

Analysing And solving problems

by Akinyemi Olabisi Akinwunmi

Submission date: 17-Jul-2026 04:24PM (UTC+0100)

Submission ID: 286893973

File name: 67246_Akinyemi_Olabisi_Akinwunmi_Analysing_And_solving_problems_746295_1175958994.docx (115.13K)

Word count: 2172

Character count: 11843

TITLE PAGE

APPLIED COMPUTING (FOUNDATION)

ANALYSING & SOLVING PROBLEMS

MODULE CODE: ACCA4019

**TITLE : PROBLEM ANALYYSIS AND SOLUTION
DEVELOPMENT**

STUDENT NAME: AKINWUNMI AKINYEMI

STUDENT ID NO: 2515474

TABLE OF CONTENTS

TITLE PAGE.....	1
PROBLEM DECONSTRUCTION	3
Describing the Problem.....	3
Constraints.....	3
MoSCoW analysis	4
Variables.....	4
PROPOSED SOLUTION.....	4
Solution Overview	4
Explaining the Flowchart	6
REFLECTION AND JUSTIFICATION.....	7
REFERENCES	9

PROBLEM DECONSTRUCTION

Describing the Problem

Group projects are an important part of college education. It fosters collaboration, critical thinking and interpersonal management[1]. However, coordinating meeting can be difficult as each student member of the group is different with varying schedules, obligations and activities[1].

The first part of problem analysis is to identify the problems/weaknesses in a current system and separate into individual issues that can be analysed independently[2].

In the given scenario, the stakeholders are the students who are members of the group. The issues faced can be broken down to differences in schedule, lack of free time (leading to burnout and work undone) and uneven workload among members. As meetings cannot be arranged regularly, there is a lack of communication among members and therefore progress cannot be tracked. Tight schedules lead to burnout and work undone, which will cause delay in the assignment and late submission.

Requirements

Requirements are simply what you want the system to do and what characteristics it must possess[3].

Business requirements: States the overall aim of the new system, which in this case is to ensure prompt submission of the assignment.[3]

User requirements: states what users (the students) need to be able to do[3].

1. They need to be able to attend meetings
2. They need to be able to avoid burnout due to overwork
3. They need to have enough time to do the work completely and right.

Functional Requirements: These describe what the system must do[3].

1. Confirm the schedule and availability of each member and note their free periods
2. Set meeting times that accommodate members' schedules
3. Ensure meetings occur before deadline
4. Notify members of scheduled meetings

Constraints

Constraints are special issues or restrictions that influence the efficiency of a system[3]. Russel and Norvig defined constraints as rules that defines when solutions are valid in a problem[2].

In this context, the constraints are:

1. Project deadlines cannot be changed
2. Personal commitments influence availability
3. Students have to meet with their part time work demands
4. Students may update their availability late

MoSCoW analysis

This method is used to prioritize requirements[4].

1. Must Have: Highlight availability of members. Identify the deadline. Generate suitable meeting times
2. Should Have: Allow members update their availability.
3. Could Have: Remind members about meetings. Agenda for meetings sent to the student.
4. Won't Have: AI meeting summaries, Automated attendance.

Variables

These are pieces of information required by the solution

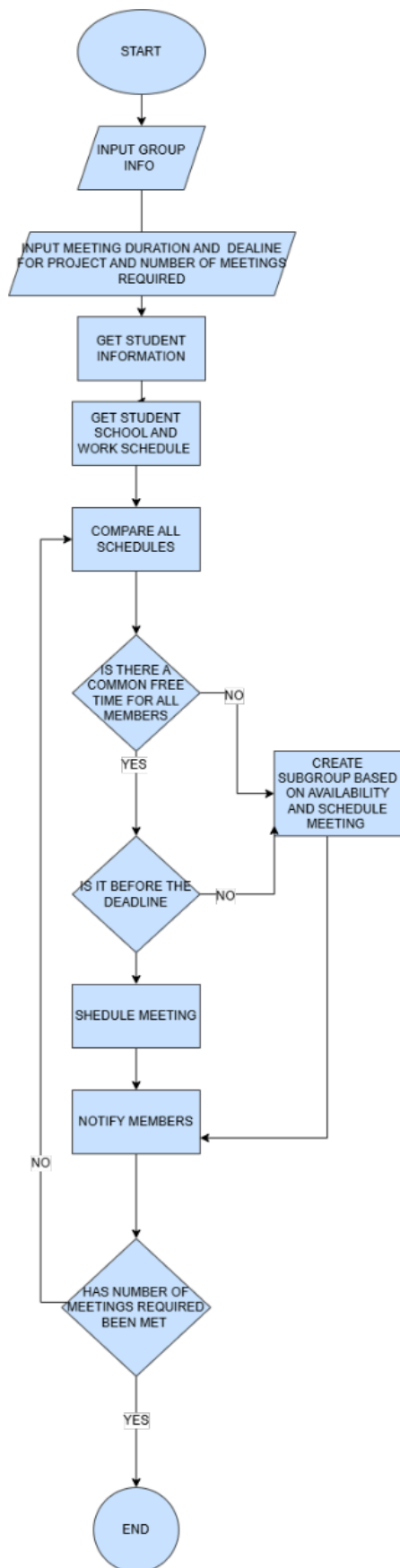
In this context: StudentID (student identifier), GroupID (identifies the group), Student School Schedule, Student Work Schedule, Meeting Length, Number of meetings required, Deadline for project.

PROPOSED SOLUTION

Solution Overview

The proposed solution involves a well-organized process to identify suitable meeting times while considering the availability of members and the project's deadlines. The solution follows a logical sequence where members availability are collected compared and reviewed before a meeting is scheduled. Data inputting, comparison and decision making are incorporated throughout the process.

Flow Chart



Input where identified as parallelograms, processes where identified as rectangles, decisions where identified as diamonds[5]

Explaining the Flowchart

Step 1: Begin the process[5]. The start symbol ideally marks the beginning of a flow chart to make the beginning of the process evident[6].

Step 2: Input the group variables. This includes group name, means of communication with members (means to notify them about the meeting). This ensures proper identification and appropriate communication of meeting schedule which will be important for later steps of the flow chart.

Step 3: Input meeting duration, deadline for project and number of meetings required. These are necessary variables the system needs to appropriately schedule the meetings. The meetings cannot be scheduled for after the completion of the project. The meeting time needs to be inserted as it tells whether the free time will be sufficient and therefore how long the overlap needs to be for it to be characterised as free time. The number of meetings required may differ depending on the project given and it tells the programme when to stop running.

Step 4 and 5: Get student information including student ID as identifiers and their work and school schedule. This is the heart of the solution since it is their schedule the system uses to ascertain possible meeting time.

Step 6: The schedules of the group members needs to be compared to identify free/available time during their days. This compares the schedules they provided and returns overlapping periods in that schedule where they are not occupied by a task.

Step 7: Assess if there is a common available time across schedules.

Step 8: If there is no common available time across schedules, subgroups should be created based on availability first, then roles. This factors in the reality of the members busy schedule and the fact that no common free time may be noted, especially with larger groups[7]. If yes, the system moves on to assess if the available time is before the deadline.

Step 9: Assess if the set meeting is before the assignment deadlines. As a rule meetings must not be set after the assignment deadline. In this instance we are assuming that post submission evaluation meetings are not necessary. If there are no available meeting time before the deadline, create subgroups.

Step 10: Schedule the meeting. The time is flagged as appropriate for a meeting and the meeting is scheduled.

Step 11: Notify the members of the group about the scheduled meeting including time and venue. Members should be notified through an appropriate communication channel. The communication method should be one suitable for everyone[8].

Step 12: Assess if the number of meetings specified as the input has been met. If it has not been repeat the process, if it has end the program

Step 13: End the program. Just as the start symbol ideally marks the beginning of a flow chart , the end symbol marks the end of the flow chart and therefore the end of the process[6].

Assumptions made:

1. The solution assumes that on the student's priority list going for meetings comes after school and work. This may not reflect the reality where students also want to schedule leisure time [7].
2. The solution assumes that students will provide accurate schedules. This doesn't factor in human error. It also assumes that a student's schedule is set in stone and there is no surprise responsibility or duty[8].
3. The solution assumes that all member have access to a means to receive notifications. It doesn't factor in the fact that notification of the meeting might not reach a group member[6].
4. Students can be split into subgroups based on their availability.
5. A suitable meeting location is always available and does not affect the timing of meetings.
6. It assumes that the system can handle organizing the schedule of the students, regardless of the number of students.

The objectives of the assignment as to accommodate all the group members schedule, minimise time table conflicts, scheduling issues, ensure all meetings take place before the project deadlines.

The comparing student's individual schedules allows ideal meeting periods to be highlighted, increasing meeting attendance. During this, conflicting schedules and timetables are also dealt with. Splitting into smaller groups when necessary allows for the inclusion of "all" group members. The deadline decision box prevents meetings from being set too late.

Dannis et al stated that a system must meet the functional requirements, meaning that it must perform processes to meet the users needs[3]. They explained that a system must also have good performance. This means that the system must be reliable, able to meet the demands placed on it, quick and available[3]. A good system should also be easy to learn (usability) and support users in completing their set tasks[3].

This system is easy to understand and implement. A good system must have compatibility and integration[3]. This means that it must work in the existing environment, which in this instance is the college. It is adaptable to groups of larger sizes due to the "splitting" feature. It meets the required objectives. It encourages participation of all the members. It is also easier to adapt into a software solution.

REFLECTION AND JUSTIFICATION

Several techniques were applied in the solving of this problem.

Decomposition:

Rich et al. defined decomposition as the process of breaking down a problem into its subcomponents[9]. This is reflected in the first segment of this paper.

The meeting scheduling problem was split into smaller part. The solution was first split into five smaller parts: get the necessary variable data, compare schedule, get appropriate meeting time, ensure the time is valid then schedule and notify the members.

Abstraction:

This involves focusing on only information important to the system[10], in this case: availability, deadline, required number of meetings etc. Details that didn't contribute to the objective of the system were not collected.

Algorithmic Thinking:

This is the ability to create a solution that follows an ordered sequence of steps which are organized logically and sequentially[11]. The solution follows a logical sequence of processes and decision points, ensuring that the meetings are set through an ordered and clear process.

Justification of the solution:

The proposed solution is appropriate because it meets the needs of the users and solves the specified objectives. It evaluates every member's availability for a meeting, this means that members do not have meeting clashing with their work or school time. It further ensures that scheduling is done based on what is best for the whole group and not a select few.

The inclusion of deadline verification helps to ensure that projects are done before the deadline and prompt submission is possible. The ability to continue scheduling meetings until the required number is met makes the process flexible and it makes the solution ideal for a college system (in a college system different projects will require different amounts of meetings).

Finally, the subgroup scheduling ensures that there is a backup plan in case there is no time where all members are available. The initial idea was to create meetings based on the availability of 70% of the members. However, this idea was chosen as it is more inclusive of the members.

Limitations:

This solution is not without its limitations:

1. Students may provide inaccurate schedules
2. Students may have ever-changing schedules
3. It doesn't factor in the quality of the meeting, just the scheduling
4. Sub group meetings may reduce communication and coherence among the larger team.
5. It has several assumptions that are discussed under the second section

Future Improvements:

Several improvements can be made as the system is tried and tested.

1. Send agenda of the meeting with notification to improve the quality of the meeting
2. Obtain real time scheduling updates
3. Enable system to take attendance and send AI generated meeting summaries to members that were not present.

REFERENCES

- [1] H. Donelan and K. Kear, "Online group projects in higher education: Persistent challenges and implications for practice," *Journal of Computing in Higher Education*, vol. 36, no. 2, pp. 435 to 468, Aug. 2024, doi: 10.1007/s12528-023-09360-7.
- [2] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Hoboken, NJ, USA: Pearson, 2021.
- [3] A. Dennis, B. H. Wixom, and R. M. Roth, *Systems Analysis and Design*, 8th ed. Hoboken, NJ, USA: Wiley, 2022.
- [4] K. S. Ahmad, N. Ahmad, H. Tahir, and S. Khan, "Fuzzy_MoSCoW: A fuzzy based MoSCoW method for the prioritization of software requirements," in *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, Kannur, India, 2017, pp. 433 to 437, doi: 10.1109/ICICICT1.2017.8342602.
- [5] Romney, M. B., & Steinbart, P. J. (2021). *Accounting Information Systems* (15th ed.). Pearson.
- [6] useWorkspace.AI Team, "Start/End (Terminal)," useWorkspace Documentation, Jan. 12, 2025. [Online]. Available: <https://www.useworkspace.dk/en/docs/shapes/start-end>.
- [7] Grund, A., Brassler, N. K., & Fries, S. (2014). Torn between study and leisure: How motivational conflicts relate to students' academic and social adaptation. *Journal of Educational Psychology*, 106(1), 242 to 257.
- [8] Donelan, H., & Kear, K. (2023). Online group projects in higher education: Persistent challenges and implications for practice. *Journal of Computing in Higher Education*.
- [9] P. J. Rich, G. Egan, and J. Ellsworth, "A Framework for Decomposition in Computational Thinking," in *Proceedings of the 2019 ACM Conference on International Computing Education Research*, Toronto, ON, Canada, 2019, pp. 124 to 133, doi: 10.1145/3304221.3319793.
- [10] N. O. Ezeamuzie, J. S. C. Leung, and F. S. T. Ting, "Unleashing the Potential of Abstraction From Cloud of Computational Thinking: A Systematic Review of Literature," *Journal of Educational Computing Research*, vol. 60, no. 4, pp. 877 to 905, 2022, doi: 10.1177/07356331211055379.
- [11] V. J. Shute, C. Sun, and J. Asbell Clarke, "Demystifying computational thinking," *Educational Research Review*, vol. 22, pp. 142 to 158, 2017, doi: 10.1016/j.edurev.2017.09.003.

70

/100

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

GRADING FORM: ACCA4019_A1 EXAM JUNE 26

AKINYEMI OLABISI AKINWUNMI

70

CRITERION 1 (0%)

1. Engagement with Literature Skills

 This is for assesment 2
10 sources used which support the information well.

15

CRITERION 2 (100%)

2. Knowledge and Understanding Skills

 The MoSCoW technique is applied quite minimally. However, functional and non-functional requirements and constraints are included. Useful variables identified, though not in a table, no description or data types specified and not all variables are in the correct case - Pascal or Camel case would have been acceptable.

13

CRITERION 3 (0%)

3. Cognitive and Intellectual Skills

-  Some good reflection, justification and limitations with some useful details regarding the techniques used.

15

CRITERION 4 (0%)

4. Practical Application Skills

-  The flow chart is syntactically ok minus a missing 'Yes' after 'Is it before the deadline' decision. Logically, the flow is ok. However, If a meeting is not before the deadline, there will be a problem with creating the meeting because it will be too late, so this is a real issue here. The description is good.

20

CRITERION 5 (0%)

5. Transferable Skills for Life and Professional Practice

-  Some spelling errors, good language used, more care required with the spacing and titles at end of page would have improved the overall appearance, a correctly structured reference list.

7

DISCLAIMER

Results are provisional until confirmed by the Examination Board. The mark in this Feedback is given out of 100. However, Late Submissions and RESIT students will have their work capped in their Final Result Letter, as according to the University's regulations in place. RESIT/REWORK DISCLAIMER Resit/rework marks will be capped at [tutor to input 40% or 50% as appropriate] following the exam board but appear uncapped here.