# 40.014 Engineering Systems Architecture

# &

# 40.012 Manufacturing and Service Operations

**2D Project Report 1**

**Group 7**


| **Names** | **Student IDs** |
|---|---|
| Toh Kin Yong | 1006875 |
| Khoo Teng Jin | 1007104 |
| Ernest Ong | 1006346 |
| Luvena Liethanti | 1007105 |
| Julianna Sherine | 1007035 |
| Phoebe Kuek | 1007065 |
| Asher Yeo | 1006950 |

## DEFINE THE PROBLEM

In the project proposal, we have collected the necessary data and drawn a concept sketch in the form of an operation network that provides us with an overview of the MRP (Material Requirements Planning) system we will be working on. Using the proposal as a reference, we will be using a "top-down" approach, generating and exploring possible solutions to tackle the problem. Before working on the solution, we will determine the processes and steps which we will be taking to design the solution.

a. *Tailoring the process*

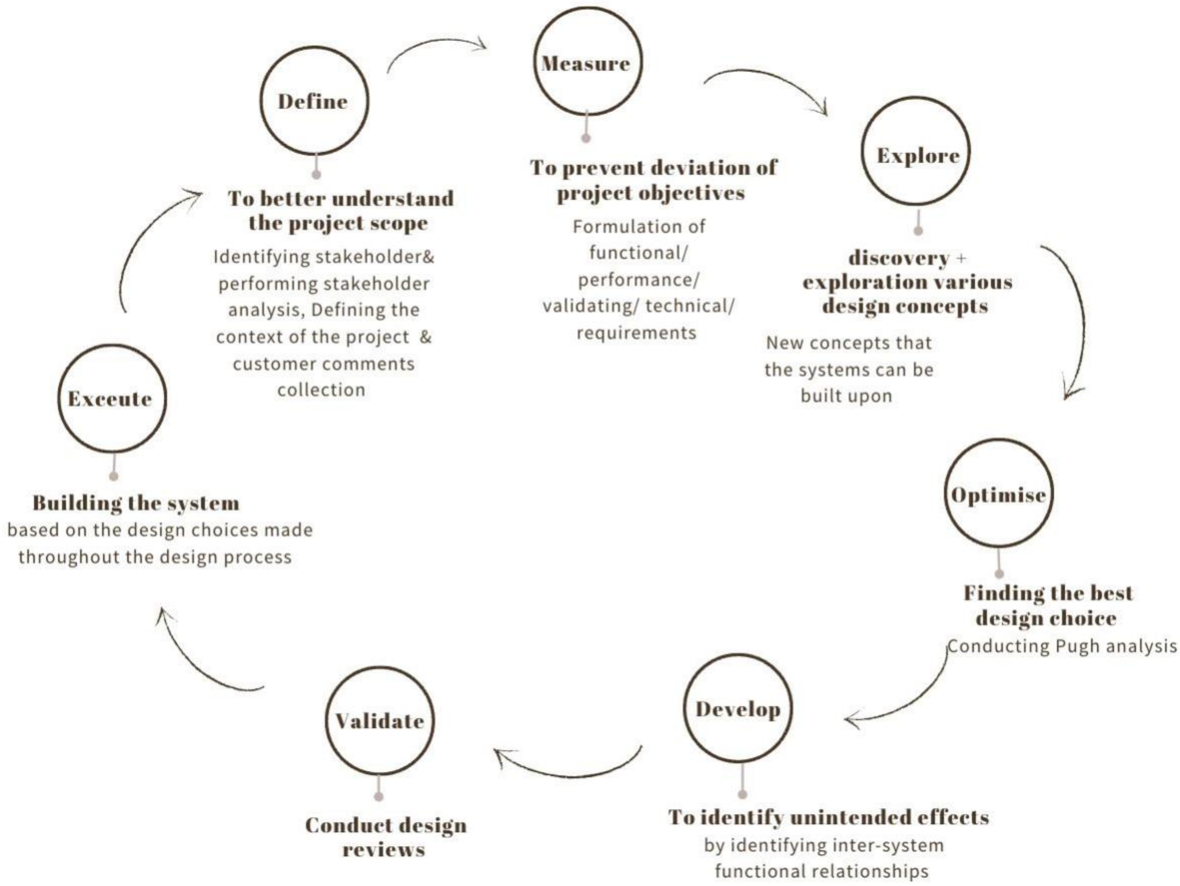


Figure 1. Project process diagram

Given the project's time constraints, we do not wish to allocate additional resources to iterate the process scheduling problem further if our system is functioning effectively. Thus, we have decided to remove the iteration step for this project.

b. *Identify the Owner, Customer, and User*

- Owner: In our project, the owners are the members of group 07.
- Customers:

- Prof. Rakesh and Prof. Zeyu: Involved in guiding the project and hence represent an authoritative approval level for the project's success. ○ Manufacturing companies: Their operational needs shape the requirements and features of the programme and their investment in the project's development makes them key customers.

- Users:
  - Project management team of the 2D project: Tasked with the development of the programme and management of the project's timeline and deliverables and are hence primary users. ○ Project manager in manufacturing companies: Seek to optimise operation schedules and are hence end users who reply on the programme's functionalities to enhance productivity and efficiency of their companies.

*c. Stakeholder Analysis*

We identified the initial customer expectations of our product:

- Needs:
  - User-friendliness: The need for a scheduling programme with an intuitive interface that allows users to input data and interpret output easily.
  - Efficiency: The need for the scheduling programme to operate efficiently, minimising computation time while delivering high-quality solutions.
  - Reliability: The need for the scheduling programme to consistently produce accurate and dependable schedules, even for complex and large-scale problems.
- Goals:
  - Efficiency: Design the scheduling programme to operate with optimal performance, reducing computation time while maintaining highquality output. Incorporate an adjustable solution optimality feature to balance between speed and precision based on user preferences.
  - Notification and Stability: Implement a robust notification mechanism to alert users of unexpected terminations or issues during the scheduling process. Ensure that the system can recover quickly from crashes and maintain overall stability.
  - Adaptability: Design the scheduling programme to accommodate various scheduling scenarios, including different types of operations and varying levels of complexity

    - Scalability: Ensure the scheduling programme can scale to handle larger datasets and more complex scheduling problems without affecting its performance.
- Objectives:
    - Achieve a reliability rating of at least 95% for schedule accuracy by industry standards for complex operations scheduling systems.
    - Implement a robust notification mechanism that alerts users within 5 seconds of unexpected terminations during the scheduling process.
    - Allows for compatibility with various operational scenarios and input types, accommodating different BOMs, due dates, and routing configurations.

We then measured the effectiveness of our product:

- Goal: To ensure that the scheduling programme provides reliable and efficient scheduling.
- Question: How reliably and efficiently does the scheduling programme produce accurate schedules based on input data?
- Idea metric: The percentage of schedules produced accurately and on time i.e. 95% accuracy
- Approximate Metric: The number of schedules produced without errors and within an acceptable time frame over a given period.

By establishing clear goals and formulating relevant questions, we can effectively measure the reliability and efficiency of our scheduling system, ensuring it meets user expectations for accuracy and timeliness.

*d. Defining the context*

Project managers or schedulers often want to find an optimal schedule for manufacturing processes. After entering their desired start and end dates, the optimization program would output a chart to display a feasible (preferably optimal) schedule of processes that start as late as possible while not exceeding due dates. This process is done from a user interface that allows the scheduler to key in the required data, which will be put into an optimization program. The program then retrieves the order file containing data on the bill-of-materials to calculate an optimal schedule by returning the start and end times of each process which will be displayed on the user interface.

We should note that since there are limited resources, the program might not be able to solve all scheduling problems, specifically larger and more complex problems. Thus, the program should have functions to notify users if the solver is unable to obtain a solution.

## Context Diagram

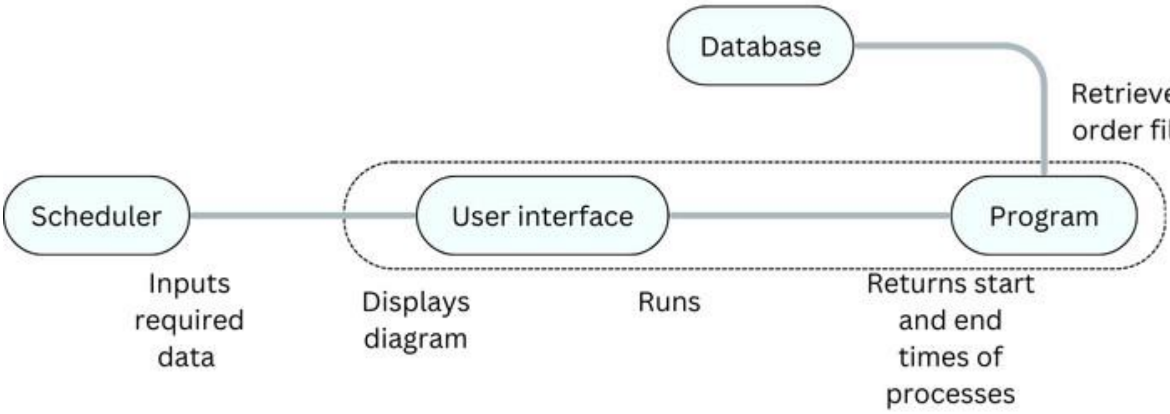


Figure 2. Context diagram of the project entities and its interactions

## Context Matrix

| | *Scheduler* | *User interface* | *Program* | *Database* |
|---|---|---|---|---|
| **Scheduler** | | Inputs required data | | |
| **User interface** | Displays diagram | | Runs | |
| **Program** | | Returns start and end times of processes | | Retrieves order file |
| **Database** | | | | |

Figure 2. Context matrix of the project entities and its interactions

*e. Voice of Customer*

Customer comments were collected online from manufacturing companies who use MRP systems, and stakeholders (Professors, classmates).

We flagged out some of the more relevant customer comments and then grouped them by their affinities to create our Voice of Customers to better visualize the needs and other gather more information in improving the functionality of the program of the system.

| | |
|---|---|
| Make the interface of the system visually appealing | Easy application configuration and the most user-friendly interface I’ve seen of any planning and scheduling applications when there is visual appeal along with simple mechanism of the system |
| | I need to be able to know the locations for my input data on the interface, so a colourful visual would be helpful |
| Make the system have greater flexibility to cater to specific needs | I want to use a system that has a calendar utility for me to place the start and end dates |
| | I want to be able to choose the % optimality of the output schedule to cater to my individual needs |
| | The more utilities the system has, the more user friendly it is for me |
| | It is helpful if the system lets me know what had caused the process to fail |
| | “Select” utility can better help me filter items in in the order file |
| Make the system have the minimal features to produce the scheduling output | I need an interface that can take in “orders file” containing the items, quantity and due dates successfully |
| | A Gantt chart to visualise the schedule (output) |
| | Using a system that produces an output at a slow rate is frustrating, i need the schedule to be produced within a reasonable time |

| | |
|---|---|
| Make the system able to pinpoint areas of errors if it fails to run | The system should let me know if my input data is incomplete or if there is an issue in the network that caused the system to fail |
| | It is helpful if the system let me know what had caused the process to fail |

## MEASURE THE NEEDS AND SET TARGETS

a. Use Cases

The table below shows the priority of each use case that has been categorised into 2 categories: primary use case and secondary use case.

| **Primary Use Case** | **Priority** |
|---|---|
| The user opens the interface on their mobile device. | H |
| The user chooses our product among some other products. | M |
| The user uses our interface. | H |
| The user navigates through the interface. | M |
| The user interacts with the interface elements. | H |
| The user inputs data via the interface. | H |
| The user retrieves the desired data from the interface. | H |
| **Secondary Use Case** | |
| The user decodes the interface, repackages it and sells it for profit. | H |
| The user opens the interface for UI/UX research purposes. | L |
| The user hacked the system and stole the data. | H |

| | |
|---|---|
| The user uses the system to test their wifi speed. | L |

b. Functional Requirements

After analysing the use case and the voice of the customer, the following table summarizes the functional requirements:

| **Functional Requirements** |
|---|
| The system shall take in a CSV file (that contains the Bill-of-Materials, Operation Network, and Routings) as the input. |
| The system shall provide a feasible solution that is within 5% of the optimal solution. |
| The system shall provide a schedule of operations as the output. |
| The system shall provide a schedule that is formatted via a Gantt chart for readability. |
| The system shall allow users to learn how to navigate the utilities by themselves within 5 minutes of first seeing the interface. |
| The system shall deliver the output in the shortest time possible for the given dataset and solver. |

c. Validate Requirements

| **Functional Requirements** | **Written Correctly** | **Technically Correct** | **Satisfy Stakeholder** | **Feasible** | **Verifiable** | **Redundant/ Over specified** |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| **System shall take in a CSV file (that contains the Bill-of-Materials, Operation Network, and Routings) as the input.** | Yes | Yes | Yes | Yes | Yes | No |
| **System shall provide a feasible solution that is within 5%** | Yes | Yes | Yes | Yes | Yes | No |
| **of the optimal solution.** | | | | | | |
| **System shall provide a schedule of operations as the output.** | Yes | Yes | Yes | Yes | Yes | No |
| **System shall provide a schedule that is formatted via a GanttChart for readability.** | Yes | Yes | Yes | Yes | Yes | No |

| | | | | | | |
|---|---|---|---|---|---|---|
| **System shall allow users to learn how to navigate the utilities by themselves within 5 minutes of first seeing the interface.** | Yes | Yes | Yes | Yes | Yes | No |
| **System shall deliver the output in the shortest time possible for the given input and solver.** | Yes | Yes | Yes | Yes | ***No*** | No |

After validating the functional requirements that our team developed, we can conclude that only one of the functional requirements (system shall deliver the output in the shortest time possible for the given input and solver) is not valid. This is because it will be challenging for us to verify if this functional requirement has been satisfied, since it will be difficult for us to determine what the shortest possible solving time is for a given dataset and solver.

## EXPLORE THE DESIGN SPACE

In this phase, we intend to explore the alternative choices on how we are going to build the system.

| **Area of Focus** | **Possible Choices** |
|---|---|
| Mixed Integer Linear Programme Solvers | Mosek Tools Optimizer |
| | GLPK Optimizer |
| | Gurobi Optimzer |

| | |
|---|---|
| | (IBM) CPLEX Optimizer |
| | (Python) PyGLPK/ PyMathProg/ CMPL/ PuLP/ Pyomo |
| Mixed Integer Linear Programme Formulations | Formulation 1 (Agrawal, A., Harhalakis, G., Minis, I., & Nagi, R. (1996)):<br>$\text{Maximize}: \text{cost}(\mathcal{S}) = \min\{S_i:\ i = 1,\ 2, \ldots, n\}$ (2.1)<br>subject to:<br>$S_{d(i)} \geq C_i \quad i = 1, 2, \ldots, n;$ (2.2)<br>$C_i = S_i + t_i \quad i = 1, 2, \ldots, n;$ (2.3)<br>$C_e \leq D_e \quad e = 1, 2, \ldots, n_f,$<br>$D_e =$ due date of $e$th final assembly;<br>$C_e =$ completion time of the last of $e$th final assembly; (2.4)<br>$\delta_{ij} + \delta_{ji} = 1 \quad i,\ j \in I_{\mathcal{K}},\ i \neq j;\ \mathcal{K} = 1, 2, \ldots, m;$ (2.5)<br>$S_i - C_j \geq M\left(\delta_{ij} + \Delta_{ik} + \Delta_{jk} - 3\right)$<br>$i,\ j \in I_{\mathcal{K}},\ i \neq j;\ \mathcal{K} = 1, 2, \ldots, m;\ k = 1, \ldots, f_{\mathcal{K}};$ (2.6)<br>$\sum_{s=1}^{f_k} \Delta_{is} = 1 \quad i \in I_{\mathcal{K}};\ \mathcal{K} = 1, 2, \ldots, m;$ (2.7)<br>$S_i \geq 0 \quad i = 1, 2, \ldots, n;$ (2.8)<br>$\delta_{ij} \in \{0, 1\} \quad i,\ j \in I_{\mathcal{K}};\ i \neq j;\ \mathcal{K} = 1, 2, \ldots, m;$ (2.9)<br>$\Delta_{ik} \in \{0, 1\} \quad i \in I_k; \mathcal{K} = 1, 2, \ldots, m; k = 1, 2, \ldots, f_{\mathcal{K}}.$ (2.10) |
| | Formulation 2 (Xu, J., & Nagi, R. (2013)): |

| | |
|---|---|
| | $\text{Minimize}:\ Z$<br>Subject to:<br>$S_j \geq F_i, \quad \forall i, j \in N, \chi_{ij} = 1$ (1)<br>$S_j \geq ES_j, \quad \forall j \in N$ (2)<br>$F_j \leq Z, \quad \forall j \in N$ (3)<br>$F_j = S_j + t_j, \quad \forall j \in N$ (4)<br>$\psi_{ij} + \psi_{ji} = 1, \quad \forall i, j \in I_Y, i \neq j, Y \in \{1, 2, \ldots, w\}$ (5)<br>$S_i - F_j \geq (\psi_{ij} + \phi_{iy} + \phi_{jy} - 3)M, \quad \forall i, j \in I_Y, i \neq j,$<br>$y \in \{1, 2, \ldots, f_Y\}, \quad Y \in \{1, 2, \ldots, w\}$ (6)<br>$\sum_{y=1}^{f_Y} (\phi_{jy}) = 1, \quad \forall j \in I_Y, Y \in \{1, 2, \ldots, w\}$ (7)<br>$\psi_{ij} \in \{0, 1\}, \quad \forall i, j \in I_Y, i \neq j,$<br>$Y \in \{1, 2, \ldots, w\}$ (8)<br>$\phi_{jy} \in \{0, 1\}, \quad \forall j \in I_Y,$<br>$y \in \{1, \ldots, f_Y\}, Y \in \{1, 2, \ldots, w\}.$ (9) |

| | |
|---|---|
| Programming Languages for Interface | HTML, CSS, JavaScript |
| | Python, PyQt, PyTkinter |
| | Swift, SwiftUI |
| | Kotlin, Jetpack Compose |
| | C#, Xamarin |
| | Java |
| | C++ |
| | Tauri (aka Rust) |
| Programs for Interface Design | Figma |
| | React |
| | Sketch |
| | InVision |
| | Adobe XD |
| | R Shiny |
| Programming Languages for Running the Mixed Integer Linear Programme | Julia/JuMP |
| | Matlab |
| | R Studio/R |
| | Octave |

| | |
|---|---|
| | Python |

## REFERENCES


1. Agrawal, A., Harhalakis, G., Minis, I., & Nagi, R. (1996). ‘Just-In-Time’ Production of Large Assemblies. IIE Transactions. 10.1080/15458830.1996.11770710
2. Humphreys, J. (2024, July 5). *Production planning and scheduling for manufacturing*. Retrieved July 11, 2024, from https://katanamrp.com/blog/production-planningscheduling/
3. Landau, P. (2024, April 24). *Production Scheduling Basics: Creating a Production Schedule*. Retrieved July 11, 2024, from https://www.projectmanager.com/blog/production-scheduling
4. MRPeasy Team. (2017, November 2). *MRP System Series #5: What Makes a Good MRP System?* Retrieved July 11, 2024, from https://www.mrpeasy.com/blog/mrpsystem-3/
5. Premier Tech Ltd. (2024). *Operations Scheduling is Key to Success*. Retrieved July 9, 2024, from https://www.ptdigital.com/article/operations-scheduling-key-success
6. Rob. K (Global Head of Capacity Planning in Pharmaceuticals). (2024). *Reviews and Feedback From Real Clients*. https://www.scheduleit.com/reviews.htm
7. Xu, J., & Nagi, R. (2013). Solving assembly scheduling problems with tree-structure precedence constraints: A Lagrangian relaxation approach. IEEE Transactions on Automation Science and Engineering. 10.1109/TASE.2013.2259816