



Hochschule RheinMain
Fachbereich Design Informatik Medien
Studiengang Informatik

Master Thesis

Submitted in Partial Fulfillment of the requirements for the Degree of
Master of Science

WORKFLOW AUTOMATION FOR EFFICIENT ON-DEMAND DISTRIBUTION OF SENSITIVE DATA IN THE FINANCE SECTOR

Vorgelegt von Rahul Thiruthinmel Premnavas
am 17. December 2024
Referent Prof. Dr. Nikolay Tcheltchov

Declaration in accordance with ABPO

I hereby declare that

- have completed this thesis independently,
- used no sources other than those specified,
- the passages taken verbatim or in terms of content from other works passages, pictorial representations and the like have been precisely identified as such and
- did not make use of any unauthorized outside help.

Wiesbaden, 17. December 2024

Rahul Thiruthinmel Premnavas

Explanation on the use of AI tools

- Which tools were used ?
- Which areas of the text / software created were created / revised with the help of AI tools and how? (pure revision, creation, research, summarisation, translation,?)
- Are you fully responsible for the transfer of any machine-generated text passages?

Wiesbaden, 17. December 2024

Rahul Thiruthinmel Premnavas

Declaration on the use of the Masterthesis

I hereby declare my consent to the following forms of distribution of this thesis:

Form of Dissemination	Yes	No
Placement of the work in the university library with the data carrier		×
Placement of the work in the university library without data carrier	×	
Publication of the title of the work on the Internet	×	
Publication of the work on the Internet		×

Wiesbaden, 17. December 2024

Rahul Thiruthinmel Premnavas

Abstract

Timely Data Distribution in the financial sector is very crucial, because of the changing business environments and the amount of competence in the ever-changing area. In this scenario, automation of tasks can significantly reduce the amount of time one requires to deliver on-demand data. The Automation of the File-Transfer-System is carried out in this thesis, emphasising on optimising everyday transactions and enhancing service delivery. FTS (File Transfer System), a vital component in an industry where various file transfers occur on a day-to-day basis across several time-zones, is the primary focus of this research.

Another important component of this research is the internal platform for provisioning requests. A significant number of tasks that get accumulated in Request for Product Development (RPD), the above-mentioned internal platform for provisioning requests or tasks, are done manually, which obviously would take more time given the benefit of the doubt to the proposed idea of automating the frequent repetitive tasks. The tasks to be done can be of various types, which span throughout the various departments in a firm, from account creation in a particular application (business units) to changing the protocols, etc.

This research at FTS (File Transfer System) aims to enhance the customer experience management thereby making the daily client to the organisation and vice versa file transfers in the File Transfer System at Factset fluent. Principally there are five tasks to solve.

1. SFTP(Secure File Transfer Protocol) Key-generation
2. Password reset
3. FTS/SFTP protocol change
4. Management of Client IP address restrictions (add/modify/delete)
5. Creation of EH (Enterprise Hosts) accounts

Valid study and research regarding each service's RESTAPI's have been carried out, Python programming is used to implement the logic and the code being managed in the prominent version control system git/GitHub. Logging methods are implemented with the help of a Python module named logging, steps are also carried out to parse the arguments from the command line interface using the Python module Argparse. As each task mentioned above uses different services, API architecture had to be carefully examined and understood before proceeding from one task to another.

The first task, SFTP Key-Generation is completed at the beginning and it yielded results, the code repositories are forwarded to the production servers. Further explanation and details regarding the tasks will be reported in the Implementation part of this report.

Contents

1	Introduction and Motivation	2
1.0.1	The OSI Model	2
1.0.2	Other methods in FTS	7
1.0.3	Motivation	8
2	Problem Statement	11
2.1	Requirements	13
3	State of the the Art	14
3.1	Related Articles	14
3.2	Standards	21
4	Structure of the Thesis	24
4.1	Requirements	24
4.2	Architecture	24
4.3	Implementation	24
5	Experiment and Discussion	25

Chapter 1

Introduction and Motivation

For an organization, it is paramount that the processes and undertaking are expected to reach the customers faster and more efficiently. It is also a vital factor in the firm's development, determined by how fast the services and products are delivered. In any financial firm the failure of service deliveries can lead to financial loss monetarily and subsequently in the market value, many financial software firms have immediate access to financial data and analytics thereby making sure that the choices they make are relevant and quick. In Information Technology FTS stands for File Transfer System and it uses the FTP (File Transfer Protocol) to transfer files between networks and hosts, although outdated, it still has its usage and implementation in places. The SFTP (Secure File Transfer Protocol) is the secure way of transferring files, which many companies are still using, to achieve a secure line of connection between FTS Servers and the client hosts. The detailed description of the architecture will be discussed in the later parts of the work. Ports 21 and 22 are used for both protocol.

Anyhow it is also worth mentioning and diving deep into the networking aspects of this area of study. As we have already seen, the File Transfer System is a vital component in the firm, and FTP is the respective protocol, what is its architecture? and all the information that comes along with it is also worth delving into.

1.0.1 The OSI Model

Firstly we will have a look at the OSI (Open System Interconnection Model) Model. According to [10] OSI Model, whose conception began in the 1970s, is the outcome of efforts of ISO (International Organization for Standardization) and made the layer-to-layer architecture of the communication channels, which in turn made it easier to categorize and comprehend rather a complex circuit of layers. The main objective of the OSI model is to make an interoperable network circuit that would

help the developers and vendors [10]. Layering is the underlying idea of the OSI model and there are seven layers included to describe the OSI model.

1. Physical Layer
2. Data Link Layer
3. Network Layer
4. Transport Layer
5. Session Layer
6. Presentation Layer
7. Application Layer

Physical Layer

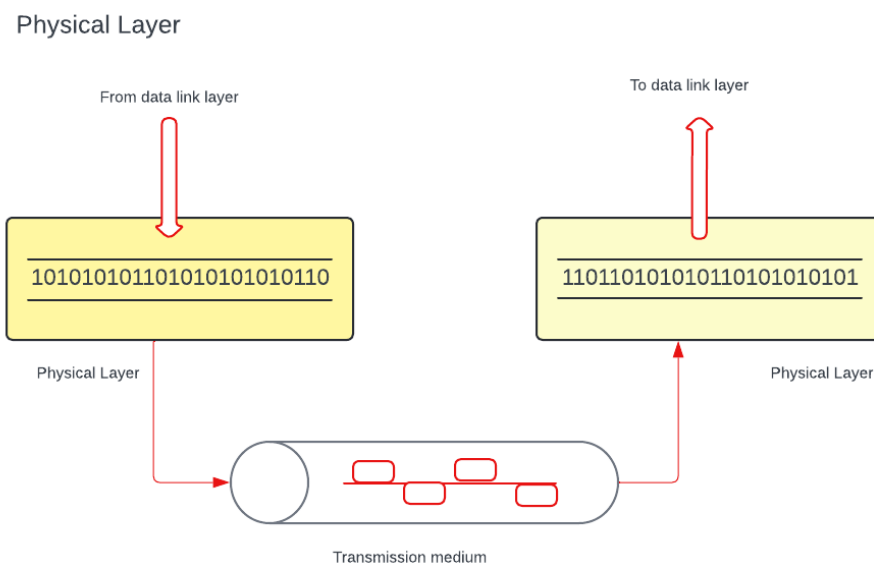


Figure 1.1: The physical layer [10]

It is the lowest layer of the OSI model, figure 1.1 depicts the physical layer and determines the medium of physical transmission and all the other physical components while making two nodes connect, for example, cables, pins, voltages, etc. While establishing connections between nodes, it also controls the flow and its modulation[10].

Data Link Layer

The main function of the Data Link Layer is to check the error using the Cyclic Redundancy Check (CRC) [10], shown in figure 1.2. When the nodes are not directly connected, the checkers will be done through different links which operate differently, the job of sending packets to MAC addresses of hosts is done by this layer, the common protocols at the Data Link layer are Ethernet, HDLC (High-Level Data Link).

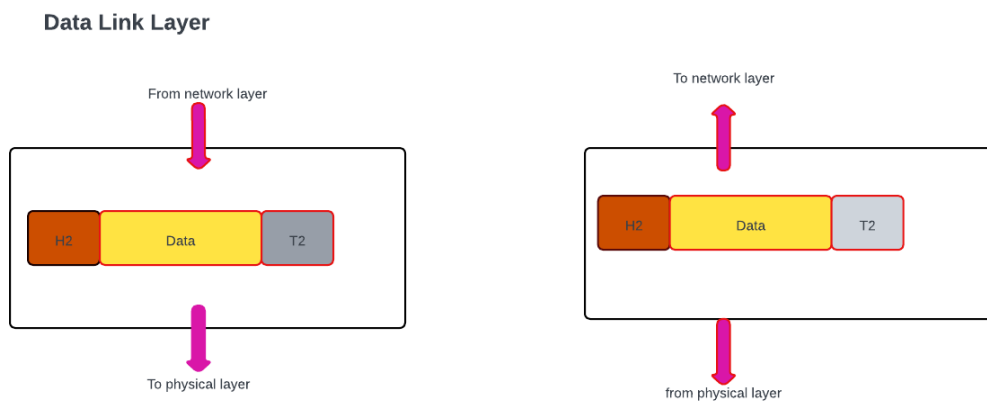


Figure 1.2: The data link layer [10]

Network Layer

The main function of this layer is to make a path and define the address for the nodes. In a network, every node is connected via addresses and this is done by routing, by placing addresses of senders and receivers on headers and this is done in this layer [10]. The figure 1.3 below shows how the data is carried from the transport layer to the data link layer. The network may (or may not) implement message delivery in addition to message routing. This could involve breaking the message up into multiple pieces, sending each fragment over a different route, and then piecing the pieces back together, reporting delivery failures, etc.

Network Layer

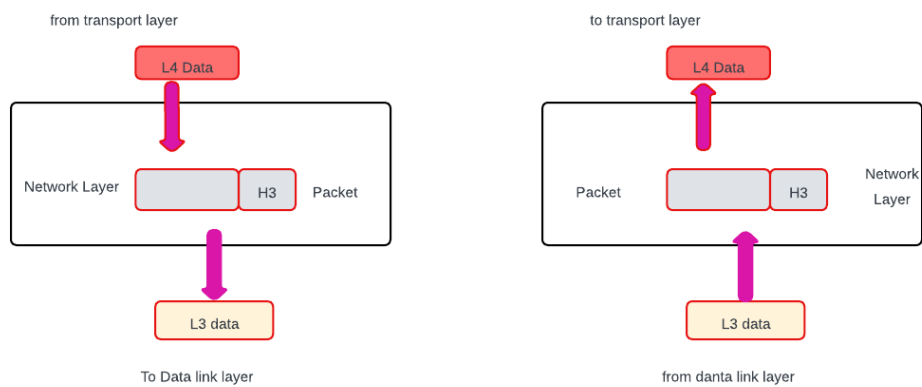


Figure 1.3: The network layer [10]

Transport Layer

Transport Layer

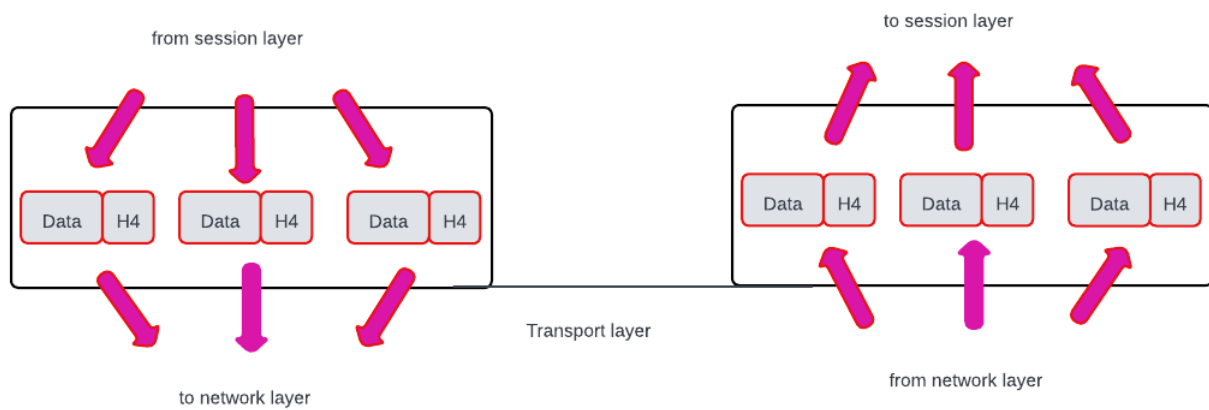


Figure 1.4: The transport layer [6]

It provides an end-to-end connection to users on different nodes. Provides reliability, and security like encryption thereby ensuring complete data transfer[6]. Figure 1.4 shows how data is being carried out in a transport layer.

Session Layer



Figure 1.5: The session layer [10]

As its name suggests it holds a session between two users/nodes, the figure 1.5, the whole setting-up of the session and data exchanges thereby monitoring the session and the security aspect as well [10].

Presentation Layer



Figure 1.6: The presentation layer [6]

It, figure 1.6, can be seen as the translator in a network[10]. It formats the data received and transfers to the application layer which is readable for humans and vice versa. In this layer, the following jobs are also done.

- Character code translation
- Data Conversion
- Data compression
- encryption

Application Layer

The user interaction happens at this layer, figure 1.7. The familiar protocols running at this layer are HTTPS, HTTP, FTP, NFS, SMTP, and SSH.

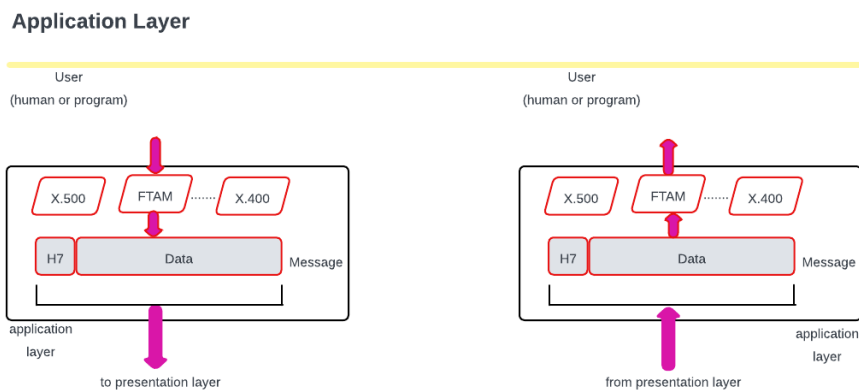


Figure 1.7: The application layer [6]

1.0.2 Other methods in FTS

In this area of work, the focus on FTP is in the foreground. File Transfer Protocol finds its usage in the application layer of the OSI model and it is still widely used by many enterprises to transfer data between clients and servers. The major advantage of using FTP is that it uses no intermediary to carry out the data transfer, like we otherwise use browsers or emails to deliver data, in most cases an application (FTP Client Applications) dedicated to the File Transfer System is used to do the file transfer [14].

This paragraph discusses the various methods that are implemented in FTS. As per [11] they have proposed a C2CFTP an efficient and indirect file transfer between a client and server structure, because of the drawbacks they have pointed out in this paper, and the major drawback is said to be indirect file transfer which makes a lot

of Input Output of files which are done by the server, which delay time is increases and thus the efficiency compromises. The C2CFTP proposes the idea of not storing the file in indirect transfers, in this case, the file need not be opened and relayed to the subscriber.

In this article from [17] they have introduced a covert file transfer protocol wherein they try to covert the existence of communication, differing from encryption, where the details of communication are encrypted. This is deployed in sending Short messages and text files. CFTP is used unlike FTP ICMP protocol the IP record route option to transfer data and the IP record route option is made some changes to get the desired result.

In the master thesis provided by [3] , as the solution to bulk amounts of data being shared and exchanged, they have introduced a Multi File Transfer Protocol(MFTP), which solves the issue of time delay in downloading resources, whereby they disseminate the files to be shared and then regather it, and found successful in sharing and downloading the same data even large files without any delay.

1.0.3 Motivation

As the amount of data and transfers in a Firm increases it is quite common that the employees especially the developers and IT engineers revert to the idea of automation, as the clients increase the File transfer system is getting a lot of incident enhancement and other requests daily, this is thereby the undertaking of the FTS engineers to minimize the effort to respond to the requests from the clients, aiming at compatibility and effectiveness. Five tasks need to be addressed, log data need to be created for effective monitoring and the front end part to be taken care of as well, which might not be included in this part of the work.

The figure explains how daily transactions are being done at FTS manually, it has three main components, the Request for Product Development, first-level engineers also the TSE team, who take up requests and process them with their permissible rights, then the FTS engineers who have additional rights on AdminFTS host who can perform more tasks with the admin rights. The RPD requests can be everything from creating an account in different business units to changing the protocols and changing login restriction policies.

RPD figures which would be allocated to a particular group in this case to the first level of support or the FTS curators, take up the tasks make necessary adjustments in the applications about the tasks and sometimes on the hosts, finish it, and resolve the RPD by commenting. The RPD can be of different severity and can be raised according to how the client wants to get the problem solved.

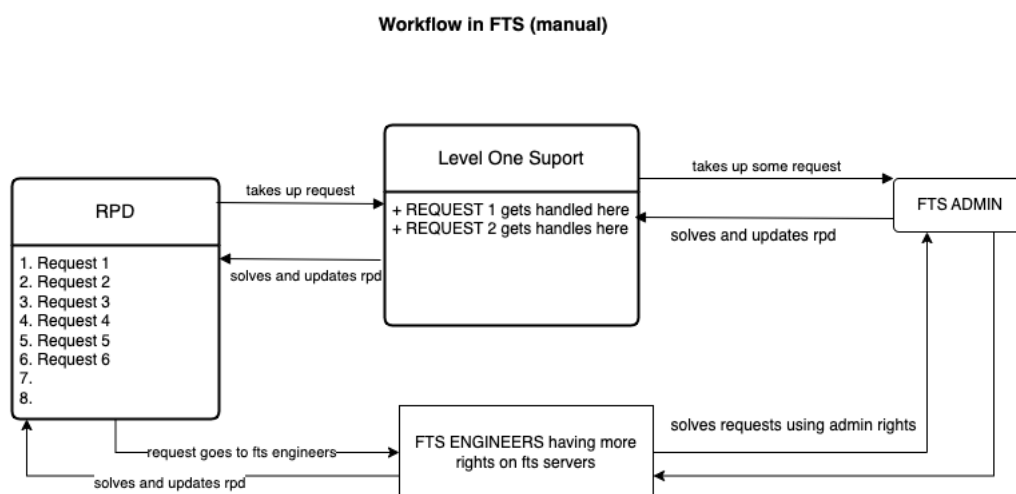


Figure 1.8: The manual process at Factset

The FTSAdmin server is the host with the host names, which can be invoked using CLI to perform.

The tasks of the TSE team and FTS engineers are given briefly below.

Technology Solutions Engineering(First layer)- is focused on the stability, performance, and functionality of the firms products, and supports its product platforms at companies client sites. Its 120,000+ workstations, 10,000+ web instances, mobile devices, and batch processes are supported 24/7, in close cooperation with the consulting support desk and operations staff. Client Technology Oversight, a subgroup of Technology Solutions Engineering - focuses on difficult (tier 3 / 4) technology issues and adapting products to be turnkey and seamless so that the firm minimizes the technology burden on its clients. Single sign-on, virtualization, packaging, performance, and stability are driven by this team aligned with Engineering and Product Development. This team is also the escalation point for new technologies so that firms can be agile and adapt elegantly to changing technology conditions.

Strategic Client Technology, a subgroup of TSE - ensures that top clients by ASV are receiving personalized, directed, proactive assistance for rollouts and upgrades. This group also works to ensure our clients have a single point-of-contact in our engineering group for road map discussions as well as connectivity, performance, and stability conversations.

Desktop Solutions, a subgroup of TSE - focuses on automation, macros, and tailored configurations which allow Factset's clients to build efficient workflows.

Again, the TSE group would be handed over most of the scripts so that they could

cater to the client's needs. The automation tasks that we are attempting through this research would be production-ready and the TSE team will run the scripts (python, shell scripts) on their CLI that how the processes go, apart from that they also do troubleshooting techniques before escalating the request to FTS engineers. The troubleshooting steps can be given below to get an overview of what is happening under the curtain.

1. File an RPD -as soon as they get a request from the client
2. Verify if the client has a valid subscription
3. Investigate the issue
4. Verify if the client can make a connection
5. Adding additional notes - proxies, file lock permissions, etc
6. Gather verbose logs

The FTS engineers have additional rights on admin ftp and their duties can be listed below.

- Manage scalable high availability File Transfer System (FTS) for use by Standard data feeds, Cornerstone, Enterprise Hosting, and Portfolio Batch services
- Installation, upgrades, and system configuration
- Internal tools for automation of client and vendor account management workflows
- FTS Monitoring and health check system
- Support of the FTS environment
- Migration to cloud-hosted infrastructure

Another important aspect of the research is the Ticketing System for project management is also essential for better communication between management people and the customers, the major among is Jira, in our work we use RPD (Request for Product Management) as a ticket management system and the API of its been used to fetch details such as user names, host names, IP addresses, etc.

Surely, somewhere, somehow, in the history of computing, at least one manual has been written that you could at least remotely attempt to consider possibly glancing at.

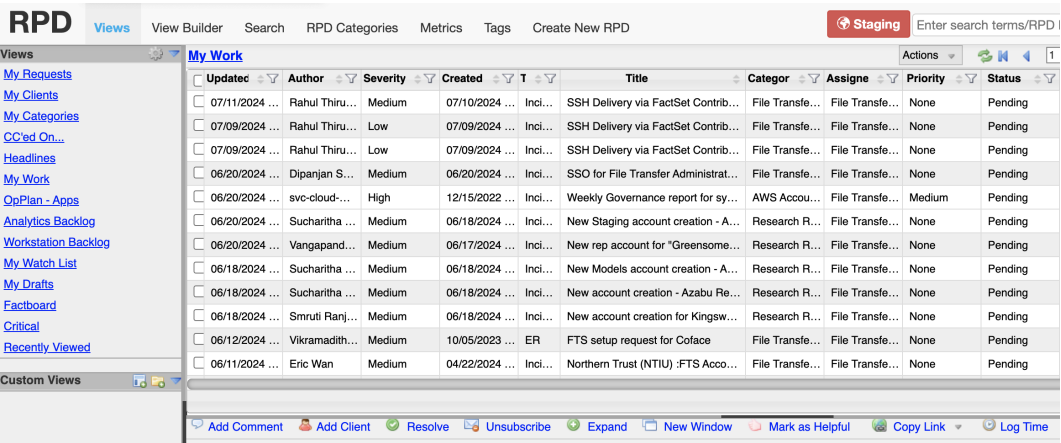
Adam Rixey

Chapter 2

Problem Statement

In Chapter 1 we have vividly gone through the importance of FTS and its various usages in enterprises, as we have understood if the clients become larger in number, to ease the process of transactions it is important to automate processes. Talking with the API of the application is the best to automate solution for robust processes, although bash scripts can be also composed to get the desired results, python gives rather extended modules to ease the processes and can be run in most of the Operating systems.

The Request for Product Development is the ticketing system that is used in the firm to address issues such as providing access to hosts, creating users in a variety of services for example AWS within the company, and also client requests for specific products, etc. A small overview of the homepage of the RPD is shown below 2.1



The screenshot shows the RPD interface with a sidebar on the left containing links like 'My Requests', 'My Clients', 'My Categories', 'CC'd On...', 'Headlines', 'My Work', 'OpPlan - Apps', 'Analytics Backlog', 'Workstation Backlog', 'My Watch List', 'My Drafts', 'Factboard', 'Critical', and 'Recently Viewed'. The main area displays a table of tickets with columns: Updated, Author, Severity, Created, Title, Category, Assignee, Priority, and Status. The table contains 10 rows of ticket data, all with a status of 'Pending'.

Updated	Author	Severity	Created	Title	Category	Assignee	Priority	Status
07/11/2024 ...	Rahul Thiru...	Medium	07/10/2024 ...	SSH Delivery via FactSet Contrib...	File Transfe...	File Transfe...	None	Pending
07/09/2024 ...	Rahul Thiru...	Low	07/09/2024 ...	SSH Delivery via FactSet Contrib...	File Transfe...	File Transfe...	None	Pending
07/09/2024 ...	Rahul Thiru...	Low	07/09/2024 ...	SSH Delivery via FactSet Contrib...	File Transfe...	File Transfe...	None	Pending
06/20/2024 ...	Dipanjani S...	Medium	06/20/2024 ...	SSO for File Transfer Administrat...	File Transfe...	File Transfe...	None	Pending
06/20/2024 ...	svc-cloud...	High	12/15/2022 ...	Weekly Governance report for sy...	AWS Accou...	File Transfe...	Medium	Pending
06/20/2024 ...	Sucharitha ...	Medium	06/18/2024 ...	New Staging account creation - A...	Research R...	File Transfe...	None	Pending
06/20/2024 ...	Vangapand...	Medium	06/17/2024 ...	New rep account for *Greensome...	Research R...	File Transfe...	None	Pending
06/18/2024 ...	Sucharitha ...	Medium	06/18/2024 ...	New Models account creation - A...	Research R...	File Transfe...	None	Pending
06/18/2024 ...	Sucharitha ...	Medium	06/18/2024 ...	New account creation - Azabu Re...	Research R...	File Transfe...	None	Pending
06/18/2024 ...	Smruti Ranj...	Medium	06/18/2024 ...	New account creation for Kingsw...	Research R...	File Transfe...	None	Pending
06/12/2024 ...	Vikramadith...	Medium	10/05/2023 ...	FTS setup request for Coface	File Transfe...	File Transfe...	None	Pending
06/11/2024 ...	Eric Wan	Medium	04/22/2024 ...	Northern Trust (NTIU) :FTS Acco...	File Transfe...	File Transfe...	None	Pending

Figure 2.1: Request for Product Development

In the above figure, the queuing tickets Can be seen, as the business day progresses

the tickets accumulate and sometimes it can go up to the tickets being not addressed by the on-call engineer, this research in FTS is thereby planning to solve this particular problem whereby the engineer has more time to address important issues taking care of servers other than clicking on the User Interface for many times, hence saving time and speeding up the processes. As the RPD shown below comes with a well-arranged REST API structure, this attempt to automate can go no wrong.

The other important requirement of this research is the Axway service, it is a French-American enterprise that provides various solutions including data monitoring, API development and enterprise solutions. The other four tasks excluding SFTP Key-generation are done by using Axway service, the engineers generally use the Axway interface to make adjustments to the requests, and now is again an attempt to talk to the API endpoints of Axway and thereby automate the processes.

In tackling the first problem, ie the generation of SFTP keys for different hostnames and usernames, we are planning to perform our logic where we wrote a Python program to talk to the API of RPD, the ticketing system, fetch the mentioned details from the API JSON output and use this information to generate keys, with the respective RPD number and save the private key to the database in Microsoft and simultaneously commenting out the RPD with the public key where the client can use this for inbound connections to the FTS servers.

In the other problems, we plan to carry out a similar procedure, where the REST API of Axway's service can be used to change the password for the existing users, change the protocol of given FTS/SFTP, etc.

The first task has been successfully carried out and seems to address the issues, the testing environment has given the desired results and soon be implemented in production environments. The project can be taken out, ie making a front end, and can be implemented as a real problem solver. This is particular to the FTS department similar steps can be made used in other departments in an enterprise such as Backend operations, database, orchestrations, and security

Among all others the major concern for the on-call engineers could be not so a formulated Request for Product Development, in this case, it can be the missing information about server names, host names, etc, in turn, might cause the program to behave tangentially and might cause issues, programmatically we can make necessary adjustments to raise log errors, but in real scenarios most of the time, the RPD would be again commented out to give the real-time information, which indeed is time-consuming. The basic error check should be also done from the client side to minimize these shortcomings.

In this study, the backed portion of the automated application will be presented, and the Python code will be run from the command line interface, where the production

hosts run on Linux servers.

2.1 Requirements

Most of the servers at the firm we carried out our research are derivatives from Redhat(Linux Distribution), furthermore, Python programming language is used for scripting(Python 3.11 v), developing environment Visual Studio Code is being used, and Iterm for UNIX command line interface.

The above information is being put into the table below. 2.1.

Platform	Code Editor	Programming language
Linux/Unix	Vscode	Python/bash
MacOSX	Vscode	Python/zsh
Windows	Vscode	Python

Table 2.1: Distributions und Editor as platforms

I cant go to a restaurant and order food because I keep looking at the fonts on the menu.

Donald Knuth

Chapter 3

State of the the Art

This chapter explores the current trends in this area including the findings of it, the study of the historical background of the attempts which have been made, the analysis of those structures and researches, and different approaches.

3.1 Related Articles

According to [9] global digitalization is making tremendous changes in business solutions, where they reveal how this transformation has paved the way to server-client type architecture and addressed the backend front-end structure of web applications, where the programming logic is performed on serve on the backed side and the latter interacts with the user interface. They also listed some examples of backed programming languages such as(Python, Ruby, Java, etc) where they also need frameworks to make production-ready web solutions.

In the article from [9] they also touch upon API, where they define API as a set of rules used to communicate between the client and the software, API according to them secures the interaction. According to them, API has different types of architectures. and they are listed down such as.

1. RPC
2. REST
3. SOAP
4. GraphQL

They [9] also describe how the above-mentioned API schemes differ from each other. The RPC specification, provides API integration just like HTTP API, thereby getting

almost the functionalities by forming a functional message to the contacting server. The server then processes this call and returns the result. The other schema namely gRPC paves the way for the old version obliterating the bad functionalities and adding new capabilities of load balancing, message tracking, etc. They are mostly used in the field of IOT(Internet of Things)

Advantages of RPC/GRP

- Simple mechanism
- Good performance
- Security addition

Disadvantages

- Less abstract
- Difficult to check for mistakes in queries
- Less abstract

Graph QL is introduced by Facebook, where the data representation is done by graphs, unlike in tables. It can get data of various complexities using one query and works very well as linking data, the major disadvantage would be low performance because of nested or linked structure.

[9] Another important schema or style is the Simple Object Access Protocol (SOAP), it uses XML format and is specific for web communication.in SOAP the data are more secure and hence used in many protocols such as FTP, HTTP, and SMTP. Advantages:

- used collectively in various protocols
- existence of error handling
- availability of metadata

Disadvantages:

- it supports only XML
- XML is a heavy object that would need more bandwidth
- parsing of data is difficult

REST is an alternative to the existing styles and is a lightweight stateless architecture, by stateless it means that no session information is saved in the server or anywhere, ie, what is needed is in the query itself, XML and JSON formats are mostly used in communication stages, IT also uses the HTTP methods to communicate: GET; POST; PATCH; PUT and DELETE.

Advantages:

- very simple architecture
- availability of caching
- acts as a direct agent to server / no third party involved

Disadvantages:

- high load on the server side

[9]In the article they also introduce the requirements while using REST, all the important are given below.

- Authentication
- Availability of logs
- Connecting to the application(to the logic)

According to [9] for the API development the access rights must be put down clearly by the developers. That is the checking of vulnerabilities of API Endpoints. Since the REST API is stateless, the authentication also takes place from endpoints. The common authentication schemes are API Key, JSON Web Tokens, Cookie-based authentication, etc.

As we have briefly gone through the API architecture, its different schemas, and its methods, as much as important as that is the awareness of automation using Python as a programming language and related works, in the article written by [18] they have stressed the importance of CRM (customer relationship management) which somewhat resembles the FTS client-server transactions, using python as the tool to perform the CRUD operations, whereas CRUD stands for Create, Read, Update, Delete. The characteristics of the Python library `requests` are looked upon and later implemented to make the API calls and other tasks.

In the above-mentioned article, [18] they provide code snippets, where they go step by step defining the functions to get the JSON data, then further manipulating the data for desired results. Firstly, they have done the authentication with the API of

Microsoft Dynamics, using OAuth2 protocol [18], which the important aspect of it is that the token is stored in memory not persisting, whereas in our environment the firm, at which we carry our research provides its authentication mechanism which we can discuss in later parts of this article. Requests library is implemented to perform CRUD operation, whereby the corresponding Python variants would be: GET - READ

POST - Create

PATCH - Update

The workflow of their project is enumerated below to make it easier to comprehend. The authentication steps would be as follows:

- the setting of parameters(Initialisation)
- creating a post request to get the authentication token
- various other parameters are included on the request(password, username, etc.)
- next step would be the checks done by the API
- the response would be sent back in JSON format
- extract the required token using Python dictionary

Secondly, in the attempt to retrieve data from a random table, the workflow would be:

- the API URL is assigned to a variable
- writing a function(GET request)
- decode the response in UTF
- parsing the JSON response

To make changes in the program, they [18] have made a patch request to update the table present in it. The corresponding workflow would be:

- creating the function using the credentials from the previous method
- include the keyword `patch` to update the data
- gets the JSON response

- parse the data when needed
- checking for responses

Similarly, another function that uses the `post` method to insert data into a table, is also given in the [18] project. Furthermore, performance and efficiency checks have been also carried out which justifies the quantitative measure that they promised for this particular project, which is quite improved when Python is used for the implementation. All the relevant data can also be found on this web page [1].

As we investigate further the article presented by [5] shows extensive insight into the automation of the testing process in software development using Robot Framework. As the authors mention it uses a keyword type of development, which means the syntax is of a table form and the testing is done uniformly. One can also make custom-made libraries of it in major programming languages given that it provides XML output. The results shown [5] in the paper have brought about significant results which show the success in automating tasks rather than doing them manually, which we can take the confidence from to further work on the topic.

In a study conducted by [15] they used Python scaling using large volumes of data in investment and security sectors. The data is offered by such companies in large volumes on their websites, which subsequently can be accessed on particular open APIs, using particular programming languages, which in this case used Python. They conducted a study on two different companies and found differences in authentication and method of transfer. The table below shows the companies in South Korea where open APIs are supported, HTS is a system that is being used to trade stocks at any time [15].

Securities and Invest Services	HTS	Open API
SK Securities Co.	Yes	No
Hanwha Investment & Securities Co.	Yes	No
Shinyoung Securities Co.	Yes	No
Bookook Securities Co.	Yes	No
Hanyang Securities Co.	Yes	No
Samsung Securities	Yes	No
Kyobo Securities Co.	Yes	No
KB Securities Co.	Yes	No
Hyundai	Yes	No
Daishin Securities Co.	Yes	Yes
Hi Investments and Securities Co.	Yes	No
NH Investments and Securities Co.	Yes	No
Kiwoom Securities Co.	Yes	Yes
eBest Investments and Securities Co.	Yes	Yes
Mirae Assests Daewoo Co.	Yes	No
Yuanta Securities Co.	Yes	Yes
Eugene Investment Co.	Yes	No
Korea Investment and Securities Co.	Yes	No
KTB Investments and Securities Co.	Yes	No
Hana Financial Investment Co.	Yes	Yes
IBK Securities Co.	Yes	Yes
Cape Investments and Securities Co.	Yes	No
Total	22	6

Table 3.1: Companies in the study [15]

Of all the companies they have taken out two companies with which they make

API calls using Python further studies show that many of the companies are yet to support Python in open API. The table is shown below.

Securities and Investment Services	Supported Software
Dashin Securities Co.	Microsof Visual Studio (VB, C#, C++, .Net etc.), Microsoft Ofiice (Word, Excel, Powerpoint, Access etc.), Microsoft Internet Explorer Dynamic HTMLM, Python, and Borland Delphi
Kiwoom Securities Co.	VB, C/C++, C#, Excel,and Delphi
eBest Investments&Securities Co.	VB, C/C++, C#, and Delphi
Yuanta Securities Co.	VB, C/C++, C#, and Delphi
Hana Financial Investment Co.	MFC, C#, VB, Delphi, Excel
IBK Securities Co.	Microsof Visual Studio (VB, J++, C++ etc.), Microsoft Ofiice (Word, Excel, Powerpoint, Access etc.), Microsoft Internet Explorer Dynamic HTMLM, Delphi

Table 3.2: API supporting companies in the study [15]

To connect to the open API of Daishin Securities [15] CYBOS plus which is the name of the open API has to be incorporated, thus after coding the CYBOS plus

and enhancing the connection to the server of the company, authentication, and all other checks are done using this portal of CYBOS.

To connect to the open API of eBest Investments they provide a package called Xing-API, which like above enhances the connection to the server. The relevant data is given in the [15] article.

The attempt conducted by [7] to extract efficient data using Python programming language using the Application Programming Interface of beam therapy equipment, turned out to be much better for obvious reasons, than manual extraction. It is a simple experiment as shown in the picture given below.

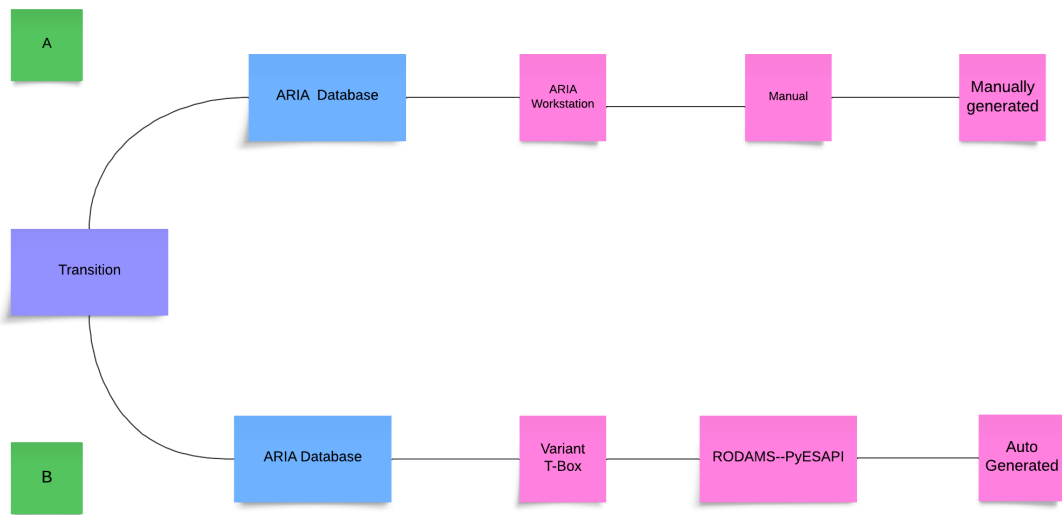


Figure 3.1: A. manual extraction, B. python implemented [7]

The facilities used in the study, particularly for Radio Therapy and Radiation Oncology, developed an automation strategy, where their native instrument is named TPS (Treatment Planning System), in this approach, they have created an environment using the API of the vendor, in this case, Varian Medical systems [7], and thus newly created environment being the PyESAPI, which fetches data from the database from Varian TPS. The script developed is named RODAMS, Radiation Oncology Data Mining Script, which has resulted in immense improvement. The details of the pilot project can be found in the [7].

3.2 Standards

In this attempt to inform oneself about the existing standards, although the REST API (Representational State Transfer) structure that we discussed earlier in the other

capital, a study conducted by [12] where they have analyzed 32 open guidelines for the REST API design, the attempt is to distinguish between the API and get to a best practice of designing custom API. A lot of big firms are making their API available so that the public can access the data and services [13]. They have scrutinized 32 REST API documents found mainly through Git Hub.

REST API is a stateless protocol, that is any change in the API implementation does not affect the user's code or data, and has a separate server-client setup [8]. The study collected the most important topics 27, where they coded the rules and looked out for the similarities and dissimilarities in the given guidelines and again categorized the set into the 10 most used. The 10 most stood-up categories of the chosen 27 are given below.

- Status Codes
- Response Structure
- Standard Methods
- Naming
- Versioning
- URL
- Error Responses
- Backward Compatibility
- Documentation
- Custom methods

They also highlight some differences in the API structure used by different agents, for example when it comes to versioning, Google and Zalando recommend using three-number versioning while others suggest a two-number versioning, another issue is the placement of version at the beginning or in the URL and also highlights some differences in the API structure used by different agents, for example in versioning, Google and Zalando recommends using three-number versioning while others suggest a two number versioning, another issue is the placement of version at the beginning or in the URL.

Another important divergence could be seen in Backwards Compatibility, which means the older version code could be also used in newer versions, then the newer version is said to be backward compatible [12], some guidelines discourage breaking

compatibility although it is costly, changing codes for every new version. While some guidelines encourage backward compatibility.

The work done by [16] puts forward 11 tips for building usable REST API's in a particular industry of health sciences. REST APIs are easier to develop than classical SOAP (Simple Object Access Protocol) according to [16]. According to [16], although requirement documentation can be found on the two websites, namely [2] [4], about the obscure explanations the study has decided to come up with their findings. The suggested tips for building a useful REST API are given below.

1. good documentation
2. clear URLs for APIs
3. uses standardized formats for data example: JSON
4. standardized HTTP headers
5. standardized HTTP responses
6. enabling sharing the resources across networks
7. sharing links to clients for the usage of API
8. standard way of authenticating
9. nonstop running of API
10. versioning

I have always wished for my computer to be as easy to use as my telephone; my wish has come true because I can no longer figure out how to use my telephone.

Bjarne Stroustrup

Chapter 4

Structure of the Thesis

4.1 Requirements

4.2 Architecture

4.3 Implementation

Chapter 5

Experiment and Discussion

References

- [1] data-processing. <https://www.semanticscholar.org/paper/Concurrent-Data-Processing-in-Microsoft-Dynamics-Yadav-Sehgal/>. Accessed: 2024-09-30.
- [2] ietf. <https://www.ietf.org/>. Accessed: 2024-09-30.
- [3] masterthesisbern. http://rvs.unibe.ch/research/pub_files/Gel0.pdf. Accessed: 2010-09-30.
- [4] w3. <https://www.w3.org/>.
- [5] Neha S Batni and Jyoti Shetty. A comprehensive study on automation using robot framework. 2020.
- [6] Gaurav Bora, Saurabh Bora, Shivendra Singh, and Sheikh Mohamad Arsalan. Osi reference model: An overview. *International Journal of Computer Trends and Technology (IJCTT)*, 7(4):214–218, 2014.
- [7] Rohit Singh Chauhan, Anirudh Pradhan, Anusheel Munshi, and Bidhu Kalyan Mohanti. Efficient and reliable data extraction in radiation oncology using python programming language: A pilot study. *Journal of Medical Physics*, 48:13–18, 2023.
- [8] Roy T. Fielding. Architectural styles and the design of network-based software architectures"; doctoral dissertation. 2000.
- [9] DV Kornienko, SV Mishina, SV Shcherbatykh, and MO Melnikov. Principles of securing restful api web services developed with python frameworks. In *Journal of Physics: Conference Series*, volume 2094, page 032016. IOP Publishing, 2021.
- [10] Sumit Kumar, Sumit Dalal, and Vivek Dixit. The osi model: overview on the seven layers of computer networks. *International Journal of Computer Science and Information Technology Research*, 2(3):461–466, 2014.
- [11] Mingyu Lim. C2cftp: Direct and indirect file transfer protocols between clients in client-server architecture. *IEEE Access*, 8:102833–102845, 2020.
- [12] Lauren Murphy, Tosin Alliyu, Andrew Macvean, Mary Beth Kery, and Brad A Myers. Preliminary analysis of rest api style guidelines. *Ann Arbor*, 1001(48109):4, 2017.
- [13] Brad A. Myers and Jeffrey Stylos. Improving api usability. *Commun. ACM*, 59(6):6269, may 2016.

- [14] Robbi Rahim, Solly Aryza, Pristisal Wibowo, Adek Khadijatul Z Harahap, Abdul Rahman Suleman, Edi Epron Sihombing, Yuswin Harputra, Muhammad Rahman Rambe, Andysah Putera Utama Siahaan, H Hermansyah, et al. Prototype file transfer protocol application for lan and wi-fi communication. *Int. J. Eng. Technol*, 7(2.13):345–347, 2018.
- [15] Gui-Yeol Ryu. A study on comparison of open application programming interface of securities companies supporting python. *The International Journal of Advanced Smart Convergence*, 10:97–104, 2021.
- [16] Aleksandra Tarkowska, Denise Carvalho-Silva, Charles E Cook, Edd Turner, Robert D Finn, and Andrew D Yates. Eleven quick tips to build a usable rest api for life sciences. *PLoS computational biology*, 14(12):e1006542, 2018.
- [17] Zouheir Trabelsi and Imad Jawhar. Covert file transfer protocol based on the ip record route option. *Journal of Information Assurance and Security*, 5(1):64–73, 2010.
- [18] Akash Yadav and Jai Sehgal. Concurrent data processing in microsoft dynamics crm using python. *International Journal of Innovative Research in Computer Science and Technology*, 2023.

Online-Resources