Ollscoil na Gaillimhe
University of Galway

Final Assessment 2022/23

End-of-Module Assignment

Module Code: **MS804**

Module Name: **Systems Development & Project Management Assignment**

Student Name and ID: **Dilip Venkatesan Sankar 22225743**

"In submitting this work, I confirm that it is entirely my own. I acknowledge that I may be invited to undertake an online interview if there is any concern in relation to the integrity of my submission."

Assignment Title: Summation Learning Journal

Part One: Introduction

Background: The healthcare sector is among the largest and fastest growing in the world. Client health records are extremely sensitive and important, and since they are updated daily, the data must be refreshed regularly to ensure that the customer has accurate results in their records so they may use them in emergency situations. These days health sector produces an enormous amount of data that cannot fit in relational databases. So, these industries are using a parallel computing solution called **HADOOP** with the help of **Hadoop distributed file system(HDFS)** for storing and **SQL** for querying the data. Files are used to hold consumer information, and more files are being created every day. HDFS is not designed to access these files fast and it requires more resources from the cluster to query a customer table. This is called an **HDFS small file issue** [1.0]

Motivation: This is a highly intriguing problem because many other industries and businesses that use Hadoop as a big data platform are also experiencing the same issue. All of these businesses are struggling with this problem and spending resources, time, and money trying to fix it.

Fig 1.1 Hadoop framework issues

Issue with Small Files
Vulnerable by Nature
Security
Processing Overhead
Iterative Processing
Supports only Batch Processing
Data Flair

For instance, if a business has 10 customers, Hadoop will produce 10 files to store 10 customer data, each of which may be 1 MB in size. If developers wish to run some analyses on these customers, Hadoop will open and close these files 10 times to locate each client, which is costly, demands more resources, and time-consuming process. And over time, the number of files could reach millions, creating a disastrous problem. The issue can be resolved by creating one file with all the 10 customers' data so there will be only one open and close process with minimum resources required for accessing the customer data. This process is called **Compaction.**

According to my prior experience, I encountered a similar problem whereby querying or loading data into a table took longer than expected utilizing more resources from clusters, leading to SLA breaches and client complaints.

Due to inadequate preparation and a lack of futuristic thinking, this is a widespread issue, and business analysts and developers are seeking solutions later on.

Part Two: Old and New Assumptions

Any software development will involve management and planning, which is where **project management** comes into play. Project management involves using information, skills, tools, and procedures to deliver something important to the consumer.

Challenging Assumptions: We will explore the below assumptions made in relation to the problem and provide recommendations and evidence respectively. The assumptions are based on my prior experience with Hadoop.

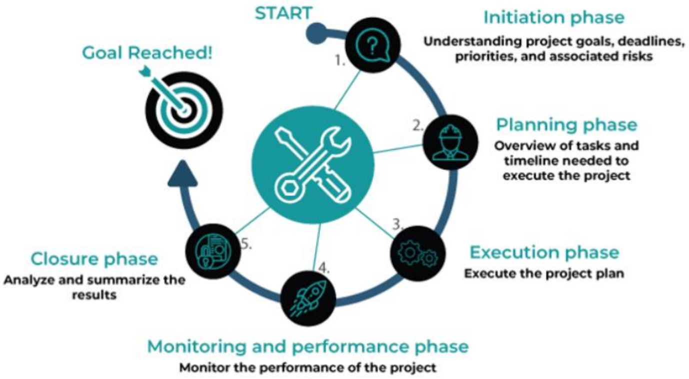
1. Improper planning, management, and disregarding steps cause setbacks

The project life cycle is the "**Series of phases that a project passes through from its start to its completion**".

The project life cycle contains five parts, which are crucial because they offer a methodical technique for project delivery. However, in certain businesses, not all the aspects are considered during these processes, and some steps are skipped altogether, which leads to chaos.

Assumption: The health industry is experiencing rapid data expansion and due to poor planning, most projects encounter problems in the mid-phase because managers and leaders are more concerned with establishing goals and objectives than they don't take the phases in the middle seriously.

Fig 1.2 Project life cycle


START
Initiation phase
Understanding project goals, deadlines, priorities, and associated risks
Goal Reached!
1.
2.
Planning phase
Overview of tasks and timeline needed to execute the project
3.
Execution phase
Execute the project plan
4.
Monitoring and performance phase
Monitor the performance of the project
5.
Closure phase
Analyze and summarize the results

In the early stages, managers must adopt a futuristic mindset and make plans for these future issues. The task is simple while there is little data, but problems such as performance, cluster resource limitations, and space issues may arise as the data increases. Similar to this, my project started to have slowness issues because of several small files and this resulted in a significant negative impact because downstream were not receiving any data, leading to SLA breaches and escalations.

Recommendation: In the beginning, the managers should consider both the roadblocks and the final objective during the **project cycle**. These problems should be brought to the developer's attention, and during the **planning and implementation stages**, developers need to plan and determine how to store the data in an efficient manner because initially the records and files may be few and easy to access for analysis and insights, but over time there may be millions of files and records.

The testing and analyzing team must record the project's performance during the **project monitoring phase** and during **post-production validation** (PPV) and must inform the managers so that the problem may be fixed in the following sprint. During **project closure**, all the issues and problems need to be documented instead of only documenting the success stories.

Despite the fact that many businesses still use the traditional approach, it is viewed as risky since it relies on upfront planning, in which all tasks are specified from beginning to end and progress is difficult to gauge as the project moves forward at each stage.

2. Determining requirements leads to project success

Document Analysis: The foundation of project development is determining requirements, and document analysis is a useful tool for gathering data about the technology currently in use. Even though we have other methods like **surveys, interviews, and observations** it has their own limitations. So, examining the organizational documents will give us more information about the current systems and tools they support. Document Analysis will provide us with the below information's,

- Why are the current systems designed in this way?
- Why some of the features are left out?
- Issues with the existing systems?

Outcome: The project managers and leaders gather information on the current system and organizational operations, and the project participants debate the viability and constraints of the system/tool that the organization now employs.

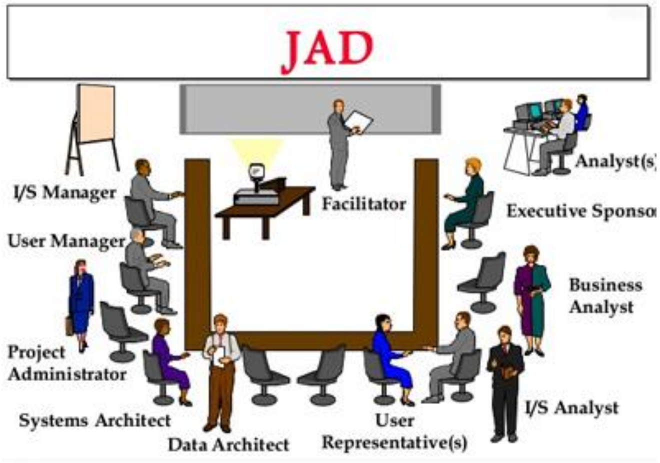
Assumption and supporting evidence: The most crucial choice a system analyst must make while gathering data is what a system should perform and how it should operate. My previous projects were excellent at **document analysis**, and they took this stage quite seriously, in my experience. We looked at Hadoop's restrictions and the reasons it can't support some of its capabilities. We have a background on the kinds of problems that can arise with this instrument thanks to the document analysis from other projects and businesses.

The project decided to switch from **HADOOP to Amazon S3 for storage and Snowflake for querying the data** when it reached a certain level because the data in the healthcare sector will be growing day by day and, based on the document analysis, Hadoop storage and accessing files will reach stagnation as the data grows. This was one of the crucial decisions made thinking about the futuristic problem which may come in HADOOP like maintenance, storage, and resource issue.

JAD: One of the most popular techniques and modern methods of determining requirements for project teams and management is to work together to identify the system requirements which is called JAD. It was created by IBM in the 1970s and is widely utilized in many projects because it encourages active conversation. Business users, session leaders, managers, and system analysts are present at the JAD workshop.

Given that they are the intended users and key stakeholders, the JAD should be planned to include all essential participants. Due to schedule issues, meetings may be held without them in some projects. As a result, the requirements may not be clear, and the final product may not be what the clients wanted.

Fig 1.3 Joint application development



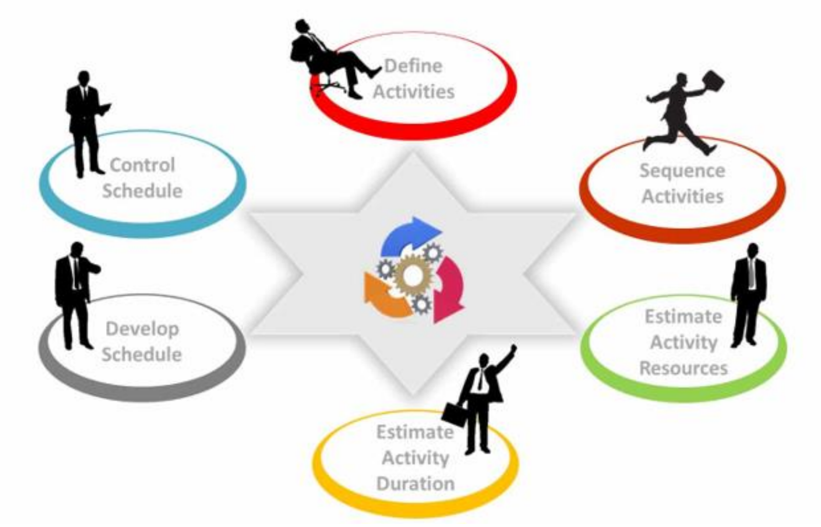
Assumption and supporting evidence: For instance, the project I worked on was extremely organized, and the **JAD** involved all of the important team members, including the stakeholders and users. This was done because it is crucial to have a clear understanding of the client's needs, and because all conflicts and discrepancies can be resolved in the same meeting, saving time.

Both traditional and modern techniques of identifying needs are in use around the world, but in my experience, they work best together because skipping even one criterion can result in a project failing at any stage.

3. Multiple user stories and poor time management a reason for sprint failure

User stories are short, straightforward statements that describe the expected results and explain how a piece of work will add a specific value to the client. The product owner typically creates the user stories during sprint planning, and the team members choose which stories they will focus on during this sprint. The member must carefully consider the task's needs, time, and difficulty before assigning the story points. However, it is crucial to note that the story is scored such that it may be finished in a single sprint. Hence time management plays a vital role.

Fig 1.4 Practices for a successful User Story



Assumptions: In a few cases based on my prior experience, below are a couple of scenarios that caused user story failures,

1. Prior to assigning an activity, it's crucial to ascertain whether the story can be completed by a **single person or a team, how much time has to be allocated, and whether the resource has one or more deliverables scheduled**. An 8-pointer story with 3 points for analysis and 5 points for implementation using agile methodology was given to a single resource to address the Hadoop small file issue. Even though it had predecessor and successor processes, the task was difficult for one person to complete because it required a variety of skills, various tools, and a lot of resources.
2. The resource failed to alert about the blockers and the time constraint. So based on his improper updates in the daily scrum call, the PO assumed that meaningful progress had been achieved. As a result, another 3-pointer story was assigned to him. Even while there was progress, it was very slow and did not get finished in the allotted time, so it was moved to the

following sprint and engaged a well-defined group because the Hadoop small file issue was creating so many issues and escalations.

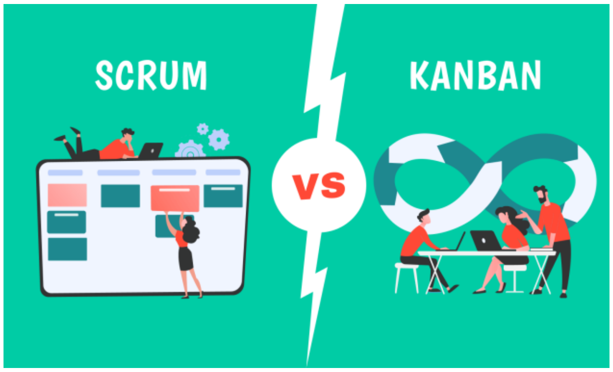
Recommendations: Some of the traditional tools used to track the progress of the resources are **1) PERT 2) Gantt and 3) Network diagram**. Despite their widespread use, these have several drawbacks, such as setup, usage, and visibility, which can be challenging in complex and large-scale projects. An alternative that displays all the details about the user stories on one page with a description and chronology is the **Kanban board.**

It is crucial for the scrum master to monitor each resource's progress with respect to their user stories and to allot enough time, points, and resources to each user story for better results. The resource in the aforementioned scenario had enough time to fix the Hadoop small file issue, yet failures still occurred as a result of **improper updates, poor tracking, poor time management, and backlogs**. In order to minimize last-minute rushes or delivery delays, it is crucial to disclose any issues and blockers during the scrum call.

4. Successful usage of Kanban rather than Agile(SCRUM):

Team members may view the status of every piece of work at every step of development because **Kanban** makes the work more visible. A team member gains insight into who is doing what and can spot and fix process bottlenecks. In order to align the product with the needs of the client, the developers also work directly with the leadership. Whereas Agile is a cross-sectional model changes are made continuously according to feedback and requirement.

Fig 1.5 Agile(Scrum) Vs Kanban



Assumption: Although Agile is commonly utilized, from my experience I would choose Kanban since it allows for better **reprioritization and the elimination of tasks** that never benefit the team. During the small file issue in Hadoop, even though it was a shop stopper, a story cannot be allocated to a resource right away because it adheres to the agile methodology and does not support the sprint backlog as well. Whereas in Kanban, due to its flexibility the story can be assigned based on the criticality

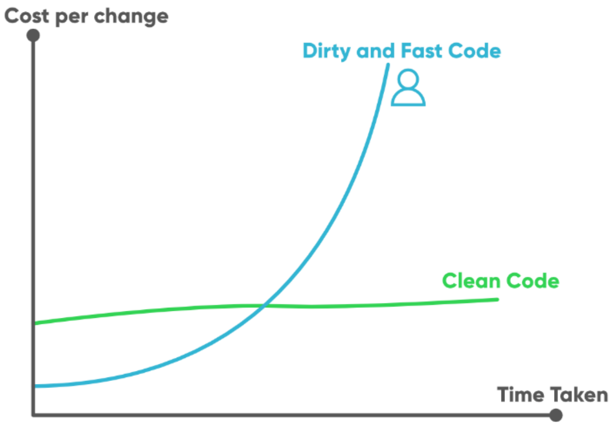
Recommendations: Even though Agile allowed iterative development, the **quality assurance (QA)** was overworked in the end even though the sprint was broken down into smaller segments which helps the team to focus on high-quality development, testing, and collaboration. Based on Kanban, QA must be used throughout the whole development process, and **post-production validation** (PPV) must be carried out following each release. Therefore, the PPV will ensure that data flow is constant, data is as per standards, flow is optimized, cluster resource utilization is not beyond the provided level for the jobs, and HDFS space occupied is not more or less than the expected level which could have identified the Hadoop small file issue in the earlier stages. Agile sprint planning also takes up the entire day for the scrum team, but Kanban requires far less setup work to get started, and because it is visible to everyone on the team, anybody can point out problems and offer input to help prevent problems from being overlooked or missed.

5. Error-free code does not mean a clean code

Assumption: Making sure the code is both clean and bug-free is crucial for novices. Developers neglect standards in their haste to write bug-free code, which raises costs and complicates things. It's crucial to create your code in a way that your coworkers can comprehend it because performance deterioration is another significant problem that most projects suffer as they advance and makes it challenging for the following developers to understand the work.

From the below figure, for any analysis and enhancement, a clean code will require less cost and less time whereas a bad code will require more time and more cost.

Fig 1.6 Time Vs Cost for program enhancement and analysis



Recommendation: The majority of the codes in my earlier projects, which ran without bugs for many years, were written complexly, which made them more difficult to understand. Below are some of the common issues found in the majority of programs,

- Improper and less commenting
- Usage of nested codes
- Redundancy
- Code reusability

It was crucial for the developers to improve the code and address the Hadoop small file issue as soon as possible, but because the problem wasn't discovered for a while, the initial developers and POC left the organization, and it fell to the new team members to address the problems.

Even though the solution was already known, the code's complexity made it difficult for the developers to identify the problematic section and the location where the fix should be made. The cost to the business was greatly increased as a result. The problem might have been resolved quickly and put into production if the original developers had **adhered to standards**.

Conclusion

It is crucial for the project managers to run the project in a very organized manner, arrange, and prioritize each teamwork for success. Team members should actively participate at every stage of the project cycle and have comprehensive knowledge of the tool and need because the success of the project is determined not only by its output but also by how well it performs over time. Proper research, adherence to procedures, task delegation, time management, and knowledge of the necessary resources can all help with this.

Regular team meetings and one-on-one conversations with the managers are also advised as a means of resolving problems within the team. To ensure there are no backlogs, obstacles, or escalations, the manager/scrum master must effectively track each team member's progress and maintain a good rapport with them. While the testing and QA team is in charge of performing continuous testing and ensuring that all procedures are efficient and effective, the development team is also accountable for writing code that is free of bugs and follows all standards.

Even with all the protective measures in place, a project can occasionally be unpredictable and not go as planned, but it is the managers', team leaders, and members' responsibility to be ready for all roadblocks, and possible outcomes and ensure that the project is successfully completed within the specified deadline and budget.

References:

[1.0] https://blog.cloudera.com/the-small-files-problem/#:~:text=If%20you're%20storing%20small,as%20a%20rule%20of%20thumb.

[1.1] https://data-flair.training/blogs/advantages-and-disadvantages-of-hadoop/

[1.2] https://www.invensislearning.com/blog/5-phases-project-management-lifecycle/

[1.3] https://jaymeholmes.com/jad.html

[1.4] https://aims.education/study-online/what-is-project-time-management/

[1.5] https://digitalleadership.com/blog/kanban-vs-scrum/

[1.6] https://www.geeksforgeeks.org/7-tips-to-write-clean-and-better-code-in-2020/