

PROBLEM ANALYSIS AND SOLUTION DEVELOPMENT

Module Name: Analysing and Solving
Problems

Module Code: ACCA 4019

Student Name: Hannah Lwetutte

Student ID: 2509323

Group:2

Lecturer's Name: Dr Redhwan Al-amri

Table of Contents

Table of Contents.....	1
1. Introduction	2
2. Deconstruct the Problem.....	2
2.1 MoSCoW Technique Requirement	3
2.2 Variables Identification	4
3. Proposed Solution	5
3.1 Flowchart	6
3.2 Flowchart Explanation	7
4. Reflection and Justification	8
4.1 Abstract thinking	8
4.2 Sub-dividing	8
4.3 Justification of solution	8
4.4 Limitation.....	8
4.5 Assumption.....	9
5. Conclusion.....	9
References.....	9

1. Introduction

This report analyses a real world problem using recognised problem solving techniques. This will be surrounding the topic of group projects. Group projects help develop teamwork, communication, and problem-solving skills. It ensures that effective collaboration is practiced as it enables students to share knowledge, improve decision making, and prepare for professional working environments [1]. Without effective scheduling, collaboration becomes less organised, increasing the risk of poor project outcomes [3].

However, organising regular group meetings can become a challenge due to the fact that students often have different class timetables, part-time employment, and personal commitments. These scheduling conflicts may result in missed meetings, unequal distribution of work, and delays in completing project tasks before submission deadlines.

This report analyses the scheduling problem by identifying the key requirements, constraints and variables involved. The MoSCoW prioritisation technique is applied to classify the requirements, while a logical solution is developed using a flowchart to show the meeting scheduling process. Finally, the report evaluates the problem solving techniques used, including abstract thinking and problem decomposition, and the assumptions, strengths, and limitations of the proposed solution are discussed.

The analysis is based on the following problem scenario provided within the assessment

Problem Scenario: Students in a college computing course are facing difficulties in organising group project meetings. Many students have part-time jobs, different timetables, and personal commitments, making it hard for all group members to meet regularly. As a result, some groups struggle to complete their tasks on time, leading to delays in project submissions and uneven workloads among members.

2. Deconstruct the Problem

The problem concerns college computing students who are experiencing difficulties arranging group project meetings due to having different timetables, part time jobs, and personal commitments. These scheduling conflict reduce the availability of common meeting times, making it difficult for all group members to attend. These scheduling conflicts reduce the availability of common meeting times, making it difficult for all group members to attend. According to Bransford and Stein, breaking a complex problem into smaller, manageable components enables a more systematic approach to identifying suitable solutions [2].

The first component of the problem is student availability. Every student has a different/personal timetable which usually consists of: lectures, work commitments,

and personal responsibilities such as: family, some may have children etc. If student availability is not accurately identified, the scheduling process cannot generate a reliable meeting time.

The second conflict is timetable conflicts. If one or more students are unable to attend the meeting during the proposed meeting time, there must be an alternative slot/meeting time to minimise conflicts.

The third component is project deadlines. Meetings must be arranged before the weekly deadline to ensure that tasks are discussed, allocated and completed on time. Effective planning improves collaboration and increases the likelihood of successful completion of the project [3].

Finally, equal participation is important and should be considered. Every student has different commitments, including lectures, work and personal responsibilities. There should be regular attendance to meetings which helps distribute work fairly and improves communication among group members [4].

2.1 MoSCoW Technique Requirement

The MoSCoW prioritisation technique is commonly used in software development and project management to classify requirements according to their importance [5]. The MoSCoW technique ensures that the essential system requirements are identified first which allows to focus on solving on the important parts of the scheduling problem. This technique supports structured decision making.

Priority	Requirement
Must Have	Store each student's availability.
Must Have	Identify a meeting time that all group members can attend.
Must Have	Ensure meetings occur before the weekly project deadline.
Must Have	Prevent timetable conflicts.
Should Have	Suggest an alternative meeting time if the first choice is unavailable.
Should Have	Notify all group members of the confirmed meeting.
Could Have	Recommend the most suitable meeting time automatically.
Could Have	Support online meetings if face-to-face meeting are not possible.
Won't Have	Integration with external calendar systems within the first version of the solution

The “Must Have” requirements address the main problems identified in the scenario and ensuring that suitable meeting times are found, timetable conflicts are avoided, and meetings are arranged before project deadlines. The “Should Have” and “Could Have” requirements improve the usability and flexibility of the system without affecting its core functionality, while the “Won’t Have” requirement defines the scope of the initial solution and prevents unnecessary complexity.

2.2 Variables Identification

Variable	Description
studentName	Stores the name of each group member.
groupSize	Number of students within the project group.
availableTime	Available meeting times for each student.
lectureSchedule	Fixed Lecture Timetable.
workSchedule	Part- time working hours.
personalCommitments	Other unavailable periods
meetingTime	The meeting time selected
deadlineDate	Weekly project deadline
meetingConfirmed	Boolean value indicating whether all members can attend
ConflictFound	Boolean value indicating whether a there is an existing scheduling conflict.

The variables identified above represent the data that is needed to develop a scheduling solution. Each variable performs a specific role within the scheduling process. For example : availableTime and workSchedule are compared to identify the common free slots group members have and the deadlineDate ensures that meetings are arranged before the project. These variables allow the system to compare availability, detect conflict and identify suitable meeting times. Together with the prioritised requirements identified using the MoSCoW technique, these variables provide the foundation for designing the flowchart presented in the next section.

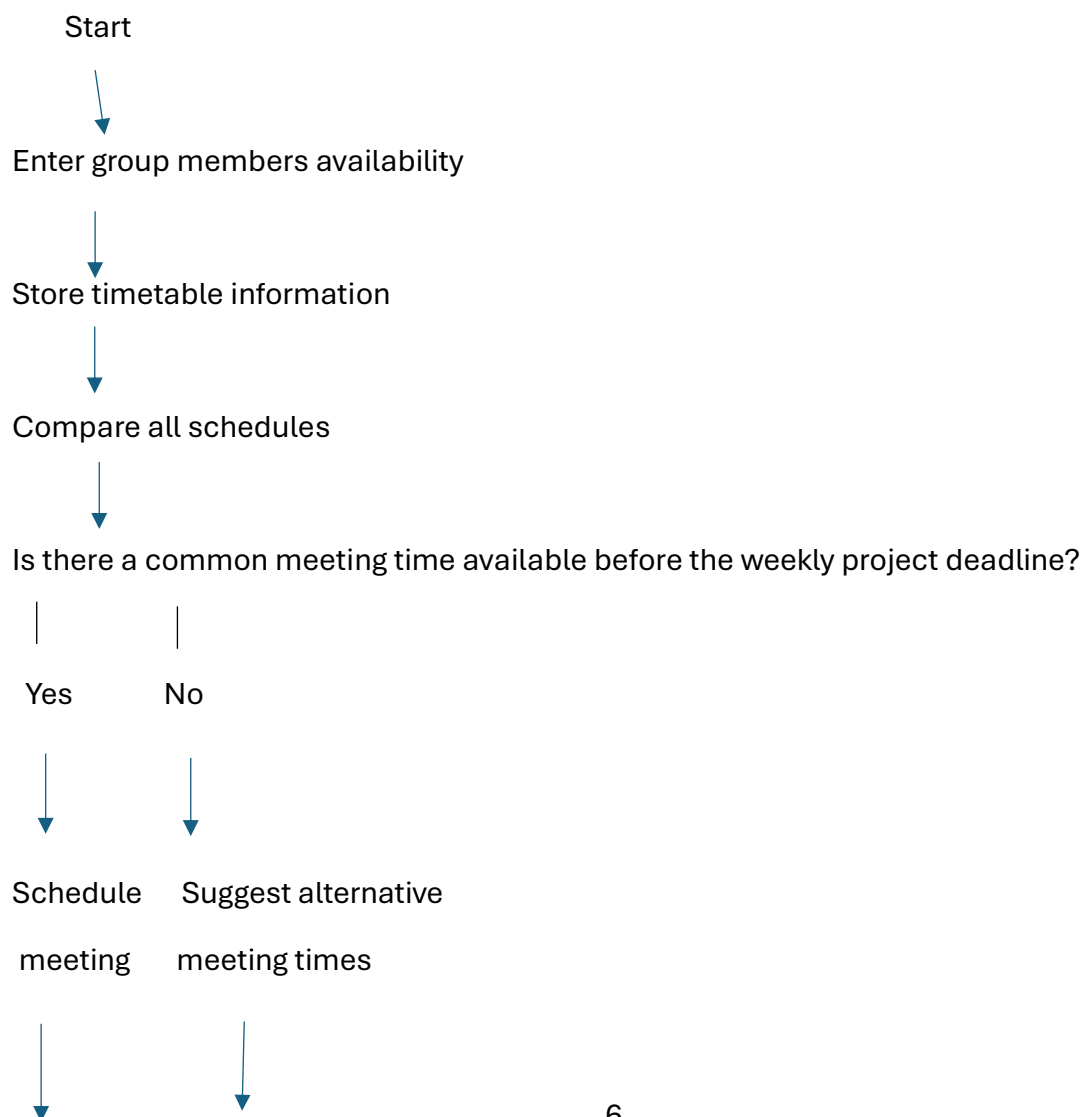
3. Proposed Solution

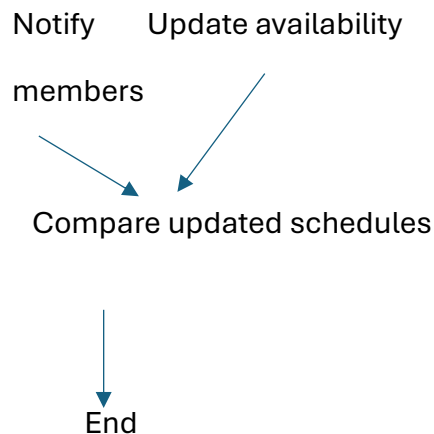
The solution I propose is a computer-based meeting scheduling system that automatically identifies suitable meeting times for student project groups. The system requires each group member to enter their weekly timetable, including lectures, work commitments and other unavailable periods. This information is stored in a database and analysed to identify common free time slots.

The system compares every member's availability and checks whether a suitable meeting time exists or is available before the weekly project deadline. If a common time is found, the meeting is scheduled automatically, and all group members receive a notification with the meeting information such as: date, time, location or online meeting link. If there is no common availability found, the system will suggest the closest available alternatives and requests members to update their availability until a suitable meeting time can be arranged.

The solution reduces manual planning, minimises timetable conflicts, improves communication and increases the likelihood of meeting project deadlines. By automating the scheduling process, the system reduces timetable conflicts, improves communication between group members and increases the likelihood of completing project tasks before deadlines while all members are still involved. This approach also follows software engineering principles, where complex problems are divided into logical processes before implementation [1]. Effective planning and scheduling are also recognised as essential components of successful project management [3].

3.1 Flowchart





3.2 Flowchart Explanation

In this section, The flow chart process is explained, the process begins when each member of the project group enters their availability into the scheduling system. This includes lecture timetables, part-time work, and any personal commitments that will or may prevent them from attending meetings. The information is kept securely and used during the scheduling process.

Once all the data has been collected, the system compares each timetable to identify overlapping free periods. A decision is then made to determine whether a common meeting time exists before the weekly project deadline.

If a common meeting time is available, the meeting is automatically scheduled by the system, and notifications are sent to each group member.

If there are no common meeting times, the system recommends alternative time slots that create the fewest scheduling conflicts. The group members are then asked to update their availability, if possible, after which the scheduling process is repeated until an acceptable meeting time is identified.

The use of decision points within the flowchart ensures that every possible scheduling option is considered before a final meeting time is selected. The loop within the flowchart allows the scheduling process to continue until a suitable meeting time is identified, this logical approach improves efficiency, reduces missed meetings and supports successful project completion. The solution reflects software engineering principles that recommend analysing requirements and following structured processes when designing systems [1]. It also supports effective planning practices discussed in project management literature [3].

4. Reflection and Justification

4.1 Abstract thinking

Abstract thinking was applied by focusing only on the information necessary to solve the scheduling problem. Instead of considering every personal detail of each student, the solution only required data such as availability, timetable, project deadlines, and existing commitments. Irrelevant information, such as personal preferences or unrelated activities, was excluded. This made the solution simpler and more efficient. This made the solution simpler and more efficient. According to Sommerville, abstraction is an important software engineering technique because it allows developers to focus on the essential aspects of a problem while ignoring unnecessary complexity [1].

4.2 Sub-dividing

The scheduling problem was divided into several smaller tasks. These included collecting availability from each group member, storing the information, comparing schedules, identifying common free time, checking deadlines, and notifying students of the meeting time. Breaking the problem into smaller stages made it easier to develop a logical solution. Bransford and Stein [2] explain that decomposing a problem into manageable parts improves problem-solving by allowing each stage to be solved individually before combining them into a complete solution.

4.3 Justification of solution

The solution that is proposed was selected because it automates the meeting scheduling process, reducing the amount of manual planning required by students. The system identifies suitable meeting times by comparing the availability of all group members, which helps minimise timetable conflicts and ensures meetings take place before weekly deadlines. This approach saves time, improves communication, and reduces human error. Kerzner [3] states that effective scheduling and planning are essential for successful project management because they help teams meet deadlines and improve overall performance.

4.4 Limitation

Although the proposed system offers several advantages, there are some limitations. The system relies on students entering accurate and up-to-date availability. If incorrect or outdated information is entered, the suggested meeting times may not be suitable. In addition, unexpected events, such as illness or changes to work schedules, cannot always be predicted by the system. Technical issues could also affect the scheduling process. Limitations could also be reduced in future version by introducing features such as calendar synchronization and automatic reminders.

4.5 Assumption

Several assumptions were made when designing the solution. It was assumed that every group member has access to the internet and is able to use the scheduling system. It was also assumed that students would regularly update their availability whenever changes occurred. Finally, it was assumed that all project deadlines were known in advance and that notifications sent by the system would be received by all users. These assumptions define the conditions under which the proposed solution is expected to operate effectively.

5. Conclusion

In conclusion, the proposed meeting scheduling system provides an effective solution to the problem of organising group meetings for college projects. By collecting student availability, comparing schedules and automatically selecting suitable meeting times, the system reduces scheduling conflicts and improves communication. The use of problem-solving techniques such as abstraction and decomposition enabled the problem to be analysed systematically and supported the development of a logical solution.

References

- [1]-[5] I. Sommerville, Software Engineering, 10th ed. Harlow, UK: Pearson, 2016.

- [2] J. D. Bransford and B. S. Stein, The IDEAL Problem Solver, 2nd ed. New York, NY, USA: W.H. Freeman, 1993.

- [3] H. Kerzner, Project Management: A Systems Approach to Planning, Scheduling, and Controlling, 13th ed. Hoboken, NJ, USA: Wiley, 2022.

- [4] A. P. Moran, Managing Agile: Strategy, Implementation, Organisation and People, 2nd ed. Cham, Switzerland: Springer, 2015.

- [5] D. Clegg and R. Barker, Case Method Fast-Track: A RAD Approach. Reading, UK: Addison-Wesley, 1994.