Integrative Project Assignment

sem3-pi-2024-25 - version 1.0

September 30, 2024

Abstract

This document describes the practical work to be developed within the Integrative Project for the 3rd Semester courses: ARQCP, BDDAD, ESINF, FSIAP, and LAPR3. The assignment involves the development of a computer solution to support production management in an industrial facility. This document briefly introduces the problem to be solved, and some fundamental concepts of Production Management, in general, and of Product Engineering, in particular. Finally, the functional and non-functional requirements of the solution to be developed are presented as well as the operational and technical details of the Integrative Project.

Table 1: Version History

Version	**Description**
1.0	Kick-off version

Contents

1 Integrative Project

In this project, students should be able to analyze, design, and implement a computer-based solution to support production management in industrial facilities, focused on short-run production and project-based manufacturing. This project serves as a proof-of-concept covering a set of relevant modules for production management, such as Product Engineering, Production Planning, and Production Control. To develop these modules, it will be necessary, among other aspects, to support the appropriate management of the plant floor, products, machines, and operations; to manage orders and production orders within the production plan context; to develop tools for simulating production scenarios; to support project management comprising the definition of tasks and critical paths; to monitor machines in a production context; and to plan the equipment preventive maintenance.

In line with the best practices learned in the course, particularly in the subjects Computer Architecture (ARQCP), Databases (BDDAD), Information Structures (ESINF), Applied Physics (FSIAP), and Laboratory-Project III (LAPR3), an iterative and incremental process will be used. Therefore, an agile approach based on SCRUM will be applied in team management, structured in three sprints of four weeks each.

The solution to be developed should consist of a set of applications created in PL/SQL, Java, and C/Assembly, depending on the requirements. To enhance the solution's maintainability and align with software development best practices, the implementation must follow a Test-Driven Development (TDD) approach.

2 Description of the Problem

This section describes the problem to be solved: the development of a software solution that addresses critical aspects of planning, managing, and controlling production in an industrial facility. The industrial management problem is thus presented, along with its specificities, and the most relevant associated concepts, particularly aspects related to: Product Engineering, such as the structure of the facility (e.g., products, machines, operations, and layouts); Production Planning, processing customer orders, generating production orders; and Production Control, such supervising machine operation and scheduling preventing maintanence.

2.1 Product Engineering

The company KeepItSimple Solutions (KS) is specialized in creating software solutions for a wide range of businesses. Recently, an opportunity was identified to develop a solution for the planning, management, and production control of flexible industrial units suited to short-run production and/or project-based manufacturing. The system could be used across multiple sectors (e.g., furniture, metalworking, cork, footwear, cutlery). As a proof of concept, the goal is to develop a solution focused on Production Management.

Product Engineering encompasses the activities related to the conception and design of the product and its components, as well as load analysis and planning. These elements are crucial for determining the products/projects' technical and economic feasibility.

2.2 Factory Facilities

The factory plant is designed according to multiple factors, such as: type of industry, safety issues, common sequences of operations, efficiency concerns, space limitations resulting from the organic evolution of the facility over time, access to raw materials warehouses, components, and/or finished products (see figure 1).

In a factory, one or numerous products can be produced, depending on the market demand, optimization, production costs, and flexibility intended for the facility. The classification of the facilities can vary from "Process" (a limited number of products, high optimization) to Project (unique products, typically with complex structures, and long lead times), including JobShops (a wide variety of products, short runs, high flexibility). Flow-shop is a specific case of a JobShop (where products are limited to a restricted number of product families). In the context of the Integrative Project, the focus is on Project-and-JobShop-oriented facilities.

2.3 Products, Components and Raw Materials

The items produced in a manufacturing facility can be described using a tree-like structure (see figure 2), named BOM (Bill of Materials). The tree represents final products (root), raw materials (leaf nodes), and components (intermediate nodes), as well as the required quantities of each of those elements to produce the final product (see figure 3; note that, in this diagram, the assembly operations are represented on the branches).

Figure 1: Manufacturing Plant

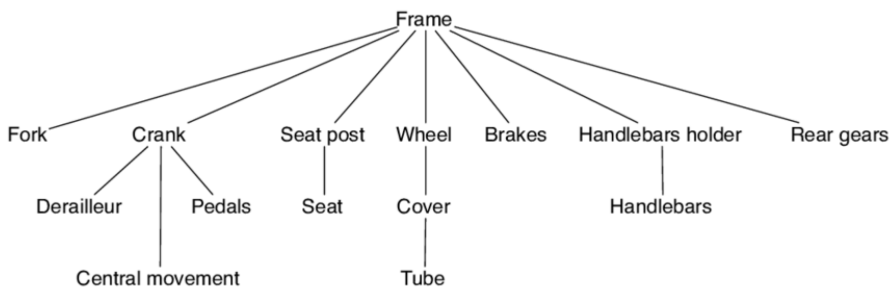

Figure 2: Product Structure of a Bicycle

It is relevant to note that the classification is not rigid; it depends on the manufacturing process. For example, in producing a bicycle, the factory uses a component (e.g., front brake) that was a final product in another manufacturing unit. Similarly, raw material is the result of a Process industry that performs one or more operations of extraction, refining, processing, and packaging of that raw material. The raw material in question is the final product for those manufacturing units.

2.4 Operations, stations and layouts

The plant floor comprises a set of stations organized according to the layout (see figure 4 for an example). These stations can be composed of robotic units, automated machines, machines operated by human operators, or human operators' workstations. The stations are capable of performing a set of manufacturing operations that depend on the type of industry; for example, in the furniture industry, the most common operations include cutting, drilling, gluing, assembly, polishing, varnishing, and painting.

A particular product is subjected to several operations with one or more possible sequences throughout its production process. The different sequences can be represented by a tree-like structure known as the Bill of Operations (BOO). Products can be organized into families, where a family consists of products with similar operation sequence.

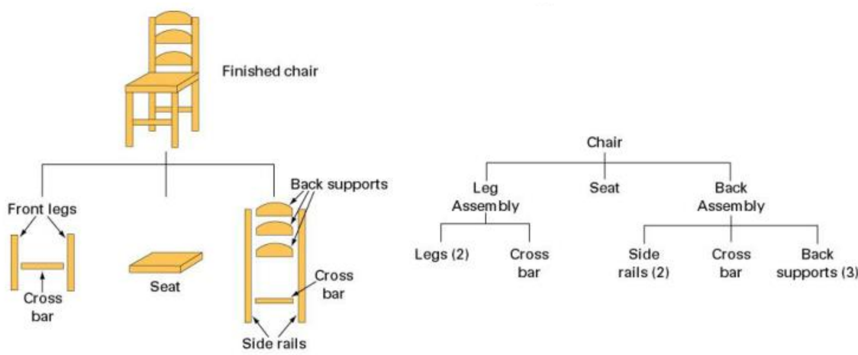

Figure 3: Product Structure of a Chair

2.5 Production Planning

The production planning module generates the production plan for a specific time. The production plan is defined based on the aggregated needs of customer orders, and it is limited by the constraints set in the product tree (BOM) and in the operational ranges of the items to be produced (BOO). The production plan can be instantiated as a set of orders related to the multiple items to be manufactured, and the sequences of operation/station to be used.

2.6 Production Orders

In the production planning module, customer orders are considered, with each order potentially including multiple items. A product can have variants in terms of sizes and colours, meaning that each variant corresponds to a distinct item. However, it is important that each item/variant can be easily associated with the main product, which essentially aggregates all the variants.

Internally, an order has a number, the customer's identification, a list of products, and the expected delivery date and location. To simplify, all items are assumed to be delivered on the same date and location. Customers are identified by their tax identification number (NIF). The customers' names, addresses, and contact details are also stored. Customers can be categorized in two types: individual or company.

2.7 Production Orders

An order can lead to one or more production orders (there may be more than one production order for the same item, but different items must have distinct production orders). When generating production orders, it is essential to consider the BOM (Bill of Materials) defined in Product Engineering, as this structure outlines the components/raw materials needed for producing the item. On the other hand, the production order must include the list of operations necessary for completing the respective item (according to the BOO), and potentially also the stations where those operations will be carried out.

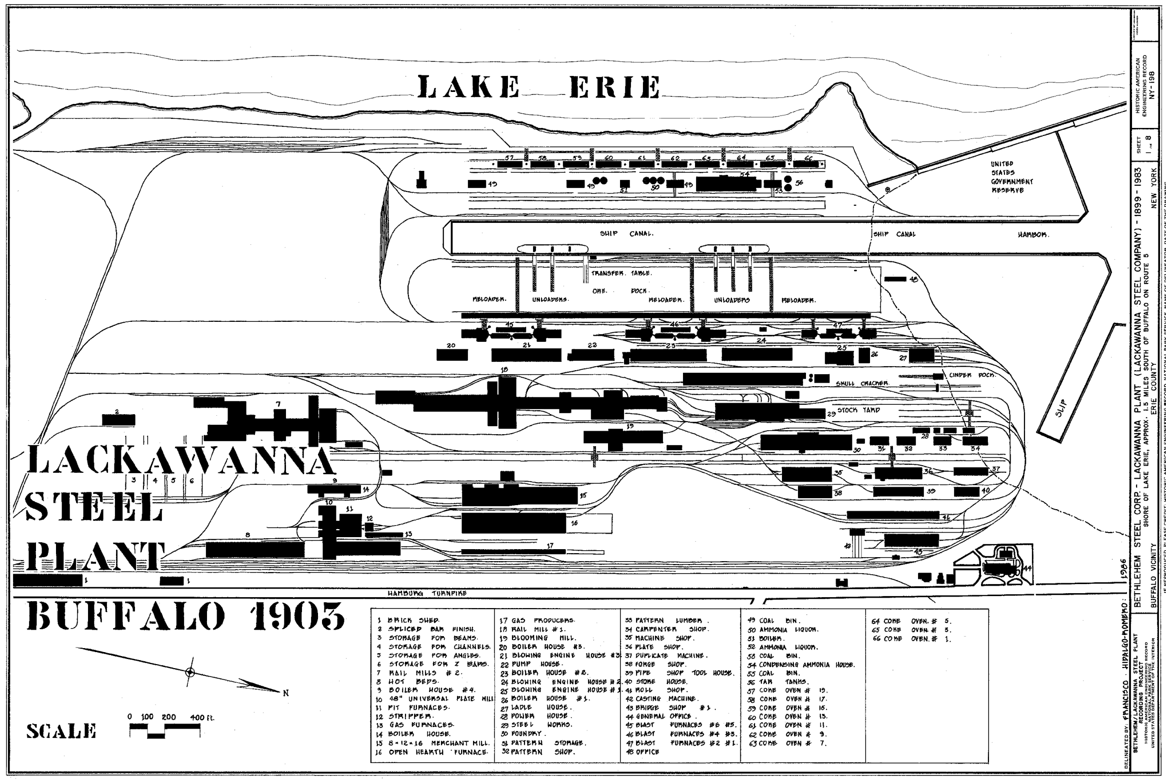

Figure 4: Lackawana Steel Plant

2.8 System Users

Below, the most common user profiles and respective functions are introduced:

- Production Manager: responsible for maintaining information related to products and used raw materials, and for controlling and managing information associated with production orders;
- Plant Floor Manager: responsible for specifying the factory production lines and their respective machines;
- Administrator: responsible for managing (e.g., creating, deactivating) the various system users and their respective permissions, as well as for the overall system configuration.

3 Minimum Viable Product (MVP)

The project to be developed must consider the architecture presented in the diagram of the components in figure 5.

This project aims to develop a Minimum Viable Product, iteratively and incrementally, hence, the assignment is divided into three Sprints:

- Sprint 1 – weeks 3 to 6 – from 1/October to 27/October

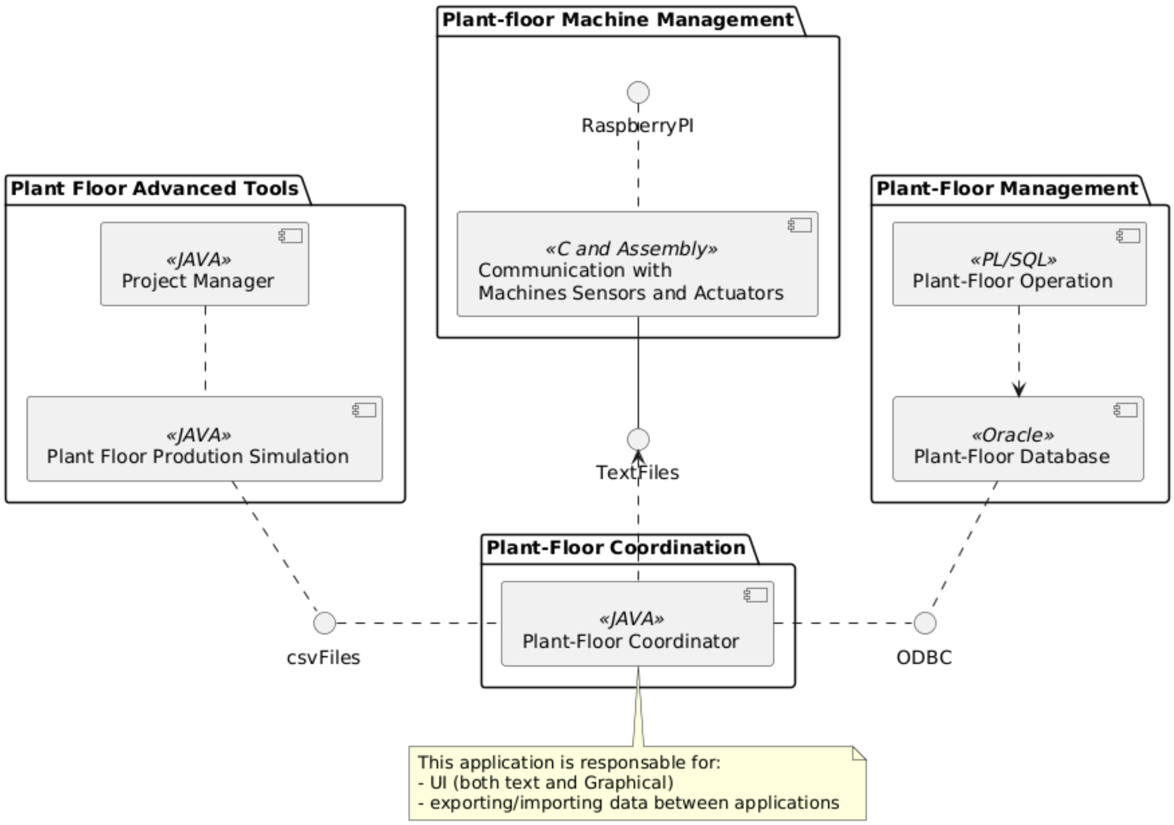

Figure 5: Architecture suggested for the solution to be developed

- Sprint 2 – weeks 7 to 10 – from 28/October to 24/November
- Sprint 3 – weeks 11 to 14 – from 25/November to 11/January

A description of the MVP is provided for each sprint. The teams must follow the provided User Stories (US) and consider their sequencing and respective dependencies. By the end of each sprint, each team should be able to meet the specified requirements. The teams should be able to add the US to the Backlog, estimate them appropriately, and distribute them among team members. For easier reading, the US are separated by Course Unit (Subject).

3.1 Databases (BDDAD)

In this component, a Database should be designed to model an industrial facility that would support the following User Stories (USs).

3.1.1 Sprint 1

- USBD01 - As a Product Owner, I want a data dictionary/glossary to be created.
- USBD02 - As a Product Owner, I want the relational model to be created (logical level).

  Acceptance criteria:

  - The data model should cover the industrial facility's activity described in the document, as well as any additional requirements resulting from the provided user stories.
  - Minimum expected requirement: a complete model in Visual Paradigm, including all connections and constraints, which allows the physical model script to be automatically generated.
  - Minimum requirement above the expected: the presentation of a conceptual model developed in Visual Paradigm in addition to the expected level.

- USBD03 - As a Product Owner, I want the relational model to be instantiated (physical level).
  - It will be shown on Oracle LiveSQL.
  - Minimum expected requirement: automatic generation from Visual Paradigm (centralized change management).

- USBD04 - As a Product Owner, I want to import data from a legacy system and deliver it on a spreadsheet.
  - Minimum expected requirement: manual creation of the data input scripts.
  - Minimum requirement above the expected: automatic generation of SQL input code from the spreadsheet (e.g., Excel formulas, scripts in any other language, etc.).

- USBD05 - As a Production Manager, I want to know, for each product, the orders to be delivered (customer, product, quantity, date) within a given time frame.

- USBD06 - As a Production Manager, I want to know the types of workstations used in a given order.

- USBD07 - As a Production Manager, I want to know the materials/components to be ordered to fulfill a given production order, including the quantity of each material/component.

- USBD08 - As a Plant Floor Manager, I want to know the different operations the factory supports.

- USBD09 - As a Plant Floor Manager, I want to get the operations sequence as well as get the respective type of workstation, from a BOO of a given product.

  Acceptance Criteria for USBD05 to BD09:

  - Minimum acceptance criteria: only the User Stories with data allowing their proper functioning will be evaluated.
  - Minimum expected requirement: demonstrated with data imported from the legacy system.
  - Minimum requirement above the expected: demonstrated with data provided for Sprint 1 evaluation.

3.2 Information Structures (ESINF)

In the Product Engineering process within Production Management, planning and evaluating production capacity is of great importance. Simulation tools generate statistical information that allows for the quick identification of relevant concepts such as machine utilization, time spent on operations, bottlenecks in the production flow, and underutilized resources. Consider a factory where simple items that do not require assembly are produced; that is, simple items like chair legs and tabletops, which will be subsequently joined together (e.g. on a plant floor) or sold in that format (e.g. marketing model IKEA-type). In its production, each item is subject to a set of sequential operations (e.g., cutting, drilling, polishing, painting, labelling, inspection). These operations are carried out by machines/stations installed in the factory. There may be different machines performing the same operation, possibly with different execution times. At this stage, for simplification, a given machine only performs one operation. These stations can be robots, automated machines, or operated by humans.

3.2.1 Sprint 1

Consider the development of a basic simulation tool that consumes the following files:

```
- artigos.cvs with the format:
<id_item, priority, name_oper1, name_oper2, ...,  name_operN>

- maquinas.cvs with the format:
<id_machine, name_oper, time>
```

- USEI01 - Define the adequate data structures to store the information imported from the files.
- USEI02 - Implement a simulator that processes all the items according to the following criteria:
  - AC1: The simulator should create a preliminary queue for each operation, containing all the items, whose next operation (according to the sequential production process) is that of the specified queue.
  - AC2: The items in the queue should be assigned based on the processing availability to the available machine capable of performing the required operation faster, in the order of their entry into the queue.
- USEI03 - Calculate the total production time for the items.
- USEI04 - Calculate execution times by each operation.
- USEI05 - Present a list of machines with total time of operation, and percentages relative to the operation time and total execution time, sorted in ascending order of the percentage of execution time relative to the total time.
- USEI06 - Present average execution times per operation and the corresponding waiting times.

- USEI07 - Produce a listing representing the flow dependency between machines. The listing should be sorted in descending order of processed items. Consider the following example: the items a, b, c, d, e were processed in the machines m1, m2, m3, m4, m5:

  ```
  a: m1 -> m5
  b: m1 -> m2 -> m4 -> m5
  c: m1 -> m2 -> m3 -> m5
  d: m1 -> m4 -> m3
  e: m1 -> m3 -> m5
  ```

  After the complete processing of these items, the following listing should be produced:

  ```
  m1 : [(m2,2),(m5,1),(m3,1),(m4,1)]
  m2 : [(m4,1),(m3,1)]
  m3 : [(m5,2)]
  m4 : [(m5,1),(m3,1)]
  ```

- USEI07 - Consider an improvement to the simulator developed in USEI02 that takes into account a processing order based on priority.

  Acceptance criteria:

  - AC1 - The items in the queue should be assigned, according to their priority (high, normal, low) to the available machine that can perform the required operation the fastest.
  - AC2 - Statistical measures should be produced similarly for this variant of the simulator.

3.3 Laboratory-Project 3 (LAPR3)

3.3.1 Sprint 1

- USLP01 As a Product Owner, I want the domain model to be created (conceptual level). This model will be an essential communication element between all the stakeholders. The domain model diagram is a "dynamic" document, which should continuously reflect the stakeholders' shared understanding of the domain.

  Acceptance Criteria:

  - The data model should cover the activity of the industrial facility, excluding the simulation tool.
  - What is expected: a "basic" model shared in Visual Paradigm or the documentation in GitHub (e.g., PlantUML).

- USLP02 As a Product Owner, I intend to present the simulation tools' functionality and results through a user-friendly text-based interface.

- USLP03: As a Product Owner, I want the visualize the Product Struture (graphically) for a given article.

3.4 Non-functional Requirements

This section describes some of the non-functional requirements that must be considered in the implementation of the project.

- The validation of business rules that must be followed during data registration and updating.
- The database will be the main repository of information for the system and must reflect the necessary data integrity. The information should be persisted in a remote DBMS (Database Management System).
- To enhance interoperability between existing or developing systems, the main application will be developed in Java. However, some components will need to be developed in other languages; the creation and management of the database will use PL/SQL and the interaction with stations will be developed in C/Assembly. A significant part of the integration will be carried out through files.
- The class structure should be designed to allow for easy maintenance and the addition of new functionalities, in line with best practices in object-oriented programming (OOP).
- During the system development, the team must: (i) adopt best practices for identifying requirements, and for OO software analysis and design; (ii) adopt recognized coding standards (e.g., CamelCase); (iii) use Javadoc to generate useful documentation for Java code.
- All the images/figures produced during the software development process should be recorded in SVG format.