier.
- Account management server: This component from client server style is allocated to the hardware in the core tier.
- Brokerage database: This component from client server style is allocated to the hardware in the database tier.
- Banking database : This component from client server style is allocated to the hardware in the banking system tier.
- Central database: This component from client server style is allocated to the hardware in the database tier.
- Legacy banking system client: This component from client server style is allocated to the hardware in the banking system tier.

Step 2: Verify the accuracy of the mapping of each component

- Desktop client : This component in client server style serves a purpose for its users to access the TNBS from the desktop and the client tier in the deployment style resides all the hardware related to the client side. This verifies the Desktop client component is precisely positioned in the client tier.
- Web Client : This component in client server style serves a purpose for its users to access the TNBS from the desktop or mobile through the browser and the client tier in the deployment style resides all the hardware related to the client side. This verifies the web client component is precisely positioned in the client tier.
- IOS Client : This component in client server style serves a purpose for its users to access the TNBS from the mobile through an iOS application and the client tier in the deployment style resides all the hardware related to the client side. This verifies the iOS client component is precisely positioned in the client tier.
- Andoid Client : This component in client server style serves a purpose for its users to access the TNBS from the mobile through an Android application and the client tier in the deployment style resides all the hardware related to the client side. This verifies the Android client component is precisely positioned in the client tier.
- Trade execution server: This component in client server style serves the major functionality of TNBS to perform stock transactions by users. Core tier in the deployment style contains the environmental elements which performs the major functionality. This verifies that component is correctly mapped to the core tier in the deployment style.

- Account management server : This component in client server style serves the major functionality of TNBS to authenticate the users. Core tier in the deployment style contains the environmental elements which performs the major functionality. This verifies that component is correctly mapped to the core tier in the deployment style.
- Brokerage database: This component in the client server style stores all the transactions performed by the trade execution server and database tier in deployment style resides all the software elements related to the database. This verifies that component is correctly mapped to the database tier in the deployment style.
- Central database: This component in the client server style stores all the transactions and logs performed by the trade execution server and database tier in deployment style resides all the software elements related to the database. This verifies that component is correctly mapped to the database tier in the deployment style.
- Banking database: This component in the client server style stores all the customer information of the TNBS and database tier in deployment style resides all the software elements related to the database. This verifies that component is correctly mapped to the database tier in the deployment style.
- Legacy banking system client: This component in the client server style allows the customer to perform bank transactions through command line and this requires a separate hardware with the interface. This justifies the positioning of this component in the banking system tier.

4.2 Consistency among Service Oriented architecture and Deployment Style:

Step 1: Ensure that all components have been allocated to hardware

- Desktop Client : This component from Service Oriented architecture is allocated to the hardware in the client tier.
- Web/Browser client : This component from Service Oriented architecture is allocated to the hardware in the client tier.
- iOS Client : This component from Service Oriented architecture is allocated to the hardware in the client tier.
- Android client : This component from Service Oriented architecture is allocated to the hardware in the client tier.
- Trade execution server : This component from Service Oriented architecture is allocated to the hardware in the core tier.
- Account management server: This component from Service Oriented architecture is allocated to the hardware in the core tier.
- Brokerage database: This component from Service Oriented architecture is allocated to the hardware in the database tier.
- Banking database : This component from Service Oriented architecture is allocated to the hardware in the banking system tier.
- Central database: This component from Service Oriented architecture is allocated to the hardware in the database tier.
- Legacy banking system client: This component from Service Oriented architecture is allocated to the hardware in the banking system tier.

- Graphing/Charting library: This is a COTS component which resides in the Core tier to support the trade execution server.

Step 2: Verify the accuracy of the mapping of each component

- Desktop client : This component in client server style serves a purpose for its users to access the TNBS from the desktop and the client tier in the deployment style resides all the hardware related to the client side. This verifies the Desktop client component is precisely positioned in the client tier.
- Web Client : This component in client server style serves a purpose for its users to access the TNBS from the desktop or mobile through the browser and the client tier in the deployment style resides all the hardware related to the client side. This verifies the web client component is precisely positioned in the client tier.
- iOS Client : This component in client server style serves a purpose for its users to access the TNBS from the mobile through an iOS application and the client tier in the deployment style resides all the hardware related to the client side. This verifies the iOS client component is precisely positioned in the client tier.
- Android Client : This component in client server style serves a purpose for its users to access the TNBS from the mobile through an Android application and the client tier in the deployment style resides all the hardware related to the client side. This verifies the Android client component is precisely positioned in the client tier.

- Trade execution server: This component in client server style serves the major functionality of TNBS to perform stock transactions by users. Core tier in the deployment style contains the environmental elements which performs the major functionality. This verifies that component is correctly mapped to the core tier in the deployment style.
- Account management server : This component in client server style serves the major functionality of TNBS to authenticate the users. Core tier in the deployment style contains the environmental elements which performs the major functionality. This verifies that component is correctly mapped to the core tier in the deployment style.
- Brokerage database: This component in the client server style stores all the transactions performed by the trade execution server and database tier in deployment style resides all the software elements related to the database. This verifies that component is correctly mapped to the database tier in the deployment style.
- Central database: This component in the client server style stores all the transactions and logs performed by the trade execution server and database tier in deployment style resides all the software elements related to the database. This verifies that component is correctly mapped to the database tier in the deployment style.
- Banking database: This component in the client server style stores all the customer information of the TNBS and database tier in deployment style resides

all the software elements related to the database. This verifies that component is correctly mapped to the database tier in the deployment style.

- Legacy banking system client: This component in the client server style allows the customer to perform bank transactions through command line and this requires a separate hardware with the interface. This justifies the positioning of this component in the banking system tier.
- Graphing/Charting library: These are COTS component which can be purchased on demand to support the trade execution server and the core tier consists the functionalities which support the core functionalities. This justifies the placement of Graphing/Charting library in the core tier if at all it is purchased and embedded in our system.

4.3 Consistency among Pipe & Filter style and Deployment Style:

Step 1: Ensure all components are allocated to hardware

- Legacy account management console interface: This component from Pipe & Filter style is allocated to the hardware in the banking system tier.
- SQLite banking database: This component from Pipe & Filter style is allocated to the hardware in the banking system tier.
- Account management server: This component from Pipe & Filter style is allocated to the hardware in the core tier.
- Trade execution server: This component from Pipe & Filter style is allocated to the hardware in the core tier.

- TNBS: This component from Pipe & Filter style is partitioned based on the user and core functionality. User functionality of the TNBS resides in the client tier and core functionality of TNBS resides in the core tier.
- Tradier API: This component from Pipe & Filter style is allocated to the hardware in the core tier which is an on demand component works in collaboration with the trade execution server
- Central database: This component from Pipe & Filter style is allocated to the hardware in the database tier.
- Brokerage database:This component from Pipe & Filter style is allocated to the hardware in the database tier.

Step 2: Verify the accuracy of the mapping of each component

- Legacy account management console interface: This component in the Pipe & Filter style is is a command line interface through which clients can perform the bank transactions and in deployment style it is rightly placed in the client server to serve its purpose.
- SQLite Banking database: This database stores all the transactions performed by the customer with the TNBS through the command line interface. Banking system tier is a place where all the elements related to the banking functionality resides in and the as this component id one of the element resides in banking system tier to provide the required functionality.
- Account management server: This component in Pipe & Filter style serves the major functionality of TNBS to authenticate the users. Core tier in the deployment

style contains the environmental elements which performs the major functionality. This verifies that component is correctly mapped to the core tier in the deployment style.

- Trade execution server: This component in Pipe & Filter style serves the major functionality of TNBS to perform stock transactions by users. Core tier in the deployment style contains the environmental elements which performs the major functionality. This verifies that component is correctly mapped to the core tier in the deployment style.
- TNBS: This component is the main subject of the brokerage system. This should be functionally split to provide the functionality for user access and perform transactions. Thus, user access functionality resides in the client tier and core functionality for transactions resides in the core tier.
- Tradier API: This component is used to get the real time market data from Dow Jones and it works in association with the trade execution server. This is aptly placed in the core tier to support trade execution server.
- Central database: This component in the Pipe & Filter style stores all the transactions and logs performed by the trade execution server and database tier in deployment style resides all the software elements related to the database. This verifies that component is correctly mapped to the database tier in the deployment style.
- Brokerage system: This component in the pipe & filter style stores all the transactions performed by the trade execution server and database tier in deployment style resides all the software elements related to the database. This

verifies that component is correctly mapped to the database tier in the deployment style.

4.4 Consistency among uses style and Deployment Style:

Step 1: Compare module views to deployment views

- Client apps: This module in the uses style is intended to provide a way to access the TNBS. This module is intact with the Client tier of the deployment style serves the same purpose of user interaction with the TNBS.
- Tradier API: This is the module in uses style intended for providing the real time market data from Dow Jones and its absence in the deployment style can be attributed to on demand nature of the module which can be purchased from an external provider and integrated with the deployed software.
- Third party services: This module in the uses style is intended for the usage of the Trade execution server for graphing and charting. It is not impaired in the deployment style as it is present in the core tier.
- Trade execution server: This module is used by the clients to transact with the TNBS and it is implemented in the core tier of the deployment style.
- Account management server: This module in the uses style is used by the trade execution server for authentication of the users and it is allocated to the core tier in the deployment style.
- Banking system: This module serves the user to perform bank transactions through the command line interface and it is implemented in deployment style without any modification to its functional properties.

- Banking database: This database is used to store all the customer details of legacy banking system. This is implemented of deployment style without any change.
- Central database: This database is used to store the transactions and logs of the trade execution server and account management server. This is deployed in the database tier of deployment style.
- Brokerage database: This database is used to store the transactions of the trade execution server. This is deployed in the database tier of the deployment style.

Step 2: Ensure that organization of the modules in the deployment view makes sense.

- Account management server: This module in the uses style is used by the trade execution server for authentication of the users and it is correctly allocated to the core tier in the deployment style to be used by the trade execution server.
- Banking system: This module serves the user to perform bank transactions through the command line interface and it is placed in the banking system tier of deployment without any modification to its functional properties.
- Banking database: This database is used to store all the customer details of legacy banking system. This is placed in deployment style for serving the legacy banking system.
- Central database: This database is used to store the transactions and logs of the trade execution server and account management server. This is placed in the database tier to be used by trade execution and account management server.
- Brokerage database: This database is used to store the transactions of the trade execution server. This is placed in the database tier to be used by trade execution

server.Client apps: This module in the uses style is intended to provide a way to access the TNBS. This module is rightly placed in Client tier of the deployment style to serve the same purpose of user interaction with the TNBS.

- Tradier API: This is the module in uses style intended for providing the real time market data from Dow Jones and its absence in the deployment style can be attributed to on demand nature of the module which can be purchased from an external provider and integrated with the deployed software.
- Third party services: This module in the uses style is intended for the usage of the Trade execution server for graphing and charting. It is aptly placed in the deployment style as it is present in the core tier to be used by the Trade execution server.
- Trade execution server: This module is used by the clients to transact with the TNBS and it is implemented in the core tier of the deployment style without altering its functional significance.

4.5 Consistency among Layered view and deployment view

Step 1: Compare module views to deployment views

- Third party services: This is intended for providing the graphing and charting services and it is implemented in the core tier of the deployment style.
- Trade execution server: This module is used by the clients to transact with the TNBS and it is implemented in the core tier of the deployment style.

- Account management server: This module in the uses style is used by the trade execution server for authentication of the users and it is allocated to the core tier in the deployment style.
- Banking database: This database is used to store all the customer details of legacy banking system. This is implemented of deployment style without any change.
- Central database: This database is used to store the transactions and logs of the trade execution server and account management server. This is deployed in the database tier of deployment style.
- Brokerage database: This database is used to store the transactions of the trade execution server. This is deployed in the database tier of the deployment style.

Step 2: Ensure that organization of the modules in the deployment view makes sense.

- Third party services: This module in the uses style is intended for the usage of the Trade execution server for graphing and charting. It is aptly placed in the deployment style as it is present in the core tier to be used by the Trade execution server.
- Trade execution server: This module is used by the clients to transact with the TNBS and it is implemented in the core tier of the deployment style without altering its functional significance.
- Account management server: This module in the uses style is used by the trade execution server for authentication of the users and it is correctly allocated to the core tier in the deployment style to be used by the trade execution server.

- Banking system: This module serves the user to perform bank transactions through the command line interface and it is placed in the banking system tier of deployment without any modification to its functional properties.
- Banking database: This database is used to store all the customer details of legacy banking system. This is placed in deployment style for serving the legacy banking system.
- Central database: This database is used to store the transactions and logs of the trade execution server and account management server. This is placed in the database tier to be used by trade execution and account management server.
- Brokerage database: This database is used to store the transactions of the trade execution server. This is placed in the database tier to be used by trade execution server.
- The functions and the protocols which exist in the the layered view are not part of the deployment view as these are internal to the system.

4.6 Consistency among Layered view and client server style

Step 1: Find the mapping between the modules and the components

There are different client modules in the Layered view which can be mapped to the client components in the client server style.

- Web client module can be mapped to the web client component in the client server style.

- Desktop module can be mapped to the Desktop component in the client server style.
- IOS application client module can be mapped to the iOS application client component in the client server style.
- Android application client module can be mapped to the Android application client component in the client server style.
- Command line banking system module can be mapped to the command line banking system client component in the client server style.

Servers used to perform the transactions and authenticate the users in TNBS

- Trade execution server : This server module in can be mapped to the trade execution server component in the client server style.
- Account management server: This server module in can be mapped to the account management server component in the client server style.

Databases used to store the transactions,logs and customer details.

- Brokerage database: This database can be mapped to the brokerage database component in the client & server style.
- Banking database: This database can be mapped to the banking database component in the client & server style.
- Central database: This database can be mapped to the central database component in the client & server style.

Step 2: Understand the mapping

- Web browser client module is mapped to the web browser client component as both are used to access the TNBS through a browser from mobile or desktop.
- Desktop client module is mapped to the Desktop client component as both are used to access the TNBS from the desktop.
- iOS client module is mapped to the iOS client component as both are used to access the TNBS from a mobile application.
- Android client module is mapped to the Android client component as both are used to access the TNBS from a mobile application.
- command line banking system client module is mapped to the command line banking system client component as both are accessed by the customers to perform the bank transactions from command line interface.
- Trade execution server module is mapped to the trade execution server component as both carry out the major functionality of performing transactions through TNBS.
- Account management server module is mapped to the account management server component as both support the trade execution server in authenticating user.
- Brokerage database in the layered view is mapped to the brokerage database in the client server as both are used by the trade execution server to store the transactions.
- Banking database in the layered view is mapped to banking database in the client server style as both are used to contain the customer details.

- Central database in the layered view is mapped to the central database in the client server style as both are used to contain the transaction and logs of Trade Net Brokerage database.

Step 3: Determine if the mapping make sense

- Mapping between the Web Browser client in Layered view and client server style is meaningful as they can serve the users in the similar way and no other component in both views provides the same functionality.
- Mapping between the Desktop client in Layered view and client server style is meaningful as they can serve the users through the destop and no other component in both views provides the same functionality.
- Mapping between the Android application client in Layered view and client server style is meaningful as they can serve the users to access the TNBS and no other component in both views provides the same functionality.
- Mapping between the iOS application client in Layered view and client server style is meaningful as they can serve the users to access the TNBS from mobile in the similar way and no other component provides the same functionality.
- Mapping between the command line banking in Layered view and client server style is meaningful as they can serve the users to access the command line banking system from interface in the similar way and no other component provides the same functionality.

- Trade execution server in the Layered view and client server style are mapped as these provide the core functionality of the TNBS and important components in there respective views.
- Trade execution server in the Layered view and client server style are mapped as these provide the core functionality of the TNBS and important components in there respective views.
- Account management server in the Layered view and client server style are mapped as the functionality is very similar and it serves the trade execution server in both views.
- Brokerage database in the layered view and client server style are mapped as they bot serve the same purpose for trade execution server to store the transactions of trade execution server.
- Banking database in the layered view and client server style are mapped as they are used to store the customer details in both the view.
- Central Database in the layered view and client server style are mapped as they provide the same functionality to the TNBS.

Step 3.1 Look for cases where there is a relationship between two components such that one relies on the other for operation …

- The Layered module view is functionally divided into different layers and top layer depends functionally on the layers below it and vice versa is not possible.
- Client modules in the top most layer relies on the functionalities in the second layer to access the trade execution server for transactions and account

management server for authentication in the next layer and it uses the banking database to store the customer details, central database for storing transaction and logs and Brokerage database to store all the transactions.

<u>4.7 Consistency among Layered view and Pipe & Filter style</u>

Step 1: Find the mapping between the modules and the components

- Web client module, Desktop client, iOS application client, Android client can be mapped to the TNBS filter component in the.pipe & filter style.
- Command Line banking system : This module can be mapped to the Legacy account management system in pipe & filter style.
- Trade execution server : This server module in can be mapped to the trade execution server component in the pipe & filter style.
- Account management server: This server module in can be mapped to the account management server component in the pipe & filter style.
- Database used to store the transactions,logs and customer details.
- Brokerage database: This database can be mapped to the brokerage database component in the pipe & filter style.
- Banking database: This database can be mapped to the banking database component in the pipe & filter style.
- Central database: This database can be mapped to the central database component in the pipe & filter style.

Step 2: Understand the mapping

- Web browser client, Desktop client, iOS application client and Android application client modules is mapped to the TNBS component as they act as gateways to access the TNBS..
- command line banking system client module is mapped to the command line banking system client component as both are accessed by the customers to perform the bank transactions from command line interface.
- Trade execution server module is mapped to the trade execution server component as both carry out the major functionality of performing transactions through TNBS.
- Account management server module is mapped to the account management server component as both support the trade execution server in authenticating user.
- Brokerage database in the layered view is mapped to the brokerage database in the pipe & filter style as both are used by the trade execution server to store the transactions.
- Banking database in the layered view is mapped to banking database in the pipe & filter style as both are used to contain the customer details.
- Central database in the layered view is mapped to the central database in the pipe & filter style as both are used to contain the transaction and logs of Trade Net Brokerage database.

Step 3: Determine if the mapping make sense

- Mapping between the Web Browser client, Desktop client, Android application client, iOS application client in Layered view and to the Trade Net Brokerage filter component is meaningful as they can serve the users in the similar way to access the TNBS and no other component in both views provides the same functionality.

- Trade execution server in the Layered view and pip & filter style are mapped as these provide the core functionality of the TNBS and important components in there respective views.
- Account management server in the Layered view and pipe & filter style are mapped as the functionality is very similar and it serves the trade execution server in both views.
- Brokerage database in the layered view and pipe & filter style are mapped as they both serve the same purpose for trade execution server to store the transactions of trade execution server.
- Banking database in the layered view and pipe & filter style are mapped as they are used to store the customer details in both the view.
- Central Database in the layered view and pipe & filter style are mapped as they provide the same functionality to the TNBS.

4.8 Consistency among Layered style and service oriented architecture

Step 1: Find the mapping between the modules and the components

Clients to access the TNBS:

- There are different client modules in the Layered view which can be mapped to the client components in the service oriented architecture.

- Web client module, Desktop client module, Android application client module and iOS application client module can be mapped to the Clients component of the service oriented style.
- Command line banking system module can be mapped to the command line banking system client component in the service oriented style.
- Servers used to perform the transactions and authenticate the users in TNBS.
- Trade execution server : This server module in can be mapped to the trade execution server component in the service oriented architecture.
- Account management server: This server module in can be mapped to the account management server component in the service oriented architecture.
- Database used to store the transactions,logs and customer details.
- Brokerage database: This database can be mapped to the brokerage database component in the service oriented architecture.
- Banking database: This database can be mapped to the banking database component in the service oriented architecture.
- Central database: This database can be mapped to the central database component in the service oriented architecture.
- Graphing/Charting libraries can be mapped to the third party services in service oriented architecture.

Step 2: Understand the mapping

- Web browser client, Desktop client module, Android application module and iOS application module are mapped clients component as they are used to access the TNBS.
- command line banking system client module is mapped to the command line banking system client component as both are accessed by the customers to perform the bank transactions from command line interface.
- Trade execution server module is mapped to the trade execution server component as both carry out the major functionality performing transactions of the TNBS.
- Account management server module is mapped to the account management server component as both support the trade execution server in authenticating user.
- Brokerage database in the layered view is mapped to the brokerage database in the service oriented architecture as both are used by the trade execution server to store the transactions.
- Banking database in the layered view is mapped to banking database in the service oriented architecture as both are used to contain the customer details.
- Central database in the layered view is mapped to the central database in the service oriented as both are used to contain the transaction and logs of Trade Net Brokerage database.
- Graphing/charting library can be mapped to third party services as they both assist the TNBS to the trade execution server.

Step 3: Determine if the mapping make sense

- Mapping between the Web browser client, Desktop client module, Android application module and iOS application module in Layered view and Clients in service oriented style is meaningful as they can serve the users in the similar way and no other component in both views provides the same functionality.
- Mapping between the command line banking in Layered view and service oriented architecture is meaningful as they can serve the users to access the command line banking system from interface in the similar way and no other component provides the same functionality.
- Trade execution server in the Layered view and service oriented architecture are mapped as these provide the core functionality of the TNBS and important components in there respective views.
- Account management server in the Layered view and service oriented architecture are mapped as the functionality is very similar and it serves the trade execution server in both views.
- Brokerage database in the layered view and service oriented architecture are mapped as they both serve the same purpose for trade execution server to store the transactions of trade execution server.
- Banking database in the layered view and service oriented architecture are mapped as they are used to store the customer details in both the view.
- Central Database in the layered view and service oriented architecture are mapped as they provide the same functionality to the TNBS.

- Graphing/ Charting library and third party services as they both provide graphing & charting service externally on demand.

Step 3.1 Look for cases where there is a relationship between two components such that one relies on the other for operation ...

- The Layered module view is functionally divided into different layers and top layer depends functionally on the layers below it and vice versa is not possible.
- Client modules in the top most layer relies on the functionalities in the second layer to access the trade execution server for transactions and account management server for authentication in the next layer and it uses the banking database to store the customer details, central database for storing transaction and logs and Brokerage database to store all the transactions.

4..9 Consistency among uses style and Client server style

Step 1: Find the mapping between the modules and the components

- Clients to access the TNBS:
- There are different client modules in the uses style which can be mapped to the web browser client, Desktop client, Android application client, iOS application client to the client server style.
- Command line banking system module can be mapped to the command line banking system client component in the client server style.
- Servers used to perform the transactions and authenticate the users in TNBS

- Trade execution server : This server module in can be mapped to the trade execution server component in the client server style.
- Account management server: This server module in can be mapped to the account management server component in the client server style.
- Database used to store the transactions,logs and customer details.
- Brokerage database: This database can be mapped to the brokerage database component in the client & server style.
- Banking database: This database can be mapped to the banking database component in the client & server style.
- Central database: This database can be mapped to the central database component in the client & server style.

Step 2: Understand the mapping

- Client apps can be mapped to the Desktop client, Android application client, iOS application client client component as they are used to access the TNBS.
- command line banking system client module is mapped to the command line banking system client component as both are accessed by the customers to perform the bank transactions from command line interface.
- Trade execution server module is mapped to the trade execution server component as both carry out the major functionality of performing transactions through TNBS.
- Account management server module is mapped to the account management server component as both support the trade execution server in authenticating user.

- Brokerage database in the uses view is mapped to the brokerage database in the client server as both are used by the trade execution server to store the transactions.
- Banking database in the uses view is mapped to banking database in the client server style as both are used to contain the customer details.
- Central database in the uses view is mapped to the central database in the client server style as both are used to contain the transaction and logs of Trade Net Brokerage database.

Step 3: Determine if the mapping make sense

- Mapping between the client apps in uses style and Desktop client, Android application client, iOS application client in client server style is meaningful as they can serve the users in the similar way and no other component in both views provides the same functionality.
- Mapping between the command line banking in uses style and client server style is meaningful as they can serve the users to access the command line banking system from interface in the similar way and no other component provides the same functionality.
- Trade execution server in the uses style and client server style are mapped as these provide the core functionality of the TNBS and important components in there respective views.
- Account management server in the uses style and client server style are mapped as the functionality is very similar and it serves the trade execution server in both views.

- Brokerage database in the uses style and client server style are mapped as they bot serve the same purpose for trade execution server to store the transactions of trade execution server.
- Banking database in the uses style and client server style are mapped as they are used to store the customer details in both the view.
- Central Database in the uses style and client server style are mapped as they provide the same functionality to the TNBS.

Step 3.1 Look for cases where there is a relationship between two components such that one relies on the other for operation …

- The Layered module view is functionally divided into different layers and top layer depends functionally on the layers below it and vice versa is not possible.
- Client modules in the top most layer relies on the functionalities in the second layer to access the trade execution server for transactions and account management server for authentication in the next layer and it uses the banking database to store the customer details, central database for storing transaction and logs and Brokerage database to store all the transactions.

4.10 Consistency among uses style and pipe & filter style:

Step 1: Find the mapping between the modules and the components

Clients to access the TNBS:

- There are different client modules in the uses style which can be mapped to the TNBS client in the pipe & filter style.

- Command line banking system module can be mapped to the command line banking system client component in the pipe & filter style.
- Servers used to perform the transactions and authenticate the users in TNBS
- Trade execution server : This server module in can be mapped to the trade execution server component in the pipe & filter style.
- Account management server: This server module in can be mapped to the account management server component in the pipe & filter style.
- Database used to store the transactions,logs and customer details.
- Brokerage database: This database can be mapped to the brokerage database component in the pipe & filter style.
- Banking database: This database can be mapped to the banking database component in the pipe & filter style.
- Central database: This database can be mapped to the central database component in the pipe & filter style.

Step 2: Understand the mapping

- Client apps can be mapped to the TNBS client component as they are used to access the TNBS as they are used to access the system.
- command line banking system client module is mapped to the command line banking system client component as both are accessed by the customers to perform the bank transactions from command line interface.

- Trade execution server module is mapped to the trade execution server component as both carry out the major functionality of performing transactions through TNBS.
- Account management server module is mapped to the account management server component as both support the trade execution server in authenticating user.
- Brokerage database in the uses view is mapped to the brokerage database in the pipe & filter as both are used by the trade execution server to store the transactions.
- Banking database in the uses view is mapped to banking database in the pipe & filter style as both are used to contain the customer details.
- Central database in the uses view is mapped to the central database in the pipe & filter style as both are used to contain the transaction and logs of Trade Net Brokerage database.
- *Step 3: Determine if the mapping make sense*
- Mapping between the client apps in uses style and TNBS client in pipe & filter style is meaningful as they can serve the users in the similar way and no other components in both views provides the same functionality.
- Mapping between the command line banking in uses style and pipe & filter style is meaningful as they can serve the users to access the command line banking system from interface in the similar way and no other component provides the same functionality.

- Trade execution server in the uses style and pipe & filter style are mapped as these provide the core functionality of the TNBS and important components in there respective views.
- Account management server in the uses style and pipe & filter style are mapped as the functionality is very similar and it serves the trade execution server in both views.
- Brokerage database in the uses style and pipe & filter style are mapped as they both serve the same purpose for trade execution server to store the transactions of trade execution server.
- Banking database in the uses style and pipe & filter style are mapped as they are used to store the customer details in both the view.
- Central Database in the uses style and pipe & filter style are mapped as they provide the same functionality to the TNBS.

4.11 Consistency among uses style & service oriented architecture:

Step 1: Find the mapping between the modules and the components

Clients to access the TNBS:

- There are different client modules in the uses style which can be mapped to the client apps in the Service oriented architecture style.
- Command line banking system module can be mapped to the command line banking system client component in the Service oriented architecture style.
- Servers used to perform the transactions and authenticate the users in TNBS.

- Trade execution server : This server module in can be mapped to the trade execution server component in the Service oriented architecture style.
- Account management server: This server module in can be mapped to the account management server component in the Service oriented architecture style.
- Database used to store the transactions,logs and customer details.
- Brokerage database: This database can be mapped to the brokerage database component in the Service oriented architecture style..
- Banking database: This database can be mapped to the banking database component in the Service oriented architecture style.
- Central database: This database can be mapped to the central database component in the Service oriented architecture style.

Step 2: Understand the mapping

- Client apps in uses can be mapped to the client apps in Service oriented architecture style as they are used to access the TNBS.
- command line banking system client module is mapped to the command line banking system client component as both are accessed by the customers to perform the bank transactions from command line interface.
- Trade execution server module is mapped to the trade execution server component as both carry out the major functionality of performing transactions through TNBS.
- Account management server module is mapped to the account management server component as both support the trade execution server in authenticating user.

- Brokerage database in the uses view is mapped to the brokerage database in the Service oriented architecture as both are used by the trade execution server to store the transactions.
- Banking database in the uses view is mapped to banking database in the Service oriented architecture style as both are used to contain the customer details.
- Central database in the uses view is mapped to the central database in the Service oriented architecture style as both are used to contain the transaction and logs of Trade Net Brokerage database.

Step 3: Determine if the mapping make sense

- Mapping between the client apps in uses style and client apps in Service oriented architecture style is meaningful as they can serve the users in the similar way and no other component in both views provides the same functionality.
- Mapping between the command line banking in uses style and Service oriented architecture style is meaningful as they can serve the users to access the command line banking system from interface in the similar way and no other component provides the same functionality.
- Trade execution server in the uses style and Service oriented architecture style are mapped as these provide the core functionality of the TNBS and important components in there respective views.
- Account management server in the uses style and Service oriented architecture style are mapped as the functionality is very similar and it serves the trade execution server in both views.

- Brokerage database in the uses style and Service oriented architecture style are mapped as they bot serve the same purpose for trade execution server to store the transactions of trade execution server.
- Banking database in the uses style and Service oriented architecture style are mapped as they are used to store the customer details in both the view.
- Central Database in the uses style and Service oriented architecture style are mapped as they provide the same functionality to the TNBS.

4.12 Consistency among Data model view and client & server style

Step 1: Find the mapping between the modules and the components

- Data model view is presents the database internally with the tables and relationships between the tables. This view consists of tables in all databases in the client & server style.
- We have person, Account, Admin, Customer, Bank_transaction, Brokerage system table which will be mapped to the brokerage system, Central database and banking database.

Step 2: Understand the mapping

- All the tables in the data model view should be present in the databases of the client server style. So data model view is mapped to the databases of client server style.

Step 3: Determine if the mapping make sense

- Mapping of the data model view to database make sense as the database tables can only reside in the databases and no other component supports these tables in client & server style.

4.13 Consistency among Data model view and Pipe & filter style

Step 1: Find the mapping between the modules and the components

- Data model view is presents the database internally with the tables and relationships between the tables. This view consists of tables in all databases in the pipe & filter style.
- We have person, Account, Admin, Customer, Bank_transaction, Brokerage system table which will be mapped to the brokerage system, Central database and banking database.

Step 2: Understand the mapping

All the tables in the data model view should be present in the databases of the client server style. So data model view is mapped to the databases of pipe & filter style.

Step 3: Determine if the mapping make sense

- Mapping of the data model view to database make sense as the database tables can only reside in the databases and no other component supports these tables in pipe & filter style.

4.14 Consistency among Data model view and Service oriented architecture

Step 1: Find the mapping between the modules and the components

- Data model view is presents the database internally with the tables and relationships between the tables. This view consists of tables in all databases in the service oriented architecture.
- We have person, Account, Admin, Customer, Bank_transaction, Brokerage system table which will be mapped to the brokerage system, Central database and banking database.

Step 2: Understand the mapping

- All the tables in the data model view should be present in the databases of the client server style. So data model view is mapped to the databases of service oriented architecture.

Step 3: Determine if the mapping make sense

- Mapping of the data model view to database make sense as the database tables can only reside in the databases and no other component supports these tables in service oriented architecture.

4.15 Consistency among Data model view and deployment style

Step 1: Compare the Module views to the deployment view

- Comparison between the data model view and deployment view cannot be possible as the former is internal to the system and latter represents the external hardware but we can make sure that each table in the data model view has a database to reside in.

Step 2: Ensure that the organization of the modules in the deployment view makes sense.

This makes sense as the database tables in the data model view can only reside in databases present in the deployment view.

5.Architectural Rationale

5.1 Layered View

In our Trade Net Layered View architecture, there are 5 layers. The first consists of various platforms how the clients can access the system. These can be through command line, web browser, desktop, IPhone (iOS app) or Android platform. These platforms give the clients a visual way interacting with the system. The second layer has all the functions that the clients can perform using the system. These can be purchasing or selling of the shares, checking funds, withdrawing, depositing, creating accounts, etc. The third layer consists of various protocols such as SSL, HTTPS, and SWIFT which allows the system to connect with external libraries such as graphing, charting, etc. The fourth layer consists of trade execution server, account management server, and banking server. These servers have a vital role in the execution of buying and selling of the shares, authenticating the users and saving all the transaction information. The last layer contains the central, banking and brokerage databases where all the information of the users and transactions are stored.

There are multiple reason of choosing layered view architecture. Layered view makes it flexible and easy to add more layers to the TNBS if we decide to do that in the future. It also helps us with information hiding. It prevents the users to view and make changes to the logic of the system while trading (buying and selling) the stocks. It also allows for efficient communication between the layers. Another great benefit of the layered view architecture is that it allows multiple applications to reuse the components. For example, in our TNBS, if we use web browser, Android, iOS, or desktop application, we can easily use the same business logic, and only a new user interface will be required. Also, when it comes to development, it reduces the dependencies and allows various teams to work on different parts. Lastly, if we want to test our Trade Net system, we can easily test the various components separately using the layered view architecture.

5.2 Uses Style View

The Uses Style view is great for addressing the concerns of the maintenance team. In the TNBS, various components such as servers, apps, api, databases are interconnected and use information from each other. Here, using the Trade Execution Server, clients perform operations such as buying and selling of the shares. They also display graphs and charts of the stocks using third party graphing and charting libraries. Also, the system uses an API called Tradier for real-time market data and account management, brokerage and central database for authenticating users, processing transactions and saving the users and transaction logs.

For a successful system to be developed, these functionalities need to be implemented in an incremental way. Uses Style Architecture can be used here as it allows an incremental development. Also, as various databases, servers and clients are implemented in subsets, it allows for an efficient debugging and testing of the system. If any change is made, this style allows to measure the magnitude of the change and manage it effectively. In the TNBS, there will be lots of times that maintenance such as adaptive, preventative or perfective will be required. This style will allow us to manage dependencies during maintenance. These can be controlling the complexity, avoiding the degradation of our system, and even ensuring that we can easily modify new components.

5.3 Run-time and Dynamic View

The Component and Connector (C&C) View shows the system as it appears at the run-time. They help in guiding development by specifying the structure and behavior of run-time elements. They show how the system works and assist the architects reason about run-time system quality attributes. In the TNBS, it shows how various components and run-time entities are connected and interacting. These consist of clients such as web, desktop, Android, iOS interacting with the brokerage, account management, and banking databases through the trade execution and account management servers.

This view shows all the pathways of interactions and complex communication protocols such as https, ssl, ftp. These protocols help us provide security features such as encryption, audit trails, and authentication. The Component and Connector View also helps with the performance of the Trade Net System by allowing us to measure its response time, latencies, and throughput. This view also addresses the reliability factor and the likelihood of failure. It helps with the resource requirements and the pressing storage needs that can be met through the databases. Lastly, in the Trade Net System, it allows us to run separate processes as a thread for better execution of the system.

5.4 Client Server View

In the Client Server Style, the various servers such as Trade Execution, Account Management and Banking Server provide information to the web, iOS, Android, and desktop client upon requests. Here, we decoupled various client applications from the Trade Execution Server for the security aspect of the system. The Client Server View of the TNBS addresses the concerns of the Quality Assurance Analyst. It allows them to visually see that all the client- server requests are successfully processing.

It also allows for the stability of the system. Any element can be upgraded when needed. Also, when new security measures need to be taken and new technology needs to be integrated, this view allows for flexibility and changes can easily be made by updating the server. Also, as the various components of the client server style are loosely coupled, it allows for easy testing of the system.

5.5 Service Oriented Architectural View

The service oriented view can better address the concerns of the developer. This view highlights the external services and core system services needed from other components in the system to carry out their functionalities as per the requirement of the user. This can help the programmer in choosing the compatible COTS products for the core systems. Our TNBS uses the services of Tradier API to get the real time market value data for the customers which aid the customers in buying and selling of shares. The Tradier API has to work in conjunction with the Trade Excution server. The programmer can decide whether compatibility between these two components and can find a way to integrate them. Our clients make use of the 3rd party services such as Graphing Services and

Financial Services to the users of the system. The programmer can decide whether compatibility between these two components and can find a way to integrate them. The Trade Execution server make use of the Account Management server which in turn make use of the Legacy banking system. The programmer can decide whether compatibility between these two components and can find a way to integrate them. So, the Service Oriented Style can better address the concern of the developer than any other component and connector views.

5.6 Pipe and Filter View

The pipe and filter view can address the concerns of the Information Technology Analyst than any other component and connector style. The reason is that this style shows all the components such as legacy account management console, trade execution and account management servers, SQLite banking, central and brokerage databases and Tradier third party application and allows the flow of data between these components in an efficient manner. It ensures quality attributes such as information hiding, high cohesion, modifiability, and reuse.

5.7. Deployment view

Deployment view can be used by people who have a broader perspective about the functionality of the project like manager. This provides details about the performance, reliability and security standards implemented in the project. By seeing the deployment view we can estimate the cost of the project. The goal of the deployment view is to allocate each software element with the hardware element and organization of the hardware elements based on the functional dependencies. This view is suitable for Trade Net Brokerage system. In this,there is a trade execution server and account management server which performs the core functionality of the system and each other component is functionally dependent on these two servers. This can provide the reliability to the trade net brokerage system and the total cost of project can be estimated as the cost to develop and deploy these two servers.

By developing the deployment overview we can easily identify the physical properties of each component like server should be placed in the centralized location and clients should provide an easy way to access by proving multiple applications and